Scenery

Your component library already has the design. Scenery turns it into the video.

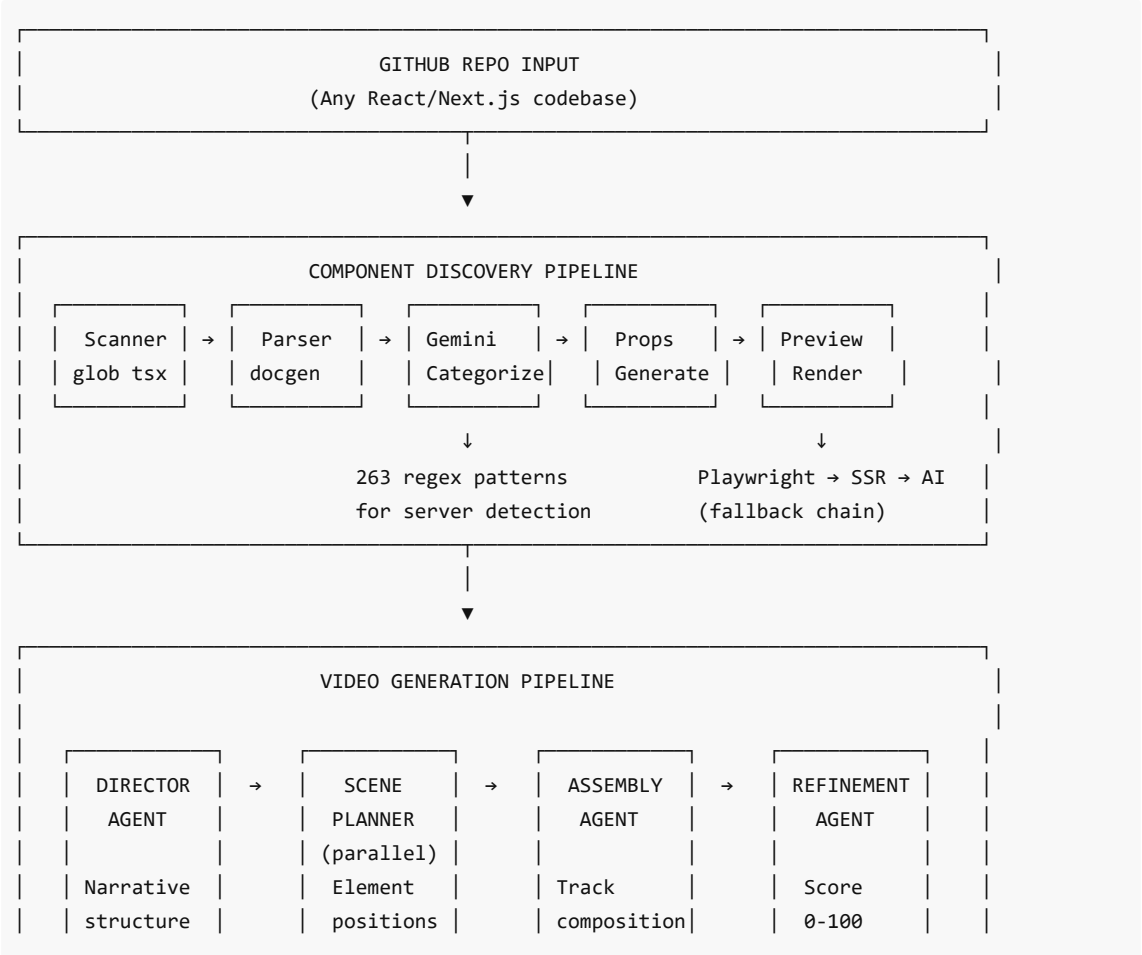
Live Demo · 8 Gemini Integrations · Architecture

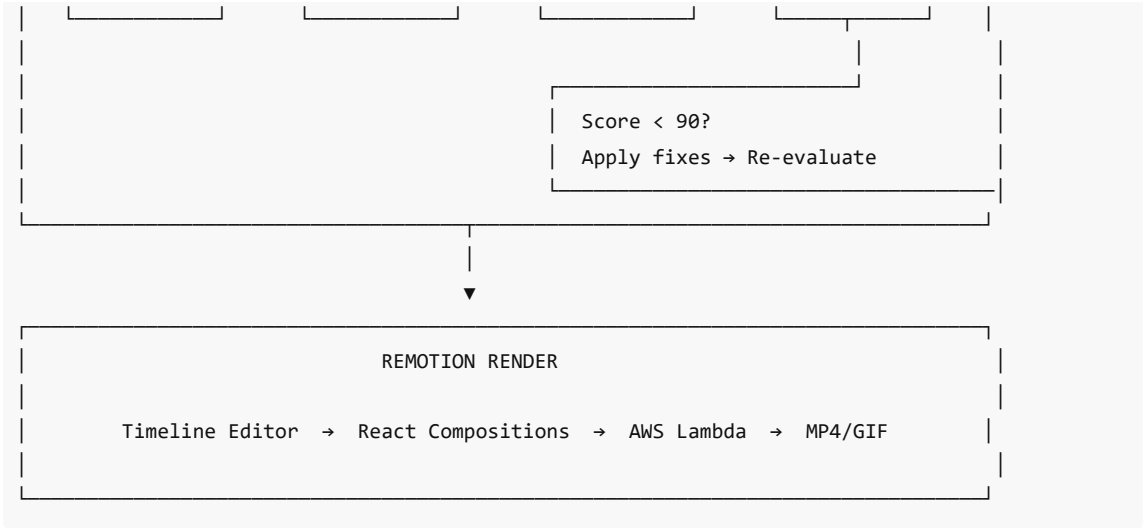
What This Does

Point Scenery at a GitHub repo. It discovers your React components, renders them in a real browser, and generates professional demo videos—complete with cursor interactions, spring animations, and AI voiceover. When your components update, videos stay in sync.

Built for the Gemini 3 Hackathon.

How It Works





Gemini 3 Integrations (8 Total)

Every integration uses structured output with JSON schemas. No prompt-and-pray.

#	Integration	Model	What It Does	Key Feature
1	**Component Categorization**	Gemini 3 Pro	Classifies UI components (button, card, form, etc.) into 27 categories with video showcase strategy	`responseSchema` with Zod validation
2	**Demo Props Generation**	Gemini 3 Pro	Generates realistic props from TypeScript interfaces	3-tier fallback: Storybook → AI → defaults
3	**Server→Client Transform**	Gemini 3 Pro	Converts async Server Components to client-safe code	Handles Prisma, Drizzle, next-auth, 15+ ORMs
4	**Tailwind→Inline CSS**	Gemini 2.0 Flash	Converts all Tailwind classes to inline styles	Portable, framework-agnostic previews
5	**AI Preview Fallback**	Gemini 3 Pro	Generates preview HTML when bundling fails	`thinkingBudget: 5000` for reasoning
6	**Director Agent**	Gemini 3 Pro	Plans video narrative structure and scene breakdown	Function calling with `create_video_plan`
7	**Scene Planner Agent**	Gemini 3 Pro	Designs element positions, animations, timing	15+ tools, runs scenes in parallel
8	**Refinement Agent**	Gemini 3 Pro	Scores composition quality (0-100), suggests fixes	Iterative loop until score ≥90
9	**TTS Voiceover**	Gemini 2.5 Flash TTS	Generates voiceover audio	`responseModalities: ['AUDIO']`, 5 voices

Integration Code Examples

Structured Output (all integrations):

```
// lib/ai/video-generation/safe-generate.ts
const response = await ai.models.generateContent({
  model: 'gemini-3-pro-preview',
  contents: prompt,
  config: {
    responseMimeType: 'application/json',
    responseSchema: toJsonSchema(zodSchema),  // Guarantees valid JSON
    thinkingConfig: { thinkingBudget: 5000 }  // Extended reasoning
  }
});
```

Function Calling (Director & Scene Planner):

```
// lib/ai/video-generation/director-agent.ts
const VIDEO_PLAN_TOOL = {
  name: 'create_video_plan',
  parameters: {
    type: Type.OBJECT,
    properties: {
      title: { type: Type.STRING },
      scenes: {
        type: Type.ARRAY,
        items: {
          type: Type.OBJECT,
          properties: {
            type: { enum: ['intro', 'feature', 'tutorial', 'outro'] },
            durationPercentage: { type: Type.NUMBER },
            componentName: { type: Type.STRING },
            animationIntensity: { enum: ['low', 'medium', 'high'] }
          }
        }
      }
    }
  }
};
```

TTS Audio Generation:

```
// lib/ai/tts.ts
const response = await ai.models.generateContent({
  model: 'gemini-2.5-flash-preview-tts',
  contents: [{ role: 'user', parts: [{ text: narrationScript }] }],
  config: {
    responseModalities: ['AUDIO'],
    speechConfig: {
      voiceConfig: {
        prebuiltVoiceConfig: { voiceName: 'Kore' }  // or Charon, Fenrir, Aoede, Puck
      }
    }
```

```
  }
});
```

Architecture Deep Dive

Component Discovery Pipeline

Location: `lib/component-discovery/`

1. Scanner (`scanner.ts`)

- Globs `**/*.tsx` and `**/*.jsx` , excludes node_modules, tests, stories
- Returns file paths for parsing

2. Parser (`parser.ts`)

- Primary: `react-docgen-typescript` extracts typed props
- Fallback: Regex-based parser for async components, arrow functions
- Handles compound components (`Card.Header` , `Form.Field`)

3. Server Component Detection (`ssr-preview.ts`)

263 regex patterns across 15 categories:

Category	Example Patterns	Count
Async	`export default async function`, `await`	4
Database ORMs	Prisma, Drizzle, Mongoose, Supabase, pg, mysql2	21
Auth Libraries	next-auth, @clerk/nextjs/server, lucia	8
Node.js Built-ins	fs, path, crypto, child_process	12
Next.js Server	next/headers, cookies(), redirect()	10
File System	readFileSync, writeFile, existsSync	6
And 9 more...	Email, payment, CMS, analytics, tRPC	~200

If detected as server component → Gemini transforms to client-safe code.

4. Preview Rendering (3-tier fallback)

```
┌─────────────────────────────────────────────────────────────┐
│  TIER 1: Playwright Browser Rendering (95% accuracy)        │
│  • Bundles with esbuild (50+ library mocks)                 │
│  • Renders in real Chromium on Fly.io worker                │
│  • 4-attempt recovery with progressive simplification       │
└─────────────────────────────┬───────────────────────────────┘
                              │ Fails?
                              ▼
┌─────────────────────────────────────────────────────────────┐
│  TIER 2: Server-Side Rendering (70% accuracy)               │
│  • renderToStaticMarkup via esbuild                         │
```

```
│  • 5-second timeout                                          │
└──────────────────────────────┬──────────────────────────────┘
                               │ Fails?
                               ▼
┌──────────────────────────────────────────────────────────────┐
│  TIER 3: AI-Only Generation (50% accuracy)                   │
│  • Gemini generates HTML from source code                    │
│  • thinkingBudget: 5000 tokens                               │
└──────────────────────────────────────────────────────────────┘
```

5. Tailwind Conversion

All classes converted to inline styles for portable, framework-agnostic previews.

Video Generation Pipeline

Location: `lib/ai/video-generation/`

Director Agent (`director-agent.ts`)

Creates high-level video plan with narrative structure.

Frame Budget Formula:

```
totalFrames = durationSeconds × 30
├── Hook:     15% of frames
├── Setup:    15% of frames
├── Showcase: 55% of frames
└── CTA:      15% of frames
```

Scene Types:

Type	Min Frames	Max Frames	Purpose
intro	60	120	Hook + title
feature	120	240	Showcase component
tutorial	180	360	Interactive demo
transition	15	30	Scene connector
outro	60	90	CTA + closing

Scene Planner Agent (`scene-planner-agent.ts`)

Translates intents to concrete specifications. **Runs all scenes in parallel.**

Critical Concept — Relative Keyframes:

```
// WRONG: 10 seconds to fade in
{ frame: 0, opacity: 0 }, { frame: 300, opacity: 1 }

// RIGHT: 0.67 seconds to fade in
```

```
{ frame: 0, opacity: 0 }, { frame: 20, opacity: 1 }
// Frame 0 = when THIS element appears, NOT video start
```

Layer Order (z-index):

```
0: background   (gradients, solid fills)
1: shapes       (rectangles, circles)
2: device-frames (phone, laptop containers)
3: components   (React UI components)
4: text         (all text overlays)
5: cursor       (tutorial cursor, always on top)
```

Spring Animation Presets:

Preset	Damping	Stiffness	Mass	Use Case
smooth	200	100	1	Professional
snappy	200	200	0.5	Quick, responsive
heavy	200	80	5	Slow, dramatic
bouncy	100	150	1	Playful
gentle	300	60	2	Soft, subtle

Assembly Agent (`assembly-agent.ts`)

Deterministic (no LLM). Transforms detailed scenes into editor-ready composition.

- Fixes keyframe timing mistakes (rescales if max frame > 90)
- Normalizes keyframe format for editor
- Organizes tracks by z-order
- Validates composition (no empty items, items within duration)

Refinement Agent (`refinement-agent.ts`)

Quality verification with scoring loop.

Scoring Weights:

Category	Weight	What It Measures
Visual Composition	30%	Positioning, hierarchy, no overlaps
Timing	25%	Min 90 frames on screen, smooth transitions
Narrative Flow	25%	Intro→Setup→Showcase→CTA structure
Animation Quality	15%	Spring animations, stagger effects
Accessibility	5%	Readable text, contrast ratios

Score Thresholds:

Score	Action

90-100	Ship it
75-89	Apply 1-2 fixes
60-74	Apply 3-5 fixes
40-59	Major patches + manual review
0-39	Regenerate from Director

Technical Decisions

Why 263 Regex Patterns?

Next.js 13+ Server Components use async/await, database calls, and Node.js APIs that crash in browsers. We detect **every pattern** that makes a component server-only:

```
// lib/component-discovery/ssr-preview.ts
const serverPatterns = [
  /export\s+default\s+async\s+function/,
  /from\s+['"]@prisma\/client['"]/,
  /from\s+['"]drizzle-orm/,
  /from\s+['"]next-auth/,
  /from\s+['"]@clerk\/nextjs\/server['"]/,
  /params\s*:\s*Promise\s*</,  // Next.js 15 async params
  // ... 257 more
];
```

Why Fly.io Playwright Workers?

Component preview rendering needs real browser execution for accuracy. Playwright in a container gives us:

- Headless Chromium with full CSS/JS support
- Isolated from main app (scales independently)
- Bearer token authentication
- Health checks and connection reuse

```
// playwright-worker/src/server.ts
app.post('/render', async (req) => {
  const { bundledJs, componentName, props, timeout } = req.body;
  const page = await browser.newPage();
  await page.setContent(htmlTemplate);
  await page.evaluate(bundledJs);
  return await page.screenshot();
});
```

Why Self-Improving Refinement Loop?

First-pass AI output is rarely perfect. The refinement loop:

1. Scores composition against 5 weighted criteria

2. If score < 90, identifies specific issues with fix recipes
3. Applies auto-fixes
4. Re-evaluates
5. Picks best version across all iterations

Top 5 Failure Modes & Fixes:

Issue	Detection	Fix
Text overlapping component	Text at y: 0.4-0.6 when component at y: 0.5	Move text to y: 0.10
Animation too fast	Entrance in < 10 frames	Extend to 15 frames
Elements appearing simultaneously	Same offsetFrames	Add 12-frame stagger
Component not visible	Overflows device frame	Switch to larger frame
No clear narrative	Missing intro/outro	Flag for regeneration

Why Structured Output with Zod?

Every Gemini call uses JSON schema with Zod validation:

```
// lib/ai/video-generation/safe-generate.ts
const result = await safeGenerate({
  schema: refinementResultSchema,  // Zod schema
  prompt: analysisPrompt,
  maxRetries: 2,
  thinkingBudget: 5000
});

// If Zod validation fails, error is appended to prompt for retry
// 3 total attempts before giving up
```

Benefits:

- 100% parse success (no malformed JSON)
- Type inference from schema
- Automatic retry with error feedback

Why 4-Attempt Render Recovery?

Component rendering fails for many reasons. Progressive simplification:

Attempt	Strategy	What It Does
1	`fix-props`	Analyze error, add missing definitions
2	`simplify`	Remove complex nested objects, use flat structures
3	`minimal`	Aggressive mocking, all required props only
4	`minimal`	Placeholder div, mock ALL external calls
5+	`skip`	Mark as unfixable, move on

```
// lib/component-discovery/render-recovery.ts
const failureCount = incrementFailureCount(componentName);
if (failureCount >= 5) return { shouldSkip: true };
```

Tech Stack

Layer	Technology	Why
AI	Gemini 3 Pro, Gemini 2.5 Flash TTS	8 integrations, structured output, function calling, TTS
Frontend	Next.js 15, React 19	App Router, Server Actions, React 19 features
Video Engine	Remotion 4	React-based video composition, spring physics
Video Export	Remotion Lambda	Parallelized AWS Lambda rendering
Component Preview	Playwright, esbuild	Real browser rendering, fast bundling
State	Zustand + Zundo	Lightweight state with undo/redo
Forms	React Hook Form + Zod	Type-safe validation
Database	Supabase (PostgreSQL)	Auth, projects, compositions
Hosting	Fly.io	2 apps (main + playwright worker), auto-scale
UI	shadcn/ui + Radix	Accessible components

Getting Started

Prerequisites

- Node.js 20+
- npm or yarn
- Supabase account
- Gemini API key

Local Development

```
# Clone
git clone https://github.com/Arty2001/scenery-gemini3-hackathon.git
cd scenery-gemini3-hackathon

# Install
npm install

# Configure
cp .env.example .env.local
```

```
# Run
npm run dev  # http://localhost:3000
```

Scripts

Command	Description
`npm run dev`	Start dev server
`npm run build`	Production build
`npm run lint`	ESLint check
`npm run deploy:remotion`	Deploy Remotion Lambda

Environment Variables

```
# REQUIRED
NEXT_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_supabase_anon_key
GEMINI_API_KEY=your_gemini_api_key

# OPTIONAL: Cloud video rendering
REMOTION_AWS_REGION=us-east-1
REMOTION_LAMBDA_FUNCTION_NAME=remotion-render-function
REMOTION_SERVE_URL=https://your-remotion-serve-url.com

# OPTIONAL: Accurate preview rendering
PLAYWRIGHT_WORKER_URL=https://scenery-playwright.fly.dev
PLAYWRIGHT_WORKER_SECRET=your_shared_secret
```

Project Structure

```
scenery/
├── app/                          # Next.js App Router
│   ├── api/
│   │   ├── ai/chat/              # Main chat + video generation (streaming)
│   │   ├── ai/process-html/      # HTML processing (Tailwind→CSS)
│   │   ├── ai/tts/               # Text-to-speech voiceover
│   │   ├── projects/[id]/        # Project CRUD + component discovery
│   │   └── export/progress/      # Video render job tracking
│   ├── auth/                     # Supabase auth flows
│   └── protected/                # Dashboard, editor, projects
│
├── lib/
│   ├── ai/
│   │   ├── client.ts             # Gemini client (singleton, 900s timeout)
│   │   ├── models.ts             # Model selection (Pro, Flash, TTS)
│   │   ├── tts.ts                # Voiceover generation
```

```
│   │   └── video-generation/
│   │       ├── orchestrator.ts   # 4-stage pipeline coordinator
│   │       ├── director-agent.ts # Narrative planning
│   │       ├── scene-planner-agent.ts # Detail specification
│   │       ├── assembly-agent.ts # Composition building
│   │       ├── refinement-agent.ts # Quality scoring loop
│   │       ├── safe-generate.ts  # Retry + Zod validation
│   │       └── shared-constants.ts # Glossary, design tokens
│   │
│   ├── component-discovery/
│   │   ├── scanner.ts            # Find .tsx files
│   │   ├── parser.ts             # Extract props (docgen + fallback)
│   │   ├── analyzer.ts           # Gemini categorization + props gen
│   │   ├── ssr-preview.ts        # Server detection (263 patterns)
│   │   ├── playwright-client.ts  # HTTP client to worker
│   │   ├── browser-bundler.ts    # esbuild with 50+ mocks
│   │   ├── render-recovery.ts    # 4-attempt progressive simplification
│   │   └── storybook-extractor.ts # CSF2/CSF3 story args
│   │
│   ├── composition/
│   │   ├── types.ts              # Track, Item, Animation types
│   │   ├── store.ts              # Zustand state
│   │   └── schema.ts             # Zod validation
│   │
│   └── actions/                  # Server actions
│       ├── projects.ts           # Project CRUD
│       ├── components.ts         # Component management
│       └── export.ts             # Remotion Lambda export
│
├── components/
│   ├── remotion/                 # Video composition
│   │   ├── compositions/         # Remotion React components
│   │   ├── animation/            # Spring physics
│   │   └── transitions/          # Scene transitions
│   ├── video-editor/             # Timeline editor UI
│   ├── chat/                     # AI chat interface
│   └── ui/                       # shadcn/ui components
│
├── playwright-worker/            # Separate Fly.io app
│   ├── src/server.ts             # Fastify + auth
│   └── src/renderer.ts           # Chromium management
│
└── supabase/migrations/          # 16 SQL migrations
```

Limitations & Known Issues

Current Limitations

Limitation	Reason	Workaround

React-only	Parsing relies on react-docgen-typescript	No Vue/Svelte support planned
Public repos only	GitHub OAuth scopes not requested	Private repo support possible with token
English TTS only	Gemini TTS limitation	N/A
No live preview editing	Remotion player is read-only	Edit in timeline, preview renders
30fps fixed	Simplifies frame math	Could be configurable

Known Issues

Issue	Impact	Status
Rate limits on heavy usage	Gemini API quota exhaustion	Added fail-fast detection
Complex components may fail to render	Some edge cases in bundling	4-attempt recovery helps
Server components with dynamic imports	Not all patterns detected	Add patterns as discovered
Large repos slow to scan	Many files = slow glob	Consider incremental scanning

Not Implemented

- **Collaborative editing** — Single user only
- **Version history** — No composition versioning
- **Custom fonts** — Uses Google Fonts via Remotion
- **Audio tracks** — Voiceover only, no music

Deployment

Fly.io (Recommended)

```
# Main app
fly launch
fly secrets set GEMINI_API_KEY=xxx SUPABASE_URL=xxx SUPABASE_ANON_KEY=xxx

# Playwright worker (separate app)
cd playwright-worker
fly launch
fly secrets set WORKER_SECRET=xxx
```

Remotion Lambda

```
npm run deploy:remotion
# Follow Remotion Lambda setup: https://remotion.dev/docs/lambda/setup
```

Built for the Gemini 3 Hackathon

by Athavan Thambimuthu