ELEVEN VIEWS

FREE RESOURCE

Real Images in AI Scenes

The swipe file for putting photos and screenshots into Remotion motion graphics without that pasted-on look.

Here is the moment a video stops looking like AI made it: a real screenshot slides in, gets a soft drop shadow, picks up a faint grain pass, and parallaxes a few pixels behind a headline. Suddenly the thing has weight. It looks like a person sat in an editor and placed it. Pure kinetic type, no matter how clean, reads as a template after about eight seconds. Your eye clocks it. Real imagery is the jump that buys you another thirty. This file is eight scene recipes, the asset-prep rules that keep Claude from handing you a gray placeholder box, the green-screen cleanup that kills the cutout halo, and a worked 20-second build from three real screenshots. Copy, paste, ship.

atlas.elevenviews.io

01

Why a real image is the cheapest upgrade you can buy

A motion-graphics-only video has one fatal tell: every element was born inside the renderer. Same color space, same anti-aliasing, same flat lighting. Your brain notices the uniformity before it can name it, and the verdict is "made fast, made cheap."

A real photo or screenshot breaks that uniformity. It carries lens information, real shadows, JPEG texture, the slight imperfection of something that exists. Drop one into a scene and the whole frame gains depth, because now the viewer is comparing a rendered layer against a captured one.

The catch is that a badly placed image makes it worse, not better. A screenshot floating dead-center on a solid background with hard 90-degree edges looks like a PowerPoint from 2009. The fix is never the image itself. It is the three things around it: a shadow that implies a light source, a grain pass that matches the image texture to the scene, and motion that refuses to let the image sit still and announce itself.

Three numbers to keep in your head as you read:
- A drop shadow at 8 to 12 percent opacity reads as grounded. At 40 percent it reads as a sticker.
- An image that moves 6 to 10 percent of its own width over a scene feels alive. Static feels pasted.
- A grain overlay at 4 to 7 percent opacity unifies a flat image and a rendered background. You should not consciously see it.

02

Asset prep: the rules that stop Claude handing you a placeholder

Most "Claude gave me a gray box" problems are not Claude. They are the asset. Fix the file before you fix the prompt.

Resolution. Shoot or export at 2x your render target. For a 1080p (1920x1080) timeline, your image should be at least 2160px on the long edge. For 4K, 4320px. Remotion scales down cleanly and scales up like a crime scene. A screenshot taken on a Retina display is already 2x, so capture at full window size, not a cropped region.

Aspect and crop. Decide the crop before the scene, not inside it. If the image lives behind text, you want it wider than the frame so you have room to pan. If it is a product reveal, center the subject with 15 to 20 percent breathing room on all sides so the shadow and scale animation have somewhere to go.

PNG vs JPG, the rule that actually matters. Transparent edges (a cut-out product, a headshot, a logo) must be PNG or WebP with alpha. Anything rectangular and full-frame (a photo, a full screenshot) should be JPG at quality 85 to 90, or WebP. A full-frame PNG is just a bigger JPG that slows your render. Use alpha only when you need alpha.

Where files go. This is the part that causes the placeholder box. Remotion serves from the public/ folder, and you reference assets with staticFile(). Put images in public/images/ and reference them as

staticFile('images/hero.png'), not import or a raw path. If you hand Claude a path like /Users/you/Desktop/shot.png, it has no way to bundle that, so it stubs a placeholder. Move the file into public/ first, then give Claude the relative name.

Quick pre-flight before you prompt:
- File is in public/images/
- Long edge is at least 2x the render target
- Transparent? PNG/WebP. Rectangular? JPG q85 to 90
- Filename is lowercase, no spaces (use product-reveal.jpg, not Product Reveal.JPG)

03

The prompt that gets a working scene back, not a stub

The single biggest reason Claude returns a placeholder box instead of your image is that the prompt never told it the file already exists in public/. Claude defaults to defensive: if it is unsure the asset is real, it stubs a colored div so the render does not crash. You have to remove that doubt explicitly.

The pattern that works every time: state the exact filename, confirm it lives in public/images/, name the dimensions, and tell Claude to use staticFile() with Img from remotion. Then describe the motion in plain language with timing in frames or seconds. Give it the fps and duration so it does not guess.

The difference in output is night and day. "Make a scene with my screenshot" gets you a gray box with the word IMAGE in it. The full prompt in the swipeables below gets you a mounted <Img>, an interpolated transform, and a spring on the entrance. Hand it the filename like you would hand a junior editor a file off the server, not a vague vibe.

04

8 copy-paste scene recipes

Each of these is a description you paste straight into Claude after the base image-scene prompt. They assume 30fps and a file already in public/images/. Adjust durations to taste. All eight are in the swipeables, ready to paste. Here is what each one is for and the one number that makes it land.

1. Ken Burns pan. A still photo that slowly scales and drifts. The move that makes a static image feel like footage. Scale from 1.0 to 1.08 over the full scene, drift the position 4 to 8 percent. Slow is the whole point. If a viewer can see it moving, it is too fast.

2. Screenshot with callout. A UI screenshot that mounts, then a circle or arrow draws onto a specific element with a label. The workhorse for product and app videos. Delay the callout 12 to 18 frames after the screenshot settles so the eye lands on the UI first.

3. Product reveal. A cut-out PNG that springs up from slightly below with a shadow that grows as it rises, implying it is lifting toward a light. The shadow growth is what sells the lift.

4. Photo grid build. Three to six images that pop in on a stagger to build a wall. Great for case studies, portfolios, before/afters. Stagger each tile by 3 to 5 frames. Simultaneous looks like a glitch.

5. Image behind text. A photo sits behind a headline with a dark gradient scrim so the type stays readable. The default for title cards over real footage. Scrim at 35 to 55 percent black, heavier at the

bottom where the text sits.

6. Masked reveal. The image is revealed through an expanding shape (a wipe, a circle, a vertical bar). Feels intentional and designed. Ease the mask, never linear. A linear wipe looks mechanical.

7. Parallax stack. Two or three layers (background photo, midground subject, foreground type) move at different speeds. The single most expensive-looking trick in this file. Background moves 2 percent, midground 5 percent, foreground 10 percent. The speed gap is the depth.

8. Device mockup. Your screenshot composited inside a phone or laptop frame, then the whole device animates in. Buys instant credibility for app demos. Round the screenshot corners to match the device radius (about 18px on a phone at full size) or the illusion breaks.

05

Killing the green-screen halo on product and headshot cut-outs

When you cut a product shot or a headshot off a green background, the giveaway is the halo: a thin green or light fringe tracing the edge where the keyer left spill behind. It is the difference between a clean cut-out and an obvious one.

You have two clean paths. If you shot against actual green screen, key it in DaVinci Resolve before you ever bring it to Remotion: drop the clip on the timeline, add a Qualifier or the 3D Keyer in the Color page, pull the key, then add an Edge spill node and pull saturation down on the fringe. Export as a PNG sequence or a single PNG with alpha. If you did not shoot green and just need a subject cut from a normal photo, run it through a background remover, then do the edge work yourself, because automatic removers always leave a fringe.

The edge cleanup that actually kills the halo, the part everyone skips:
- Contract the matte by 1 to 2 pixels. This eats the outermost ring where the spill lives. In Resolve it is Matte Shrink. In an image editor it is Select > Modify > Contract before you mask.
- Feather by 0.5 to 1 pixel after contracting. A razor-hard edge looks cut out. A 1px feather looks photographed.
- Desaturate the remaining edge. If a green or blue fringe survives, sample it and pull its saturation toward zero so it reads as neutral gray, which the eye forgives.

Then, and this matters, give the cut-out a shadow in the scene. A floating subject with no shadow is the second-biggest tell after the halo. See the next section.

06

The grounding pass: shadow and grain that fake real lighting

A flat image dropped into a rendered scene has no relationship to that scene's light. The grounding pass invents one. Two cheap layers do most of the work.

Shadow. The job of the shadow is to tell the viewer where the light is and that the image has thickness. For a cut-out (product, headshot), use a soft offset shadow: 8 to 16px blur, offset 4 to 8px down and slightly to one side, opacity 10 to 18 percent, color near-black but tinted toward the scene's dominant color, not pure #000. For a full rectangular image like a screenshot, use a tight contact shadow: 12 to

24px blur, 6 to 10px down offset, 12 to 20 percent opacity. The down offset implies overhead light, which is what viewers expect, so it never looks wrong.

Grain. A rendered background is mathematically clean. A real photo has sensor noise and compression texture. Put a single grain layer over the entire composition, both the image and the rendered elements, at 4 to 7 percent opacity in overlay or soft-light blend. Now both layers share a texture, and the seam between captured and rendered disappears. You can generate grain procedurally in Remotion or overlay a looping grain video. One pass over the whole frame, never per-layer.

Optional third layer for hero shots: a subtle vignette, 8 to 12 percent, darkening the corners. It pulls the eye to center and mimics lens falloff. The shadow, grain, and scrim snippet is in the swipeables as a drop-in component.

07

Worked mini-build: 3 screenshots into a 20-second product explainer

Here is the whole thing end to end, so you can see how the pieces stack into a finished cut. The brief: a 20-second explainer for a fictional invoicing app called Ledgerly, built from three real screenshots (dashboard, create-invoice screen, paid-confirmation screen).

Prep (5 minutes). Capture all three at full window on a Retina display, so they land around 2880px wide. Save as JPG q88 into public/images/ as ledger-dashboard.jpg, ledger-create.jpg, ledger-paid.jpg. Lowercase, no spaces.

Scene plan (600 frames at 30fps):
- 0:00 to 0:04 (frames 0 to 120). Title card. ledger-dashboard.jpg behind the headline "Invoicing that closes itself" using the image-behind-text recipe, dark scrim at 45 percent bottom-weighted, plus a slow Ken Burns push from scale 1.0 to 1.05. Already two recipes on one shot.
- 0:04 to 0:10 (frames 120 to 300). Feature one. ledger-create.jpg mounts with a contact shadow, then a callout circle draws onto the "Send" button at frame 150 with the label "One click to send." Screenshot-with-callout recipe.
- 0:10 to 0:16 (frames 300 to 480). Feature two. ledger-paid.jpg in a laptop device mockup that springs in from below, screenshot corners rounded 12px to match the frame. Device-mockup plus product-reveal motion.
- 0:16 to 0:20 (frames 480 to 600). Close. All three screenshots shrink into a staggered photo grid behind the line "See it live," then a CTA. Photo-grid-build recipe.

Grounding pass over everything. One grain layer at 5 percent soft-light across the full 600 frames. One vignette at 10 percent. Every screenshot gets the contact shadow.

Before and after, what changes. Before the grounding pass: three sharp screenshots cutting in and out on a flat dark background. Clean, but it reads as a generated explainer. The screenshots look like they were dropped on top of the scene, because they were. After: the grain ties the screenshot texture to the background, the contact shadows give each screen a surface to sit on, the Ken Burns and parallax keep nothing static, and the device mockup on feature two gives the eye one premium beat. Same three files. The second version is the one a founder pays for.

Render. remotion render src/index.ts Ledgerly out/ledgerly.mp4 at 1080p, or add --scale=2 config for 4K. Drop the MP4 into DaVinci Resolve if you want a final color pass and audio, and you are done.

08

Build Kit: turn this guide into video

Everything above is the spec. This section is how the Desk turns that spec into a finished companion video without you re-explaining anything.

The rule the Desk follows: this PDF is the single source of truth. The free guide tells the build what to make. The Desk reads it, pulls the recipes, and renders. You do not write a separate brief. You hand over this file plus your real images, and the swipeables at the end do the rest.

Here is the loop, start to finish.

1. Drop your assets in one folder. Use the exact naming from the Asset and resource checklist below. The Desk matches filenames to scenes, so a screenshot named dashboard-before.png lands in the scene that asks for it. No guessing.

2. Run the asset prep pass from the guide. Every image gets the resolution, aspect, and PNG-vs-JPG treatment spelled out in the Asset prep rules section above. Screenshots stay PNG. Photos go JPG at quality 90. Anything with a transparent edge gets the green-screen removal and edge cleanup pass before it ever enters a scene.

3. Feed the prompt-to-scene prompt. The guide gives you the exact prompt that turns an image filename into a working Remotion scene. The Desk runs that prompt once per image. Out comes a TSX component that places the image, sizes it to safe margins, and wires the entrance motion.

4. Assemble against the Scene map. The Scene map swipeable is a line-per-scene plan. The Desk builds each scene in order, applies the drop-shadow and grain grounding pass so the real image sits in the motion instead of floating on top of it, and stitches them on the Remotion timeline.

5. Narrate and time. The Companion video narration script is the voiceover, written to length. Record it or send it to your TTS of choice, then snap scene durations to the line breaks. The script is paced so each block lands inside one scene.

6. Render with the recipe. The Render and export recipe gives exact 1080p and 4K settings and the order to run them. Render the 1080p proof first, check the grounding pass held up, then push the 4K master.

The worked mini-build in the guide, three screenshots to a 20-second explainer, is the smallest version of this whole loop. Run that once to prove your asset folder and prompt are clean. Then scale the same loop to the full companion video. Same techniques, more scenes.

09

Your companion video guides

The free PDF you are reading is the blueprint. The paid product is the video track the Desk builds from it: a multi-part guide that shows the work on screen, in Remotion, with real images going in and a

finished scene coming out. You are not buying theory. You are buying the watch-over-the-shoulder version of every recipe in this PDF.

Three parts. Each one ends with you having built something real.

Part 1: Get one real image into a Remotion scene cleanly. The shortest path from a raw screenshot or photo to a scene that does not look pasted on. On screen you watch the asset prep pass happen in real time: a 4000px screenshot brought down to a clean 1920-wide PNG, a photo saved as JPG at quality 90, both placed inside safe margins. Then the prompt-to-scene prompt runs and a working scene appears. You finish Part 1 with one image animated in, grounded with a drop shadow and a grain pass, rendered to 1080p. The thing that stops people, the floating-sticker look, is gone by the end of this part.

Part 2: Cut out, clean up, and ground. The compositing part. On screen: green-screen removal with edge cleanup on a product shot, then the grounding pass that makes a cutout sit in the scene instead of hovering above it. You see the bad version first, the hard halo edge and the too-clean cutout, then the fixes that kill both. You finish Part 2 with a transparent-background subject dropped into a motion background, edges feathered, shadow cast, grain matched. This is the part that separates a clip that reads as real from one that reads as a slideshow.

Part 3: The full build, three screenshots to a 20-second explainer. The worked mini-build from the PDF, done end to end on camera. You watch three raw screenshots become a paced, narrated, 20-second explainer: the scene map laid out, each screenshot prepped and placed, motion and timing snapped to the narration, the grounding pass applied across all scenes, then both renders, 1080p proof and 4K master. You finish Part 3 with a complete short explainer you can post, and a repeatable folder-and-prompt setup you reuse for the next one.

What carries across all three parts: every technique shown is already specced in this PDF. The video just shows you the hands-on-keyboard version so you stop second-guessing whether you did the asset prep right.

Swipe file

Copy, paste, adjust. These are the exact prompts and templates.

BASE IMAGE-SCENE PROMPT (PASTE THIS FIRST)

```
I'm building a Remotion scene. The image already exists at
public/images/FILENAME.jpg (or .png). It is WIDTH x HEIGHT pixels. The composition
is 1920x1080 at 30fps and this scene is N frames long.

Return a single Remotion scene component that:
- Imports { Img, staticFile, useCurrentFrame, useVideoConfig, interpolate, spring }
from 'remotion'
- Renders the image with <Img src={staticFile('images/FILENAME.jpg')} /> (do NOT use
a placeholder div, the file is real and present)
- Uses position: absolute and objectFit: 'cover' so it fills the frame unless I say
otherwise
- Animates as described below

Give me clean, typed code I can paste into a scene file. No explanation, just the
```

component. Here is the motion I want:
[paste one of the recipe blocks here]

RECIPE 1 — KEN BURNS PAN

Slow Ken Burns: scale the image from 1.0 to 1.08 over the full scene, and drift its position from left:0 to left:-6% across the same range, both eased with interpolate and a gentle easing (Easing.bezier(0.4,0,0.2,1)). The move should be barely perceptible. No entrance animation, the image is already on screen at frame 0.

RECIPE 2 — SCREENSHOT WITH CALLOUT

Mount the screenshot centered at 85% width with a contact drop shadow (boxShadow: 0 10px 24px rgba(0,0,0,0.18)). It springs up 24px and fades in over the first 14 frames. Then starting at frame 18, draw an SVG circle (stroke only, 4px, brand color #C9A24B) around a region near coordinates X,Y of the screenshot using strokeDasharray animation so it looks hand-drawn, and fade in a text label 'LABEL TEXT' next to it 6 frames after the circle starts.

RECIPE 3 — PRODUCT REVEAL

The image is a transparent PNG cut-out. It springs up from translateY(60px) to 0 over 20 frames using spring({fps, frame, config:{damping:14}}), fading in over the first 10 frames. Add a soft shadow under it that GROWS as it rises: blur from 8px to 22px and opacity from 0.05 to 0.16 across the entrance, to imply it's lifting toward a light. Center it with 18% padding around the frame.

RECIPE 4 — PHOTO GRID BUILD

I have COUNT images (list filenames). Lay them out in a grid. Each tile pops in on a stagger: tile index i enters at frame i*4, springing from scale 0.8 to 1.0 with spring({fps, frame: frame - i*4, config:{damping:12}}) and fading in over 8 frames. Give every tile an 8px radius and a contact shadow 0 6px 16px rgba(0,0,0,0.15). 24px gap between tiles.

RECIPE 5 — IMAGE BEHIND TEXT

The image fills the frame with objectFit cover and a slow Ken Burns (scale 1.0 to 1.05). Over it, lay a linear-gradient scrim from transparent at top to rgba(0,0,0,0.55) at the bottom 40% so a headline stays readable. Then render the headline 'HEADLINE' bottom-left, large, white, fading and sliding up 20px over frames 8 to 24.

RECIPE 6 — MASKED REVEAL

Reveal the image through an expanding mask. Use a clipPath circle that grows from 0% to 150% radius (centered) over frames 0 to 30, eased with Easing.bezier(0.4,0,0.2,1)

(never linear). The image itself is static behind the mask. After the reveal completes, hold.

RECIPE 7 — PARALLAX STACK

Three layers moving at different speeds to fake depth. Background image: drift x by -2% over the scene. Midground (a cut-out PNG subject): drift x by -5%. Foreground text 'HEADLINE': drift x by -10%. All linear-ish, all driven by interpolate on frame. The speed difference is the entire effect, so keep the layers clearly separated in depth (background darker/blurred 2px, foreground crisp).

RECIPE 8 — DEVICE MOCKUP

Composite the screenshot inside a device. Render a laptop frame (or use a transparent PNG device frame at public/images/laptop.png on top), with the screenshot positioned inside the screen area and corners rounded 12px to match. The whole device springs in from translateY(80px) and scale 0.9 to 1.0 over 22 frames with spring damping 14, plus a contact shadow under the device that grows with the entrance.

GROUNDING PASS — DROP-IN SHADOW + GRAIN + VIGNETTE

Add a grounding pass over my whole composition. Return a <GroundingPass> component I wrap around my scenes that renders, as absolutely-positioned full-frame overlays above everything:
1. A procedural film grain at 5% opacity, mixBlendMode 'soft-light', re-seeded per frame so it shimmers slightly (use a noise texture or random offset on a repeating pattern).
2. A radial vignette: radial-gradient transparent center to rgba(0,0,0,0.12) corners.
Keep both subtle, they should unify captured and rendered layers without being consciously visible. For individual images, here are my shadow values: cut-outs use boxShadow 0 6px 14px rgba(10,10,11,0.16); screenshots use 0 10px 24px rgba(10,10,11,0.16).

GREEN-SCREEN EDGE CLEANUP CHECKLIST

Halo-killing pass for any cut-out (do in this order):
1. Pull the key (Resolve 3D Keyer, or a background remover for non-green shots).
2. CONTRACT the matte by 1–2px — this eats the spill ring. (Resolve: Matte Shrink. Editor: Select > Modify > Contract.)
3. FEATHER by 0.5–1px after contracting — hard edges look cut, 1px looks photographed.
4. DESATURATE any surviving fringe — sample the green/blue edge, pull its saturation to ~0 so it reads neutral gray.
5. Export PNG with alpha into public/images/.
6. In the scene, ALWAYS give the cut-out a shadow (see grounding pass) — a shadowless floating subject is the second-biggest tell after the halo.

COMPANION VIDEO NARRATION SCRIPT (PART 1)

[SCENE 1 — cold open]
You drop a screenshot into your video and it looks wrong. Flat. Pasted on. Like a sticker someone slapped over the motion. Here is how to fix that, in one pass, every time.

[SCENE 2 — the problem named]
Real images fight motion graphics for two reasons. The resolution is off, so the image looks soft or jagged against crisp vector type. And the lighting is off, so the image has no shadow, no grain, nothing tying it to the scene behind it. Fix those two things and the image belongs.

[SCENE 3 — asset prep, resolution]
Start with prep. A screenshot is usually too big. Bring it down to nineteen-twenty wide for a 1080p build, or thirty-eight-forty for 4K. Keep it a PNG. Screenshots have sharp text and flat color, and PNG keeps that edge clean. JPG would smear it.

[SCENE 4 — asset prep, photos]
A photo is different. Save photos as JPG at quality ninety. The file is smaller, the render is faster, and at ninety you will not see the compression. The rule is simple. Sharp interface art, PNG. Real-world photo, JPG.

[SCENE 5 — placement]
Now place it inside safe margins. Do not let the image touch the frame edge. Pull it in so there is breathing room on all four sides. An image that runs into the edge reads as a mistake. An image with margin reads as a choice.

[SCENE 6 — the prompt]
Here is the part that saves you. You do not hand-build the scene. You hand the filename to one prompt. The prompt turns dashboard-before-dot-png into a working Remotion scene: it places the image, sizes it to those safe margins, and wires the entrance. One prompt, one scene.

[SCENE 7 — entrance motion]
Give the image a real entrance. A small scale-up from ninety-six percent, a short fade, an ease that settles instead of snapping. The motion should feel like the image arrives, not like it blinks into place. Subtle wins here.

[SCENE 8 — grounding, the shadow]
Now ground it. First, a drop shadow. Soft, offset down and slightly to one side, low opacity. This one move tells the eye the image sits above the background. Without it, the image floats. With it, the image lands.

[SCENE 9 — grounding, the grain]
Second, a grain pass. The motion background has texture. Your clean image does not. Add a light grain over the image so it shares the same skin as the scene. This is the move almost nobody makes, and it is the one that kills the pasted-on look for good.

[SCENE 10 — render]
Render the 1080p proof first. Watch it back. Did the shadow read? Did the grain tie it in? If yes, push the 4K master. If no, fix the pass, not the whole scene.

[SCENE 11 – payoff]
That is the whole loop. Prep the resolution, pick PNG or JPG, place inside margins, run the prompt, give it an entrance, then ground it with shadow and grain. Same six moves on every image you ever drop in.

[SCENE 12 – close]
This is Part 1: one image, in clean. Part 2 cuts subjects out and grounds them. Part 3 builds a full explainer from three screenshots. The blueprint is in the PDF. Go build the first one.

SCENE MAP FOR THE COMPANION VIDEO

scene | on-screen text | motion note | asset needed
1 | "It looks pasted on." | hard cut to a bad floating screenshot, no shadow, slight jitter to feel wrong | bad-example-floating.png
2 | "Two reasons it fights the motion" | two text lines slide up and stagger in | none, type only
3 | "Resolution: 1920w PNG" | screenshot scales down from oversized to 1920w, crop guides fade | dashboard-raw.png
4 | "Photos: JPG, quality 90" | photo crossfades from PNG label to JPG label, file-size counter ticks down | hero-photo.jpg
5 | "Stay inside safe margins" | margin guides draw in on all four sides, image eases inward | dashboard-before.png
6 | "One prompt. One scene." | filename text types out, arrow wipes to a rendered scene thumbnail | prompt-card.png
7 | "Give it an entrance" | image scales 96 to 100 percent with a short fade and settle ease | feature-shot.png
8 | "Ground it: drop shadow" | soft shadow grows under the image, before/after toggle | feature-shot.png
9 | "Ground it: grain pass" | grain texture fades over the image, matched to background grain | feature-shot.png
10 | "Render 1080p proof first" | export progress bar fills, then a check mark | render-ui.png
11 | "Six moves, every image" | six labels stack in fast, one per beat | none, type only
12 | "Part 1 done." | logo settle, next-part teaser line fades up | eleven-views-logo.png

ASSET AND RESOURCE CHECKLIST

IMAGES (drop in /assets/images, named exactly):
- bad-example-floating.png the deliberately wrong shot for the cold open, screenshot with no grounding
- dashboard-raw.png an oversized raw screenshot to demo the downscale, leave it big on purpose
- hero-photo.jpg a real-world photo to demo the PNG-to-JPG and quality-90 rule
- dashboard-before.png a UI screenshot to demo safe-margin placement
- prompt-card.png a clean frame of the prompt-to-scene prompt text for scene 6
- feature-shot.png one product or UI image carried through scenes 7, 8, 9 (entrance, shadow, grain)
- render-ui.png a screenshot of the export panel for the render beat
- eleven-views-logo.png brand logo for the close (already in the Eleven Views folder)

FONTS:
- Eleven Views display font for headlines (from the design system)
- A clean mono or grotesk for the filename and UI labels in scenes 3, 4, 6, 10
- Load both via Remotion loadFont so they are baked into the render, no system fallback

PROMPT DOCS (from this PDF):
- The exact prompt to turn an image filename into a working scene (prompt-to-scene section)
- The green-screen removal and edge cleanup steps (needed only if any asset has a transparent subject; not used in Part 1 but keep on hand for Part 2)
- The drop-shadow and grain grounding pass settings (shadow offset, blur, opacity; grain amount)

NARRATION:
- The Companion video narration script (Part 1) above, recorded as voiceover or run through TTS
- One audio file, narration.mp3 or .wav, dropped in /assets/audio

MUSIC (optional):
- A low, steady bed track at low volume, /assets/audio/bed.mp3, ducked under the voice

PROJECT:
- Remotion project initialized, composition set to 1920x1080 at 30fps
- A 4K composition variant at 3840x2160 for the master
- This PDF in the project root as the build reference

RENDER AND EXPORT RECIPE

ORDER OF OPERATIONS:
1. Prep every image first using the guide's asset prep rules. Do not enter a scene with an un-prepped image. Screenshots to 1920w PNG (or 3840w for 4K). Photos to JPG quality 90.
2. Build all scenes against the Scene map. Apply the grounding pass (drop shadow plus grain) to every real image as you place it, not at the end.
3. Snap scene durations to the narration line breaks so each block lands in its scene.
4. Render the 1080p proof. Watch it back in full. Check two things only: did the drop shadow read, and did the grain tie each image to its background. Fix the pass if not. Do not re-render the master until the proof is clean.
5. Render the 4K master.

1080p PROOF SETTINGS:
- Composition: 1920 x 1080, 30fps
- Codec: H.264
- CRF: 18 (visually lossless for a proof, small file)
- Pixel format: yuv420p (universal playback)
- Audio: AAC, 192 kbps, 48kHz
- Command shape: npx remotion render <CompId> out/proof-1080.mp4 --crf=18

4K MASTER SETTINGS:
- Composition: 3840 x 2160, 30fps

```
- Codec: H.264 for delivery, or ProRes if the buyer needs to re-edit
- CRF: 16 for H.264 (higher quality master), or --prores-profile=hq for ProRes
- Pixel format: yuv420p for H.264 delivery
- Audio: AAC, 256 kbps, 48kHz
- Image quality flag: --jpeg-quality=100 so the prepped images render at full
fidelity at 4K
- Command shape (H.264): npx remotion render <CompId4K> out/master-4k.mp4 --crf=16 -
-jpeg-quality=100
- Command shape (ProRes): npx remotion render <CompId4K> out/master-4k.mov --
codec=prores --prores-profile=hq

FINISH CHECK before delivery:
- Scrub the master at 4K and zoom one image to 200 percent. The grain should still
match the background grain. If the image looks cleaner than the scene around it at
4K, raise the grain amount and re-render the master.
- Confirm safe margins held at 4K. No image touching the frame edge.
- Confirm audio levels: voice peaks around -3 dB, music bed sits under -18 dB.
```

Want this built into your business?

You now have the recipes, the prompts, and the grounding pass that separate a sales-ready video from an obvious one. If you would rather hand the whole pipeline to a team that runs it daily, Eleven Views builds product and case-study videos, full funnels, and dashboards with the Desk, the set of about 25 Claude-powered agents behind these workflows, at roughly a twentieth of typical production cost. Book a build or ask about licensing the Desk for your own shop at elevenviews.io. If this saved you an afternoon, you can tip ATLAS on the way out. Book a call at atlas.elevenviews.io/book.