

REFERENCE TO LIVE WEBSITE

HOW WE BUILT ELOMARVAN.VERCEL.APP

We took two poster-style images from Pinterest, rebuilt them as a real website with animations, swapped in Marvan's own photos, and put it online. This guide walks through the exact steps so anyone can do the same, even without writing code yourself. You talk to Claude Code; it writes the code.



Section 1: the poster hero



Section 2: services around the photo

WHAT YOU NEED

Claude desktop app

Use the Code tab. This is who builds the site.

ChatGPT

For editing images: removing text and putting your face in.

A Mac

Claude uses the Mac's built-in "lift subject" feature to cut you out of photos.

Node.js 20+

Runs the site on your computer. Free from nodejs.org.

GitHub account

Stores your code. Free.

Vercel account

Puts the site online. Free plan is enough.

THE 11 STEPS

- 01 Pick your reference images
- 02 Make the photos yours in ChatGPT
- 03 Start the project in Claude Code
- 04 Swap in your edited photos
- 05 Add more sections
- 06 Change the words
- 07 Animations: the Framer Motion kit
- 08 Check every screen size
- 09 Save it to GitHub
- 10 Put it live on Vercel
- 11 Make changes after launch

1

PICK YOUR REFERENCE IMAGES

Each reference becomes one section of your site. Poster-style designs work best: one person, a big headline, a few small labels.

1. Find one image per section on Pinterest, Instagram or Behance. We used three: a desk poster (hero), a "My services" chair photo, and a tea-sipping photo (contact).
2. Save them at the largest size you can.
3. Write one line per image saying what that section is for. Example: *"Hero: who I am. Section 2: my services. Section 3: how to contact me."*

Tip: the reference is only a guide. Never publish someone else's photo. In step 2 you replace the person with yourself.

2 MAKE THE PHOTOS YOURS IN CHATGPT

This is what Marvan did: he uploaded each reference to ChatGPT, removed the text, and swapped the face for his own. Claude Code can't swap faces, so do this part in ChatGPT.

1. Open ChatGPT and upload **the reference image** plus **a clear photo of your face** (front-facing, good light).
2. Paste the prompt below.
3. If something looks off (hands, hair, glasses), ask for one fix at a time.
4. Download the result at full size.

CHATGPT PROMPT: SWAP THE FACE

Image 1 is a reference poster. Image 2 is a photo of me.
Recreate image 1 with me as the person: my face, hair and skin tone.
Keep everything else the same: pose, clothes, props, lighting, camera angle and framing.
Remove all text, logos, handles, buttons and barcodes. Fill in the background where they were.
Keep the same aspect ratio as image 1.

CHATGPT PROMPT: FUN EXTRA (WHAT WE DID FOR THE MONITOR)

Put a second version of me inside the computer screen, screaming and reaching out of it, like the woman in the original. Keep everything else unchanged.

CHATGPT PROMPT: TEXT REMOVAL ONLY

Remove all text, labels and buttons from this image and rebuild the background behind them. Don't change anything else.

Check: no text left in the image, your face looks natural, and the image is the same shape as the reference.

Why remove the text? The website draws its own text on top so it can animate it and you can edit it. Baked-in text would show twice. (If a little text is left over, Claude can erase it in step 4.)

3 START THE PROJECT IN CLAUDE CODE

1. Make an empty folder for the site, e.g. `marvan website`.
2. Open the Claude desktop app, go to the **Code** tab, and choose that folder.
3. Attach your reference images (the originals are fine here) and paste the prompt.

CLAUDE CODE PROMPT

Personal portfolio website.
Image 1 is the reference for the first section. Image 2 is the second section (contact).
Use placeholder text for now; I'll replace it later.
Build a Next.js version with matching animations. Use Framer Motion.
Put all the text and image paths in one content file so I can edit them easily.

Claude sets up Next.js, builds each section to match the reference, starts a preview in the app's browser pane, and checks it at desktop and phone sizes.

Check: the preview looks like your reference: same layout, same colors, same feel. Point out anything that doesn't match, in plain words.

4

SWAP IN YOUR EDITED PHOTOS

Attach the ChatGPT image from step 2 and say which section it's for.

CLAUDE CODE PROMPT

```
Use this image for the hero section.
```

What Claude does with it:

- **Cuts you out** of the background using the Mac's built-in subject lift, so you stand on the site's own animated background (like the checkered floor).
- **Erases leftover text** if any is still in the photo.
- **Lines the text up with the photo:** labels are placed as percentages of the photo, so they stay in the right spot on every screen size.

Want a different photo later? Just send a new one with "replace with this". We swapped the hero image three times.

5

ADD MORE SECTIONS

Send the new reference plus your edited photo, and describe what goes where. Lists, links and order changes all go in one message.

CLAUDE CODE PROMPT (THE ONE WE USED)

```
Use this as the second section, and move the current second section to third.  
Add services around him like the given reference.  
Services: Marketing, Video production, Podcast production, Personal branding,  
Performance marketing.  
Each one clickable, opening its own landing page.  
Add smooth scrolling from section 1 to section 2.
```

This gave us: the services section, five landing pages (`/services/marketing` and so on), smooth scrolling, and a "curtain" effect where section 2 slides up over the hero.

6

CHANGE THE WORDS

All the text lives in one file, `content.ts`. You can edit it yourself, or just tell Claude what to change.

CLAUDE CODE PROMPTS WE USED

```
@elomarvan handle. Last section is contact: Instagram and WhatsApp.  
"Let's talk" goes to WhatsApp, the handle goes to the elomarvan Instagram.  
WhatsApp: 6235168158
```

```
Heading change: not design, focus on performance marketing.  
Not designer: marketer and creator.
```

What `content.ts` looks like (simplified):

```
export const profile = {  
  handle: "@elomarvan",  
  role: "Marketer & Creator",  
  instagram: "https://instagram.com/elomarvan",  
};  
  
export const hero = {  
  kicker: "HOW I HELP BRANDS",  
  title: "MARVAN",  
  highlight: "GROW WITH PERFORMANCE MARKETING",  
  note: "I RUN ADS AND CONTENT FOR BRANDS THAT WANT SALES, NOT JUST LIKES.",  
};
```

WhatsApp links need the country code. 6235168158 becomes `wa.me/916235168158` for India (+91).

7

ANIMATIONS: THE FRAMER MOTION KIT

Every animation on the site is one of these recipes. You don't need to write them; ask for them by name, e.g. *"make the headline letters rise up one by one"*. Each recipe below

includes the code.

Letters rise from below

MARVAN headline

Each letter sits in a box that hides anything outside it, then slides up into view, one after another.

```
<span style={{ display: "inline-block", overflow: "hidden" }}>
  <motion.span
    style={{ display: "inline-block" }}
    initial={{ y: "105%" }}
    animate={{ y: "0%" }}
    transition={{ duration: 0.9, ease: [0.22, 1, 0.36, 1], delay: 0.4 + i
* 0.07 }}
  >
    {letter}
  </motion.span>
</span>
```

Wipe in a highlight bar

yellow bar

A clip that starts fully closed and opens left to right.

```
<motion.div
  initial={{ clipPath: "inset(0 100% 0 0)", rotate: -6 }}
  animate={{ clipPath: "inset(0 0% 0 0)", rotate: -1.5 }}
  transition={{ duration: 0.75, delay: 1.05 }}
/>
```

Draw a line

hand-drawn arrow

An SVG path that draws itself from 0% to 100% of its length.

```
<motion.path
  d="M16 4 C 0 46, 10 84, 64 98"
  initial={{ pathLength: 0 }}
  animate={{ pathLength: 1 }}
  transition={{ duration: 0.8, delay: 1.6 }}
/>
```

Spring pop-in

hero photo, buttons

A spring instead of a timed animation, so it overshoots a little and settles.

```
<motion.img
  initial={{ y: 140, scale: 0.9, opacity: 0 }}
  animate={{ y: 0, scale: 1, opacity: 1 }}
  transition={{ type: "spring", stiffness: 90, damping: 16, delay: 0.75 }}
/>
```

Follow the mouse

parallax depth

Track the mouse as a number from -1 to 1, smooth it with a spring, and move layers by different amounts. Bigger numbers feel closer.

```
const px = useMotionValue(0);
const sx = useSpring(px, { stiffness: 70, damping: 18 });
const photoX = useTransform(sx, (v) => v * -18);

// on the section:
onPointerMove={(e) => px.set((e.clientX / window.innerWidth) * 2 - 1)}

<motion.div style={{ x: photoX }} />
```

Move with scroll

curtain, zoom-out photo

Read how far the section has scrolled (0 to 1) and turn that into movement. For the curtain, the hero is pinned with CSS `position: sticky; top: 0` and the next section slides over it, while the hero shrinks and darkens.

```
const { scrollYProgress } = useScroll({ target: ref, offset: ["start start", "end start"] });
const scale = useTransform(scrollYProgress, [0, 1], [1, 0.92]);
const dim = useTransform(scrollYProgress, [0, 1], [0, 0.55]);

<motion.section ref={ref} style={{ scale }}> ... <motion.div style={{
  opacity: dim }} />
```

Reveal when scrolled into view

"My services", labels

Plays once when the element appears. **Gotcha we hit:** text hidden inside a clipping box never counts as "in view", so put the trigger on the parent and pass it down with variants.

```
<motion.h2 initial="hidden" whileInView="shown" viewport={{ once: true
}}>
  <span style={{ overflow: "hidden", display: "block" }}>
    <motion.span variants={{ hidden: { y: "115%" }, shown: { y: "0%" }
}}>My</motion.span>
  </span>
</motion.h2>
```

Gentle float

service labels

A loop that never stops. Give each item a slightly different duration so they don't move in sync.

```
<motion.div
  animate={{ y: [0, -5, 0] }}
  transition={{ duration: 3.2 + i * 0.45, repeat: Infinity, ease:
    "easeInOut" }}
/>
```

Magnetic button

"Let's talk"

The button leans toward the cursor and springs back when it leaves.

```
onPointerMove={(e) => {  
  const r = e.currentTarget.getBoundingClientRect();  
  mx.set((e.clientX - (r.left + r.width / 2)) * 0.35);  
  my.set((e.clientY - (r.top + r.height / 2)) * 0.35);  
}}  
onPointerLeave={() => { mx.set(0); my.set(0); }}
```

Smooth scrolling

Lenis

Wrap the whole site once. A button can then glide to any section.

```
import { ReactLenis, useLenis } from "lenis/react";  
  
<ReactLenis root options={{ lerp: 0.085, smoothWheel: true }}>{children}  
</ReactLenis>  
  
const lenis = useLenis();  
lenis?.scrollTo("#services", { duration: 1.6 });
```

Respect "reduce motion"

accessibility

One line makes every animation calmer for people who turn on Reduce Motion on their device.

```
<MotionConfig reducedMotion="user">{children}</MotionConfig>
```

Save the kit as a Claude skill

A skill is a note Claude reads automatically when it's relevant. Save this file once and every future project gets these animations without you explaining them again.

1. Create the folder `~/.claude/skills/poster-motion/` (the `~` means your home folder).

2. Save the text below inside it as `SKILL.md`.
3. Next time, just say *"use the poster-motion skill"*.

SKILL.MD

```
---
name: poster-motion
description: Framer Motion recipes for turning a poster-style reference image
into an animated Next.js section. Use when rebuilding a reference design as a
website with entrance, scroll and hover animations.
---

# Poster motion kit

Build with Next.js (App Router) + Framer Motion. Put all copy and image paths
in content.ts.

## Layout rules
- Treat each reference image as one full-screen section.
- Cut people out of photos (transparent PNG) so they sit on the site's own
background.
- Place labels and buttons as % of the photo's frame (use container query
units for text size) so they line up at every screen size.
- Check 1440x900, 1280x720, 1000x570 and 390x844 phone before calling it
done.

## Animation recipes
- Headline: split into letters, each in an overflow-hidden span, slide y 105%
-> 0, stagger 0.07s, ease [0.22,1,0.36,1].
- Highlight bar: clipPath inset(0 100% 0 0) -> inset(0 0% 0 0), slight
rotate.
- Hand-drawn arrow: SVG path pathLength 0 -> 1.
- Photos and buttons: spring entrance (stiffness ~90, damping ~16).
- Mouse parallax: useMotionValue -> useSpring -> useTransform, different
depth per layer.
- Scroll: useScroll + useTransform; pin the hero with position: sticky and
slide the next section over it (hero scales to 0.92 and dims).
- Reveal on scroll: whileInView with viewport once. Put the trigger on the
parent when children start clipped, and use variants.
- Idle: y [0,-5,0] loop with a different duration per item.
- CTA: magnetic hover (pointer offset x0.35 through a spring) + pulsing ring.
- Smooth scroll: Lenis (ReactLenis root) and lenis.scrollTo for "scroll to
section" buttons.
- Wrap the app in MotionConfig reducedMotion="user".
```

Gotchas

- Gradient text (background-clip: text) cuts off descenders like "y": add padding-bottom with a matching negative margin.
- Masks clip italic overhangs: pad the mask on the right and bottom.
- Labels near the edge run off phone screens: give phones their own positions.

8

CHECK EVERY SCREEN SIZE

Most problems only show up at certain sizes. Ask Claude to check them all at once:

CLAUDE CODE PROMPT

Take screenshots at 1440x900, 1280x720, 1000x570 and phone size (390x844). Fix anything that overlaps, gets cut off or runs off the screen.

If you spot something yourself, send a screenshot and say what's wrong. That's how we fixed the hero spacing: *"fix the desktop spacing of text under the hero, add space in between"*.

Problems we hit, and the fix

WHAT YOU SEE	FIX
Small note touching the yellow bar	Make the photo slightly smaller and move the note down.
"My" showing as "Mv"	Gradient text cut off the tail of the "y"; give it more room at the bottom.
Heading never appears	The reveal was waiting on hidden text; trigger it from the parent instead.
Labels running off a phone screen	Give phones their own label positions.

WHAT YOU SEE	FIX
Photo has hard straight edges	The original photo cropped the shoulders; enlarge it so the edges sit below the screen and fade the sides.
Photo tiny on short laptop screens	The section had a minimum height that fought the layout; lower it.

9 SAVE IT TO GITHUB

First make sure you're logged in: in a terminal, run `gh auth login` once (GitHub's own tool). Then:

CLAUDE CODE PROMPT

```
Create a GitHub repo called elomarvan and push this project as private.
```

Check: the repo shows as **Private** on github.com, and secrets like `.env` files are not in it.

10 PUT IT LIVE ON VERCEL

CLAUDE CODE PROMPT

```
Host this on Vercel. The URL should be elomarvan.vercel.app.
```

1. Claude opens a terminal tab running `vercel login`. The first run downloads the tool, which can take a few minutes.
2. A link appears. Open it and sign in to Vercel yourself (Claude never handles your password).
3. Tell Claude you've signed in. It creates the project named **elomarvan**, deploys it, and connects it to your GitHub repo.

The project name becomes the address: project `eLomarvan` → `eLomarvan.vercel.app`.
If someone else already has that name, Vercel picks a different one, so check the final URL.

Check: open the URL on your phone (not just your computer) and make sure it loads without a login screen.

11 MAKE CHANGES AFTER LAUNCH

Because Vercel is connected to GitHub, every push updates the live site in about a minute. The loop is:

1. Ask for the change in plain words (with a screenshot if it's visual).
2. Look at it in the preview.
3. Say *"commit and push"*. The live site updates on its own.

THE REAL CONVERSATION

Every message Marvan's build used, in order. Your own build can follow the same path.

01 "Personal portfolio website. First section reference. Placeholder text now, will replace later. The tea sipping one as 2nd section to contact. Build a Next.js version with matching animations. Use Framer Motion."

Two reference images attached. Result: hero + contact sections with animations.

02 "Use those exact images for now, not animated ones."

Swapped the drawn placeholders for the reference photos.

03 "Use this image." / "Replace with this."

Marvan's ChatGPT face-swapped versions of the hero, cut out and placed.

- 04** "Use this as the second section and move the current second section to third. Add services around him... each clickable... add smooth scrolling."
Services section, five landing pages, Lenis smooth scroll, curtain transition.
- 05** "@elomarvan handle. Insta and WhatsApp contact."
Instagram and WhatsApp links everywhere.
- 06** "Heading change. Not design, focus performance marketing. Not designer, marketer and creator."
New headline and role.
- 07** "Change this to match."
New note line under the headline.
- 08** "Create GitHub repo elomarvan and push as private."
Private repo on GitHub.
- 09** "Host in Vercel. elomarvan.vercel should be the URL."
Live at elomarvan.vercel.app with auto-deploys.
- 10** "Fix the desktop spacing of text under the hero, add space in between."
Spacing fixed, pushed, live a minute later.

WHERE THINGS LIVE

FILE	WHAT IT IS
<code>content.ts</code>	All the words, links and image paths. Edit this most.
<code>public/</code>	The photos: <code>hero.png</code> , <code>services.png</code> , <code>contact.jpg</code> , <code>avatar.jpg</code> .
<code>components/Hero.tsx</code>	Section 1 and its animations.

FILE	WHAT IT IS
<code>components/Services.tsx</code>	Section 2 and the service labels.
<code>components/Contact.tsx</code>	Section 3: photo, "Let's talk", Instagram and WhatsApp.
<code>components/ServiceLanding.tsx</code>	The layout shared by all five service pages.
<code>app/globals.css</code>	Colors, fonts, sizes and positions.

Built with Claude Code, ChatGPT (image edits), Next.js, Framer Motion and Lenis. Live at elomarvan.vercel.app.