WEEK 3 FUN COOL

Add an AI Clone of Yourself

Learning Objectives

By the end of this session, students should be able to:

- **Obtain an API key from Google AI Studio and wire it into both local `.env.local` and Vercel's production env** — with the AI doing the plumbing and the student doing only the human-only OAuth moment.
- **Write a persona system prompt** that makes Gemini 2.5 Flash answer as **you**, in your voice, with your background — the student's first serious piece of prompt engineering.
- **Describe a streaming chat UI in English** and have Cursor implement it using the Vercel AI SDK, with no hand-written React.

Core Topics

- System prompts vs user prompts: why the **invisible** instructions matter more than the visible ones.
- Streaming responses — why "tokens arrive one by one" is the default mode for modern chat UIs.
- Secret management: where API keys go, where they **don't** go, and how the AI helps you not commit them.
- The "persona file" pattern — keeping your AI-self definition in one editable Markdown file.

Tools / Stack

Tool	Role this week
Cursor	Where you talk; AI writes the chat component.
Gemini 2.5 Flash	The model answering as your AI-clone. Fast, cheap, good enough.
Google AI Studio	Where you fetch the free API key. One manual moment.
Vercel AI SDK	Abstracts streaming, state, and provider differences.
`.env.local` + Vercel env	Two places your API key lives — both set by AI via CLI.

Your personal site — cumulative features through this week:

Wk 1 · Dev toolkit installed — Cursor, Claude Code, Gemini CLI, MCPs, Typst skill

Wk 2 · Personal website deployed to Vercel

Wk 3 · **AI chat widget (Gemini 2.5 Flash) emulating you** ← NEW

Session Plan

Time	Activity
0 – 15 min	**Recap & Check-in.** Quick round: everyone reads their week-2 site URL out loud. Note whose site feels the most "them" — we're going to make that feeling louder today.
15 – 40 min	**Concept Teaching.** What a system prompt is and why it matters. Why streaming feels alive and batch replies feel dead. How env vars flow through Next.js server actions.
40 – 75 min	**Live Demo.** Instructor adds the chat widget to her own site in one sitting — from "no chat" to "chat with my AI-clone streaming live" in twelve minutes.
75 – 105 min	**Hands-On Lab.** Students add the widget to their week-2 site. The fun part is writing their persona prompt — we'll share favourites at the end.
105 – 120 min	**Q&A + Wrap.** Each student demos their AI-clone answering one question from a classmate.

Hands-On Lab

Task. By the end of class your portfolio site will have a floating chat widget in the bottom-right corner. Anyone can open it and ask "tell me about yourself" — the answer arrives streaming, in your voice, grounded in your real background.

MANUAL **Step 1 · Manual (human only):**

> Open aistudio.google.com/apikey, sign in with your Google account, click **Create API key**, and copy it. Keep the tab open — you'll paste the key into Cursor in the next step.
>
> *Why manual: Google's API-key creation requires a human to accept their usage terms. No CLI or MCP can press "Agree" for you.*

PROMPT **Step 2 · Say to Cursor:**

> *Here is my Gemini API key: **[paste key here]**. Please:*
>
> 1. *Add it as `GOOGLE_GENERATIVE_AI_API_KEY` to my `.env.local` (create the file if it doesn't exist). Make sure `.env.local` is in `.gitignore` — I never want this key on GitHub.*
> 2. *Add the same key to my Vercel production environment using the Vercel CLI.*
> 3. *Install the Vercel AI SDK and the Google provider (`ai` , `@ai-sdk/google`).*
> 4. *Confirm both environments have the key set before continuing.*

VERIFY **Step 3 · Verify:**

- `.env.local` contains the key; `.gitignore` lists `.env.local` .
- `vercel env ls` (Cursor will run this for you) shows `GOOGLE_GENERATIVE_AI_API_KEY` in production.
- `package.json` has `ai` and `@ai-sdk/google` under dependencies.

PROMPT **Step 4 · Say to Cursor:**

> *Create a new file `content/persona.md` that will define my AI-clone's personality. Write it as a **system prompt** — instructions to the model about who I am, how I speak, and what I know. Use this information about me:*
>
> *• Who I am: **[name, year, degree, school]** • What I'm into: **[3–5 topics you actually care about — not what sounds cool]** • What I'm building: **[your portfolio + your week-1 "hello TECHNEST" + whatever side projects are real for you]** • Voice: **[pick two adjectives — e.g., "warm but precise," "playful and nerdy," "direct and low-hype"]** • What I won't answer as: **["don't pretend to know things I don't," "decline to roleplay as someone else"]***
>
> *Write the file as one clear, long-ish paragraph in second-person **instructions** ("You are Chan Meng…"), not first-person bio. When you're done, paste the file contents back into chat so I can read it.*

VERIFY **Step 5 · Verify:**

- `content/persona.md` exists and reads like instructions to a model, not a bio.
- The adjectives you chose are reflected in the phrasing.
- It includes explicit "don't do this" clauses.

PROMPT **Step 6 · Say to Cursor:**

> *Now add a floating chat widget to the site:*
>
> - *A round launcher button in the bottom-right corner with my accent colour.*
> - *Click opens a panel 360 px wide × 480 px tall with a clean chat UI: messages bubble list, input at the bottom, close button at top.*
> - *Wire it to a Next.js route handler at `/api/chat` that uses the Vercel AI SDK to stream responses from Gemini 2.5 Flash.*
> - *Load the system prompt from `content/persona.md` on each request.*
> - *When the model is generating, show a small animated dots indicator.*
> - *Persist the current session to `localStorage` so a refresh doesn't wipe it.*
>
> *Make sure it works on mobile (the panel becomes full-screen on narrow viewports). Commit in small logical commits — one for the API route, one for the UI, one for the localStorage wiring.*

VERIFY **Step 7 · Verify:**

- Open your site at `http://localhost:3000` . A launcher button appears bottom-right.
- Click it. Type **"What are you building this term?"**. The reply streams in token-by-token in your voice.
- Refresh. The conversation persists.
- Open the same page on your phone — the panel becomes full-screen; input is tappable.

PROMPT **Step 8 · Say to Cursor:**

> *Ship this to production. Commit all remaining changes, push to GitHub, and confirm the Vercel auto-deploy succeeds. When the production URL is live, open it and test the chat from the live site — same question as before to make sure the env vars carried over.*

VERIFY **Step 9 · Verify:**

- Your production site now has the chat widget.
- The live chat streams responses correctly from the production URL.
- No `GOOGLE_GENERATIVE_AI_API_KEY` references in any file that got committed.

RECOVER **Step 10 · If stuck, say to AI:**

*The chat says **"500 internal server error"** on the live site but works locally. Please read the Vercel function logs, find the root cause (most likely a missing env var in production), fix it through the Vercel CLI, trigger a redeploy, and confirm the chat works.*

TIP · Iterate the persona, not the code

When your AI-clone says something off-brand, **don't** edit the component. Open `content/persona.md` and describe the new constraint in plain English: **"Don't use the word 'passionate'"** or **"Always answer in under 80 words unless I ask for detail."** Save and reload. The persona file is where your prompt engineering lives.

CAREER · Screen-record your AI-clone

Record a 20-second clip of you opening the chat and asking **"What's something you're working on right now?"** This clip is gold for LinkedIn posts and hiring conversations — it's concrete proof you can build a streaming LLM feature end to end.

Weekly Assignment

Build / Implement.

- Floating chat widget on your live site powered by Gemini 2.5 Flash.
- A `content/persona.md` file committed to your repo (yes, your persona lives in source control — future-you will thank present-you).

Requirements.

- The widget streams responses (not a single dump at the end).
- Session persists across page refresh.
- API key is in Vercel production env, not in any committed file.
- 20-second screen recording of the live chat answering one question.

Submission. Live URL + short recording in the course Slack channel before Week 4.

Resources

Docs

• Vercel AI SDK — streaming with Google provider
• Google AI Studio — API keys + rate limits

Videos

• Instructor demo: "Wiring persona + streaming in 12 minutes"

Repos

- `vercel/ai` — SDK source
- `her-waka/tutorial/vibe-coding/build-with-claude.mdx` — deeper dive

• Next.js route handlers — streaming responses

Real-World Application

Every AI product you'll build in your career has a system prompt somewhere. The skill of writing clear instructions to a model — constraining voice, scope, and failure modes — is the single most durable AI-engineering skill. The "AI-clone" framing makes it concrete: you learn prompt engineering by tuning a thing that sounds like **you**, not by reading a textbook.

> **CAREER**
>
> Recruiters who visit your portfolio will now have a chat widget that answers "Why should I hire this person?" in your voice, at 2 AM their time. That's not a gimmick — that's an asymmetric advantage.

Challenges & Tips

- **"The AI says things I didn't tell it to."** Tighten the persona file. Add a line: **"If you don't know the answer, say 'I haven't talked about that yet — ask me next week.'"** Models respect explicit fallbacks.
- **"Streaming shows nothing, then the whole reply drops at once."** You're probably returning the response instead of piping the stream. Say to Cursor: **"The response isn't streaming — it arrives as a single chunk. Use `toDataStreamResponse` from the Vercel AI SDK."**
- **"I get `API_KEY_INVALID` on production."** The key wasn't added to Vercel env. Say: **"Show me which env vars are set on my Vercel project, and make sure `GOOGLE_GENERATIVE_AI_API_KEY` is set for the Production environment specifically."**
- **"The widget looks ugly on my phone."** Describe the broken layout in specific terms: **"On my iPhone the input gets hidden behind the keyboard."** Cursor knows the viewport-fit pattern to fix it.
- **"The model's answers are too long / too formal."** Add constraints to the persona: **"Default reply length: 2 short paragraphs max. Avoid corporate phrasing."**

> **STUCK? RESCUE**
>
> If the chat works locally but breaks in production after deploy, say to Cursor: **"Compare my `.env.local` with my Vercel production env vars and list every difference."** This catches 90% of deploy-time AI bugs.

Instructor Notes (Internal)

- **Pacing hazard: the API-key step.** Some students get distracted by Google AI Studio's other toys. Tell them upfront: "Get the key, don't explore, come back."
- **Watch for committed keys.** As students race to deploy, some will paste the key directly into code instead of `.env.local`. Walk the room during step 3; if you see a `const API_KEY = "AIza..."` in someone's editor, stop them immediately.

- **Make the persona exercise collaborative.** Midway through step 4, pair students to read each other's draft personas out loud. It exposes weak voice instantly.
- **Celebrate the first streaming chat.** The first student whose live chat streams should get a round of applause — this is the first **"real AI product"** moment of the term. Make it an event.
- **Rate limits.** Gemini free tier is generous but not infinite. If someone exhausts it during testing, have them wait 10 minutes or use a classmate's laptop.

Student Feedback

Challenges Observed

Extra Information

MEET YOUR INSTRUCTOR

Chan Meng

Full-Stack Engineer · AI Agent Builder · Minimalist

> *"Subtraction for life, addition for thought."*

AI AGENT BUILDER

AGENTIC SYSTEMS · LLM INTEGRATION

FEMTECH INNOVATOR

MINIMALIST CODE · UN CSW 69 SPEAKER

Chan builds **production-grade AI agents** at the intersection of **agentic systems, LLM integration, and women's health**. Between China and New Zealand, she ships systems that go to work for real users at real companies — and she believes code, like life, becomes most powerful when you subtract what isn't essential. A **Master of Applied Computing with Distinction** from Lincoln University, NZ (Dean's List), she was featured at **UN Women CSW69** for her work on AI and gender equality.

Currently · Founding Engineer at Gavigo · Senior Full-Stack Engineer at She Sharp (NZ's leading women-in-STEM nonprofit, 2,200+ members) · CTO of FemTech Weekend · Open Source Contributor to CopilotKit (24.6k★). For the full project portfolio and code, visit github.com/ChanMeng666.

Stay in the loop

NEWSLETTER · RECOMMENDED

"What's Shipping in AI" — a weekly curated digest, every Monday. chanmeng.org/#newsletter

LINKEDIN
chanmeng666

GITHUB
ChanMeng666

PORTFOLIO
chanmeng.org

EMAIL
chanmeng.dev@gmail.com

Building tools that matter. Let's connect.