TECHNEST Track Curriculum

Track: Artificial Intelligence

Track Goal

By the end of this 12-week track, students — who began the course having only used ChatGPT on the web — will be able to:

- **Drive AI assistants (Cursor, Claude Code, Gemini CLI) to build and ship full-stack AI applications** without hand-writing code, using natural-language prompts as the primary programming interface.
- **Stand up production infrastructure entirely through AI-invoked CLIs and MCP servers** — Vercel for hosting, Neon for Postgres + Auth, Vercel Blob for file storage, GitHub for source control, Slack for notifications, Typst for PDF generation.
- **Own a live, polished, AI-powered personal website** that doubles as their portfolio, their AI-clone chatbot, their guestbook, their blog, their contact funnel, and the source-of-truth for their auto-generated CV and cover letter.

Capstone Project Goal

In weeks 9–12 students form small teams (2–4) and evolve their personal-website codebase into a **deployed multi-user AI SaaS MVP** on one of three tracks:

- **Campus Life** — event finder with AI recommendations, roommate matcher, language-exchange app, dorm chore-and-bill splitter.
- **Personal Growth** — habit coach with weekly AI reviews, reading companion, journaling assistant, workout planner.
- **Creative Tools** — AI short-story studio, music-prompt playground, comic-panel generator, resume-video script maker.

Every capstone ships to Vercel, uses an LLM API, stores state in Neon, and is demonstrated in a live 5-minute team presentation.

The Delegation Rules (course-wide doctrine)

This track runs on one principle:

> **TIP · Natural language is the source code.**
>
> Students describe what they want. AI writes, installs, configures, deploys, and commits. Students verify. Students only touch files or tools with their own hands when **no CLI, API, MCP server, or Skill can do the job** — which, in practice, means signing into third-party consent screens and copy-pasting secrets that only a human can see.

Operation	AI drives via	Student role
Provision DB / Auth	Neon MCP, Neon CLI	Verify
Deploy + env vars	Vercel CLI, Vercel MCP	Verify
Git commit + push	Cursor / Claude Code	Verify
Install dependencies	Cursor runs `pnpm add`	Verify
Write / edit source files	AI edits the files	Verify + critique
Schema migrations	AI + Drizzle CLI	Verify
PDF generation	Typst skill in Claude Code	Verify
Slack webhook calls	AI fetches / posts via CLI	Verify
OAuth consent screens	— (human required)	**Do it**
API keys from provider sites	— (human required)	**Do it**
DNS / domain records	— (human required)	**Do it**
`.env.local` edits when no CLI exists	— (human required)	**Do it, lecture shows the exact line**

Every weekly Hands-On Lab applies these rules. Students will recognise the pattern by Week 2 and own it by Week 4.

8-Week Teaching Plan

The entire teaching phase evolves **one progressively-built personal website**. Each week grafts a meaningful full-stack feature onto the same codebase, so by Week 8 the student owns a single coherent production-grade product.

Week 1 — AI Developer Toolkit Setup

Topic. Install Cursor, Claude Code, Gemini CLI. Install Skills and MCP servers (Vercel MCP, Neon MCP, Typst skill). Set up GitHub and connect to Cursor. Try Wispr Flow for voice-to-prompt input. Practise the delegation pattern on a throwaway sandbox repo.

Skills. Installing dev tools from their official sources · signing into GitHub from an IDE · installing an MCP server and a Skill · dictating prompts via voice · watching AI call a CLI on your behalf.

JOB

Week 2 — Deploy Your Personal Website

Topic. Fork a Vercel personal-portfolio template (Magic Portfolio). Customise content through conversation with Cursor. Ship to production via AI-invoked Vercel CLI. Connect a custom domain if the student has one.

Skills. Forking a template through AI · describing site content in natural language · verifying a Vercel deployment · reading Vercel build logs when something fails.

JOB FUN

Week 3 — Add an AI Clone of Yourself

Topic. Integrate Gemini 2.5 Flash as a floating chat widget. Write a persona system prompt that makes the widget answer in the student's own voice (background, interests, project history).

Skills. Obtaining and storing an API key · system-prompt design as a student's "first serious prompt-engineering task" · streaming responses in a Next.js app · prompt iteration through AI collaboration.

FUN COOL

Week 4 — Go Full-Stack: Neon + Auth

Topic. Provision Neon Postgres and Neon Auth via MCP/CLI. Add a guestbook where signed-in visitors can leave messages. Drizzle ORM manages the schema.

Skills. Letting AI provision a real database · understanding (but not hand-writing) Drizzle schema · Google OAuth consent as the archetypal manual step · cross-environment env-var plumbing (local + Vercel).

JOB LIFE

Week 5 — Image Uploads with Vercel Blob

Topic. Extend the guestbook so visitors can attach images / memes alongside their messages. Use Vercel Blob for storage.

Skills. File-upload UX · secure upload tokens · verifying a binary upload round-trip end-to-end · thinking about blob-storage cost and limits.

FUN LIFE

Week 6 — Blog System

Topic. Graft a `/blog` section onto the site using a Vercel blog template as reference. MDX posts, syntax-highlighted code blocks, RSS feed.

Skills. MDX authoring · routing in Next.js App Router · consuming a template's patterns through AI rather than by reading its code · publishing a first post.

JOB LIFE

Week 7 — Real-Time Slack Notifications

Topic. Build a contact form. On submission, persist to Neon **and** fire a Slack webhook that notifies the student in real time.

Skills. Slack App creation and webhook setup · writing a Next.js server action through AI · observability-as-notification · thinking about PII and spam protection.

JOB COOL

Week 8 — Typst PDF Automation for Job Applications

Topic. Use the Typst CLI and the Typst skill inside Claude Code to generate a pixel-perfect CV and cover letter. Drive the template data from the same Neon `profile` table that powers the portfolio — one data source, many outputs.

Skills. Data-driven document generation · driving the Typst skill through a prompt · one-click CV / cover-letter download from the site · shipping a job-search-ready portfolio.

JOB LIFE

Required Tools / Stack

Layer	Tool	Added
IDEs	Cursor (primary), Claude Code, Gemini CLI	W1
Skills / MCP	Vercel MCP, Neon MCP, Typst skill	W1
Voice input (optional)	Wispr Flow	W1
Version control	GitHub, driven through Cursor	W1–2
Web framework	Next.js 14 + Tailwind CSS	W2
Hosting / deploy	Vercel (CLI and MCP)	W2
LLM in-product	Gemini 2.5 Flash	W3
Database + Auth	Neon Postgres + Neon Auth + Drizzle ORM	W4
File storage	Vercel Blob	W5
Content	MDX blog	W6
Notifications	Slack Incoming Webhook / Slack App	W7
Typesetting	Typst CLI + Typst skill	W8

Assessment Strategy

- **Weekly Assignments — 50%.** Each week the student submits (a) an updated live site URL and (b) a screenshot of the Cursor / Claude Code conversation that drove the change. The screenshot requirement reinforces the prompt-first doctrine: the prompt **is** the student's work.
- **Participation — 10%.** Attendance, voice in Q&A, helping unblock classmates.
- **Capstone — 40%.** Functionality, deployment health, team collaboration, live demo.

Learning Milestones

By Week 4, students will be able to:

- Open a Next.js project in Cursor, describe a database-backed feature in plain English, and watch Cursor provision Neon, wire Drizzle, deploy to Vercel, and commit to GitHub — end-to-end — from a single conversation.
- Recognise when a step requires human hands (OAuth consent, DNS, provider-side credentials) and know exactly what to do at those moments.

By Week 8, students will be able to:

- Ship a feature to production in a two-hour class and leave with a live URL to share with friends, family, and recruiters.
- Point a recruiter at a single personal site that proves full-stack AI competence: auth, database, file storage, blog, real-time alerts, and AI-generated CV / cover-letter downloads, all served from one Next.js app.

Capstone Evaluation Criteria

- **Functionality** — every promised feature works end-to-end in production.
- **Code quality** — reasonable component structure, no hard-coded secrets; AI wrote the code but the team understood and critiqued it.
- **UI / UX** — clean, accessible, mobile-friendly.
- **Deployment** — live on the internet, database migrations applied, error-free logs during the demo.
- **Presentation** — problem statement, solution, tech stack, live demo, challenges faced.

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.