



The AI Cheat Sheet

75 Tech Terms Explained in Plain English

No jargon. No BS. Just plain English.

by Steve @ SpeakNerd.ai

What's Inside

75 terms organized into 6 categories. Each term includes a plain English explanation and why it matters to you.

AI Basics (27 terms)

AI Agent | Alignment | Benchmark | Bias | Chatbot | Computer Vision
Diffusion Model | Embedding | Fine-tuning | Generative AI | GPT | Hallucination
Inference | LLM | Machine Learning | Model | Multimodal | NLP | Parameter
Prompt | RAG | Reinforcement Learning | Supervised Learning | Text-to-Image
Token | Training Data | Transformer

Vibe Coding (5 terms)

Claude Code | Copilot | Cursor | Vibe Coding | Windsurf

Building Blocks (28 terms)

Algorithm | API | Automation | Backend | Bandwidth | Cache | CDN | The Cloud
Cookie | Database | Debugging | Deploy | DNS | Docker | Encryption
Framework | Frontend | JSON | Latency | Markdown | No-Code / Low-Code
Open Source | Repository | Serverless | SSL/TLS | Vector Database
Version Control | Webhook

Buzzwords (3 terms)

AGI | Deep Learning | Neural Network

Money & Business (5 terms)

Churn | MRR | MVP | Product-Market Fit | SaaS

Tools & Platforms (7 terms)

ChatGPT | Claude | Cloudflare | GitHub | IDE | Supabase | Vercel

AI Basics

AI Agent

AI that actually does things

TL;DR: An AI that doesn't just answer questions -- it takes actions. The difference between a GPS that gives directions and a self-driving car that drives you there.

A chatbot answers your questions. An AI agent does your errands.

Why it matters:

Because AI agents are about to change how work gets done. Instead of spending 30 minutes doing a multi-step task yourself, you describe what you want and the agent handles it.

Alignment

Making AI do what we actually want

TL;DR: The challenge of making sure AI does what humans actually MEANT, not just what they technically said. Like the monkey's paw -- you got your wish, just not the way you wanted.

Ever given someone instructions and they did EXACTLY what you asked -- but it was completely wrong? "Clean up the living room" and they shoved everything into a closet? Technically they cleaned up. But that's not what you meant.

Why it matters:

Because alignment determines whether AI is a tool that helps humanity or one that accidentally harms it. Every time a social media algorithm radicalizes someone, or a chatbot gives dangerous medical advice, or an AI system discriminates -- that's...

Benchmark

Standardized tests for AI

TL;DR: The SATs but for robots. Standardized tests that let companies prove their AI is smarter than the competition -- or at least better at taking tests.

How do you know if one AI is smarter than another? You can't just ask it -- it'll say it's the best. You need a standardized test. That's what benchmarks are.

Why it matters:

Because every AI company will tell you their model is the best, and benchmarks are how they try to prove it. Being able to read benchmark comparisons (even casually) helps you cut through marketing hype.

Bias

When AI inherits our prejudices

TL;DR: AI learns from human data, and humans are biased. So AI picks up those biases too. If you only show it photos of male CEOs, it assumes all CEOs are male.

Here's an uncomfortable truth: AI is basically a mirror. It reflects whatever it was trained on -- including all the biases, stereotypes, and unfairness baked into that data.

Why it matters:

Because biased AI makes real decisions about real people. Loan approvals, job screenings, medical diagnoses, criminal sentencing -- AI is being used in all of these.

Chatbot

A program you talk to

TL;DR: Software that talks back. The old ones were dumb (press 1 for billing). The new ones (ChatGPT, Claude) are scarily smart. Same category, completely different league.

Remember those terrible automated phone systems? "Press 1 for billing. Press 2 for support. Press 0 to scream into the void." Those were the first chatbots -- programs designed to interact with humans through conversation. They followed rigid scripts and broke the moment you said anything unexpected.

Why it matters:

Because chatbots are rapidly becoming the interface for everything. Customer support, search engines, coding assistants, writing tools, tutoring -- conversations are replacing clicks.

Computer Vision

Teaching computers to see

TL;DR: Teaching computers to understand images and video. Like showing a toddler 10 million cat photos until they can spot a cat anywhere -- except the toddler is a GPU.

You glance at a photo and instantly know there's a dog sitting on a couch in a living room. You didn't think about it. Your brain just... did it. Computers had to be painstakingly taught to do what your eyes and brain do automatically.

Why it matters:

Because computer vision is increasingly watching you -- literally. It's in your phone's camera, your doorbell camera, security systems, retail stores, and your car.

Diffusion Model

The tech behind AI art

TL;DR: The technology powering AI image generators like Midjourney and DALL-E. It learns by adding noise to images, then learns to remove it. Like learning to clean by first messing up.

Imagine you take a beautiful photograph and slowly add static to it -- like old TV snow -- until it's nothing but random noise. Now imagine learning to reverse that process. Starting from pure static and gradually removing noise until a beautiful image emerges. That's basically how diffusion models work.

Why it matters:

Because diffusion models democratized visual creation. You used to need years of artistic training or thousands of dollars for a designer to create original images.

Embedding

Turning meaning into math

TL;DR: Converting words, images, or anything into a list of numbers that captures their meaning. It's how AI knows 'happy' and 'joyful' are close but 'happy' and 'refrigerator' aren't.

How do you explain to a computer that "happy" and "joyful" mean basically the same thing, but "happy" and "toaster" don't? Computers don't understand words -- they understand numbers. So you need a way to turn meaning into math.

Why it matters:

Because embeddings are the hidden technology that makes AI feel smart. They're why search engines understand your intent, why recommendation systems work, and why AI can find relevant information even when you don't use the exact right words.

Fine-tuning

Teaching AI to specialize

TL;DR: Taking a general AI and training it to be an expert at one specific thing. Like teaching a general practitioner to be a heart surgeon.

Imagine a doctor who just graduated medical school. They know a little about everything -- bones, hearts, skin, brains. They're smart, but they're not an expert in anything specific. Now imagine that doctor spends two years doing nothing but heart surgery. After that, they're a heart specialist.

Why it matters:

Because this is how businesses are making AI actually useful for specific tasks. A generic AI gives generic answers.

Generative AI

AI that creates new stuff

TL;DR: AI that doesn't just analyze -- it CREATES. Text, images, code, music, video. The 'gen' in GenAI. This is the revolution everyone's talking about.

For decades, AI was mostly about analyzing things. Sorting emails, detecting fraud, recommending movies. Useful, but not exactly mind-blowing. Then generative AI showed up and started CREATING things. Brand new text. Original images. Working code. Music. Video. Things that never existed before.

Why it matters:

Because generative AI is reshaping every creative and knowledge-work industry simultaneously. Writing, design, coding, music, marketing, education -- all being transformed.

GPT

Generative Pre-trained Transformer

TL;DR: The model family behind ChatGPT. It's the engine -- ChatGPT is the car you drive.

Think of GPT like an engine and ChatGPT like the car. GPT is the technology under the hood -- it's the thing that actually generates text. ChatGPT is the nice interface OpenAI wrapped around it so regular people could use it.

Why it matters:

Because GPT is the technology that kicked off this entire AI revolution. When you hear people talking about "generative AI," GPT is what started it.

Hallucination

When AI makes stuff up

TL;DR: When AI confidently tells you something that's completely wrong. Like that friend who never admits they don't know.

You know that friend who always has an answer -- even when they clearly don't know what they're talking about? The one who'll confidently tell you that "actually, the Great Wall of China is visible from space" even though that's not true?

Why it matters:

Because if you're going to use AI, you need to know it will sometimes be wrong. Not occasionally -- regularly.

Inference

When AI uses what it learned

TL;DR: Training is studying for the exam. Inference is taking it. It's the moment AI actually uses its knowledge to give you an answer.

Think about learning to drive. There's the learning phase -- months of practice, making mistakes, building skills. Then there's the actual driving phase -- you get in the car and go somewhere. Those are two very different things.

Why it matters:

Because when people talk about the cost of AI, they're usually talking about inference costs. Every question you ask costs someone a fraction of a cent in computing power.

LLM

Large Language Model

TL;DR: The brain behind ChatGPT. It read the entire internet and now it guesses what word comes next. Really, really well.

Imagine you had a friend who read every book, article, and website ever written. Now imagine you could ask that friend anything, and they'd give you a pretty solid answer -- not because they actually understand what they read, but because they've seen so many patterns that they can predict what a good answer looks like.

Why it matters:

Because LLMs are the engine behind almost every AI tool you're hearing about. ChatGPT, Claude, Gemini -- they're all LLMs under the hood.

Machine Learning

Computers that learn from experience

TL;DR: Instead of programming every rule by hand, you show the computer thousands of examples and let it figure out the pattern. Like learning to cook by tasting, not reading recipes.

Imagine you're teaching a kid to recognize dogs. You don't sit them down and say, "A dog has four legs, fur, a tail, weighs between 5 and 150 pounds..." You just point at dogs. "Dog. Dog. Not a dog. Dog." After enough examples, the kid gets it. They can spot a dog they've never seen before.

Why it matters:

Because machine learning is already running your life. It decides what shows up in your social media feed, what Netflix recommends, whether your email goes to spam, whether your loan gets approved, and increasingly, whether you get that job...

Model

The trained brain of an AI

TL;DR: The finished product after training -- the 'brain' that does the thinking. GPT-4, Claude, Llama -- those are all models. Billions of dollars of training, packaged up for you to use.

Think of a model like a trained employee. You invested time and money training them (that's the training phase), and now they're ready to work. The "model" is the finished result -- all the knowledge and capability, packaged up and ready to answer questions, generate text, or create images.

Why it matters:

Because not all models are created equal, and choosing the right one matters. Some are free, some are expensive.

Multimodal

AI that sees, hears, AND reads

TL;DR: AI that handles multiple types of input -- text, images, audio, video -- all at once. A Swiss Army knife instead of just a blade.

Early AI was like someone wearing blinders. A text AI could only read words. An image AI could only look at pictures. A speech AI could only hear audio. None of them could do what a 5-year-old does effortlessly -- see a picture, hear someone talking about it, and read a caption all at the same time.

Why it matters:

Because multimodal AI is what makes AI truly useful in everyday life. Take a photo of a restaurant menu in Japanese and get an English translation.

NLP

Natural Language Processing

TL;DR: Teaching computers to understand human language -- sarcasm, slang, and all. It's why Siri sometimes gets you and sometimes thinks you said something completely unhinged.

You know how you can say "that's sick" and your friend knows you mean "that's awesome" and not "that's diseased"? Context, tone, slang -- your brain handles all of that automatically. Computers? Not so much.

Why it matters:

Because NLP is the technology that made AI feel accessible to normal people. You don't need to learn to code to use ChatGPT -- you just talk to it.

Parameter

The dials inside an AI's brain

TL;DR: The internal 'dials' that get tuned during training. GPT-4 has hundreds of billions of them. More parameters generally means smarter (and way more expensive).

Imagine a massive mixing board in a recording studio -- thousands of dials and sliders that shape the sound. Each one controls something slightly different. Together, they determine what the final music sounds like. An AI model's parameters are those dials.

Why it matters:

Because parameter count is one of the first things people mention when comparing AI models, and it helps you understand the cost conversation. More parameters generally means more capable (up to a point) but also more expensive to train and run.

Prompt

The thing you type into AI

TL;DR: The instruction you give an AI. Better prompts = better answers. It's like asking good questions at a bar.

A prompt is just whatever you type into an AI. That's it. When you ask ChatGPT "What should I make for dinner?" -- that's a prompt.

Why it matters:

Because the difference between a useless AI response and a mind-blowing one is usually just how you asked. People who write good prompts get 10x more out of the same tools as people who don't.

RAG

Retrieval-Augmented Generation

TL;DR: Teaching AI to Google things before answering you. Reduces the 'making stuff up' problem.

Imagine you're taking a test. In scenario A, you have to answer everything from memory. In scenario B, you get to look through your notes first, then answer. You're going to do way better in scenario B, right?

Why it matters:

Because RAG is the main solution to the hallucination problem. When an AI makes stuff up, it's usually because it's guessing from memory.

Reinforcement Learning

Learning by trial, error, and treats

TL;DR: AI that learns by trying stuff and getting rewards or punishments. Like training a dog -- do the right thing, get a treat. Do the wrong thing, no treat.

Think about how you learned to ride a bike. Nobody handed you a manual. You got on, wobbled, fell, got back up, wobbled less, and eventually figured it out. Each fall taught you what NOT to do. Each successful moment reinforced what worked. That's reinforcement learning.

Why it matters:

Because reinforcement learning is what makes AI actually good at complex tasks. It's behind recommendation algorithms, game-playing AI, robotics, and the fine-tuning that makes chatbots helpful instead of unhinged.

Supervised Learning

Learning from labeled flashcards

TL;DR: Teaching AI with labeled examples: 'This is a cat. This is a dog. Now YOU tell me.' Flashcards for robots.

Remember flashcards? Someone shows you a card with a picture and the answer on the back. France -> Paris. Japan -> Tokyo. After enough cards, you know the answers without looking.

Why it matters:

Because supervised learning is behind most of the AI you interact with daily. When your phone recognizes your face, when Gmail catches spam, when your bank flags a suspicious transaction -- that's all supervised learning.

Text-to-Image

Type words, get pictures

TL;DR: You type 'a cat riding a motorcycle through space' and AI creates that image. The most mind-blowing parlor trick in tech right now.

Type a sentence. Get a picture. That's it. That's text-to-image AI.

Why it matters:

Because visual creation just became a superpower anyone can have. Need an image for a presentation?

Token

The chunks AI reads text in

TL;DR: AI doesn't read words -- it reads chunks called tokens. Think of them as syllables for robots.

You read words. AI reads tokens. A token is roughly a piece of a word -- sometimes a whole word, sometimes just a chunk. The word "hamburger" might be two tokens: "ham" and "burger." The word "the" is one token. A long word like "unbelievable" might be three or four tokens.

Why it matters:

Because tokens explain two things that will confuse you otherwise: why AI sometimes "forgets" what you told it earlier in a long conversation (it ran out of token space), and why AI services cost what they cost (you're paying per token).

Training Data

You are what you eat (AI edition)

TL;DR: The examples AI learns from. Feed it garbage, it produces garbage. Like raising a kid on nothing but reality TV -- don't be surprised by the results.

You know the saying "you are what you eat"? For AI, it's "you are what you trained on." Training data is the massive collection of examples that an AI system learns from. For a language model like ChatGPT, that's basically a huge chunk of the internet -- books, websites, Wikipedia, forums, code repositories, the works.

Why it matters:

Because understanding training data helps you understand AI's limitations. When an AI gives a biased answer, it's often because the training data was biased.

Transformer

The architecture behind modern AI

TL;DR: Not the movie robots. It's the math breakthrough from 2017 that made ChatGPT, Claude, and basically all modern AI possible. The engine inside the car.

In 2017, a team at Google published a paper called "Attention Is All You Need." Sounds like a Beatles song, but it quietly changed the world. They invented the transformer -- a new way for AI to process information -- and everything since has been built on top of it.

Why it matters:

Because when people say "AI got good suddenly," this is why. One architectural breakthrough in 2017 unlocked everything -- chatbots, image generators, code assistants, translation, all of it.

Vibe Coding

Claude Code

AI coding in your terminal

TL;DR: Anthropic's coding tool that works in your terminal. Point it at your project and it reads, writes, and edits code for you.

Cursor and Windsurf are like AI assistants that live inside your code editor. Claude Code takes a different approach -- it lives in your terminal (that black screen with the blinking cursor that hackers use in movies).

Why it matters:

Because Claude Code represents where vibe coding is heading -- AI that doesn't just write snippets but works on entire projects. It can create files, edit multiple files at once, understand how everything connects, and make changes across your...

Copilot

GitHub's AI coding assistant

TL;DR: GitHub's AI that autocompletes your code as you type. Like autocorrect, but for programming. Made by Microsoft.

You know how your phone predicts the next word when you're texting? Copilot does that for code. You start typing, and it suggests the rest of the line -- or the rest of the entire function. Accept the suggestion with a Tab key, and you just wrote code 5x faster.

Why it matters:

Because Copilot was the gateway drug that showed the world AI could write code. It normalized the idea that AI assistance is a tool, not cheating.

Cursor

AI-powered code editor

TL;DR: A code editor with AI built in. You describe what you want, it writes the code. The ultimate vibe coding tool.

Imagine Microsoft Word, but for code, and it has a genius assistant built right in. You can highlight a section and say "make this faster," or type "build me a login page," and it just... does it. That's Cursor.

Why it matters:

Because Cursor (and tools like it) are the reason non-programmers can now build software. If you've ever had an app idea but thought "I'd need to learn to code first," Cursor changes that equation.

Vibe Coding

Building apps with AI assistance

TL;DR: Writing code by describing what you want to an AI, then letting it write the actual code. It's how I built my first app.

Imagine walking into a construction site and saying, "I want a house with three bedrooms, a big kitchen, and a wraparound porch." And then a crew of expert builders just... builds it. You don't need to know how to frame a wall or run electrical. You just describe what you want, watch it happen, and say "actually, make that kitchen bigger."

Why it matters:

Because the barrier to building software just dropped to near zero. If you have an idea for an app, a website, or a tool -- you can actually build it now.

Windsurf

AI code editor with flow

TL;DR: Another AI code editor, known for smooth multi-file editing. Like Cursor's chill younger sibling.

Windsurf is another AI-powered code editor, similar to Cursor but with its own personality. If Cursor is the power tool, Windsurf is the one that feels smoother and more intuitive. It's particularly good at working across multiple files at once -- like when you need to change something that affects several parts of your project.

Why it matters:

Because the more AI coding tools you know about, the better equipped you are to find the one that clicks for you. Some people prefer Cursor, some prefer Windsurf, some use both.

Building Blocks

Algorithm

Step-by-step instructions for computers

TL;DR: A set of step-by-step instructions to solve a problem. Like a recipe, but for computers. Every time Netflix recommends a show, that's an algorithm at work.

A recipe is an algorithm. "Preheat oven to 350. Mix flour, sugar, eggs. Pour into pan. Bake for 30 minutes." A set of clear steps that, if followed correctly, produce a predictable result. Algorithms are the same thing, but for computers.

Why it matters:

Because algorithms shape your reality more than you think. They decide what news you see, what products get recommended to you, whether your resume gets seen by a human, and what interest rate you're offered.

API

Application Programming Interface

TL;DR: A menu that lets apps order from other apps. Your weather app uses an API to ask a weather service for the forecast.

Think of a restaurant. You don't walk into the kitchen and cook your own food. You look at a menu, tell the waiter what you want, and food appears. You don't need to know how the kitchen works. You just need to know what's on the menu.

Why it matters:

Because APIs are how you can build powerful apps without building everything from scratch. Want your app to send emails?

Automation

Let robots do the boring stuff

TL;DR: Making computers do repetitive tasks so you don't have to. If you copy-paste the same data between apps every day, that's a job for automation, not a human.

Every day, millions of people do the same boring tasks on their computers: copy data from one spreadsheet to another, send the same follow-up email, rename and organize files, update a report with new numbers. They do it manually, every single time, like it's 1995.

Why it matters:

Because time spent on repetitive tasks is time you'll never get back. Even automating ONE daily task that takes 15 minutes saves you over 60 hours a year.

Backend

What happens behind the scenes

TL;DR: The behind-the-scenes part of an app -- databases, servers, logic. The kitchen to the frontend's dining room.

Back to our restaurant analogy. If the frontend is the dining room, the backend is the kitchen. It's where the actual work happens -- the cooking, the prep, the inventory management. Customers never see it, but without it, there's no food.

Why it matters:

Because when someone tells you they need a "backend" for your app idea, they're saying they need the engine and plumbing -- not just the pretty face. If you're vibe coding, you'll need to decide on your backend approach: do you write your own, use...

Bandwidth

How much data fits through the pipe

TL;DR: How much data can flow through a connection at once. Think of a highway -- more lanes means more cars can pass. More bandwidth means faster downloads.

Imagine a highway. A one-lane road can only handle a few cars at a time. A six-lane highway can handle a lot more. Bandwidth is the number of lanes on your internet highway -- how much data can flow through at once.

Why it matters:

Because bandwidth determines what you can do online. Video calls, streaming, large file uploads, cloud gaming -- all bandwidth-hungry activities.

Cache

Saving a copy so you don't have to fetch it again

TL;DR: Keeping a nearby copy of something so you don't have to go get it every time. Like photocopying a library book so you don't have to drive back for every reference.

You're writing a paper and need to keep referencing the same book. Option A: drive to the library every single time you need a quote. Option B: photocopy the relevant pages and keep them on your desk. Option B is caching.

Why it matters:

Because caching is why "clear your cache" fixes so many problems. When a website looks broken or shows old content, it's often serving you a stale cached version.

CDN

Content Delivery Network

TL;DR: Copies of your website stored around the world so it loads fast everywhere. Like having franchise locations instead of one store that everyone has to drive to.

Imagine there's only one pizza shop in the entire country, located in Kansas. If you live in Kansas, great -- fresh pizza in 10 minutes. If you live in New York? Your pizza arrives cold and 3 days late. The solution? Open pizza shops all over the country. That's a CDN.

Why it matters:

Because CDNs are why the internet feels fast. Without them, every website would load at the speed of the farthest data center.

The Cloud

Someone else's computer

TL;DR: It's just someone else's computer that you rent. When someone says 'it's in the cloud,' they mean it's stored on a server in a data center somewhere. That's it.

"The cloud" is one of the most over-mystified terms in tech. Ready for the truth? It's just someone else's computer. That's it. When your photos are "in the cloud," they're on a computer in a massive data center owned by Apple, Google, or Amazon.

Why it matters:

Because almost everything you use runs on the cloud. Gmail, Netflix, Spotify, your banking app, every AI tool -- all cloud-based.

Cookie

How websites remember you

TL;DR: A tiny file websites save on your computer to remember who you are. That's why Amazon remembers your cart and why ads stalk you across the internet.

You walk into your favorite coffee shop. The barista sees you and says, "The usual?" They remember you. Now imagine if every single time you walked in, they had no idea who you were. "Welcome! What's your name? What do you like? Do you have an account?" Every. Single. Time. That would be exhausting.

Why it matters:

Because cookies control a lot of your online experience, and understanding them helps you manage your privacy. Those "accept cookies" banners?

Database

Where data lives

TL;DR: A structured place to store data. Think of it as a really organized digital filing cabinet that can find anything instantly.

You know how your phone has a contacts app? Each contact has a name, phone number, email, maybe an address. Now imagine that same idea, but for a million contacts, and instead of contacts it could be anything -- user accounts, recipes, orders, messages, product listings.

Why it matters:

Because every app that remembers anything uses a database. If you build something that needs to save user data, store content, or keep track of anything -- you need a database.

Debugging

Finding out why it's broken

TL;DR: The art of finding and fixing what's broken in code. 90% of programming is just figuring out why the thing you wrote doesn't work. Named after an actual bug in a computer in 1947.

Programming isn't mostly writing code. It's mostly figuring out why the code you wrote doesn't work. That process is called debugging -- hunting down the "bug" (the error) and squashing it.

Why it matters:

Because if you ever write any code -- even with AI doing most of the work -- you'll need to debug. AI coding assistants are great at generating code but they still produce bugs.

Deploy

Putting your app on the internet

TL;DR: Taking your app from 'works on my computer' to 'works for everyone on the internet.' The moment it gets real.

You've built an app. It works on your computer. Great. But right now, you're the only person who can use it. Deploying is the process of putting it on the internet so anyone with the link can access it.

Why it matters:

Because deploying is the moment your project goes from "hobby" to "real." It's the most satisfying moment in building anything -- seeing your creation live on the internet with a real URL you can share. And with modern tools, there's genuinely no...

DNS

The internet's phone book

TL;DR: Translates 'google.com' into the actual numbers computers use to find each other. Without DNS, you'd need to memorize IP addresses like 142.250.80.46 for every website.

You don't memorize phone numbers anymore -- you just tap a name in your contacts and your phone looks up the number. DNS (Domain Name System) does the same thing for the internet. You type "google.com" and DNS translates that into the actual numerical address (like 142.250.80.46) that computers use to find each other.

Why it matters:

Because DNS is one of the most fundamental (and fragile) pieces of internet infrastructure. When "the internet is down" but your WiFi is fine, it's often a DNS problem.

Docker

Ship it exactly as you built it

TL;DR: A way to package software so it runs the same everywhere. Like shipping a frozen dinner instead of raw ingredients -- no more 'but it worked on MY computer' excuses.

Ever had a recipe turn out perfectly in your kitchen but fail completely in someone else's? Different oven, different altitude, different brand of flour -- tiny differences that shouldn't matter but totally do. Software has the same problem.

Why it matters:

Because Docker is everywhere in modern software development. If you ever need to run an open-source AI model locally, deploy a web app, or set up a development environment, you'll likely encounter Docker.

Encryption

Secret codes for the digital age

TL;DR: Scrambling data so only the right person can read it. Like writing a letter in a secret code that only your friend can decode. Without it, the internet would be a free-for-all.

Remember passing notes in class? If you wrote in plain English and the teacher intercepted it, game over. But if you and your friend had a secret code? The teacher sees gibberish. Only your friend can decode it. That's encryption.

Why it matters:

Because encryption is the only thing standing between your private data and everyone who'd love to steal it. Your passwords, your bank transactions, your medical records, your messages -- all protected by encryption.

Framework

Pre-built code scaffolding

TL;DR: Pre-built code that gives you a head start. Like buying a house with the foundation already poured instead of starting from dirt.

Imagine you want to build a house. You could start from absolute scratch -- dig the foundation, pour concrete, frame every wall. Or you could start with a pre-built foundation and frame, and just customize it.

Why it matters:

Because when someone says "what framework should I use?" they're really asking "what foundation should I build on?" Picking the right framework matters. It determines what's easy, what's hard, and what community of developers can help you when you...

Frontend

What users see and click

TL;DR: The part of an app you can see -- buttons, text, images, colors. If you can click it or look at it, it's frontend.

Think of a restaurant again. The frontend is the dining room -- the tables, the menu, the lighting, the way the waiter greets you. It's everything the customer sees and interacts with. The kitchen? That's the backend (we'll get there).

Why it matters:

Because the frontend is what makes or breaks a first impression. Users don't care how fancy your backend is -- if the app looks ugly or confusing, they're leaving.

JSON

How computers share data

TL;DR: The universal language computers use to share structured information. It looks like a bunch of curly braces and quotes, but it's actually super logical and readable.

When two people share information, they use a language they both understand. When two computers share information, they often use JSON (pronounced "jay-son"). It stands for JavaScript Object Notation, but don't let the name scare you -- it's just a way of organizing data that both humans and computers can read.

Why it matters:

Because JSON is the data format you're most likely to encounter if you interact with any AI API, web service, or data tool. When you connect apps together, export data, or work with AI responses programmatically -- it's usually JSON.

Latency

The annoying delay

TL;DR: The delay between asking and getting an answer. When your video call freezes for 2 seconds -- that's latency. Lower is always better.

You click a button. There's a pause. Then something happens. That pause? That's latency. It's the time delay between doing something and seeing the result. In an ideal world, latency would be zero. In reality, data has to travel through wires, get processed by servers, and travel back. That takes time.

Why it matters:

Because latency affects every digital experience you have. Slow websites, laggy video games, glitchy video calls, buffering videos -- that's all latency.

Markdown

Simple formatting with symbols

TL;DR: A dead-simple way to format text using symbols like # for headings and ** for bold. You've probably used it on Reddit or Discord without knowing its name.

Want to make text bold in a Word document? You highlight it and click a button. Want to make text bold in Markdown? You put two asterisks around it: ``bold``. That's it. No buttons, no menus, no mouse required.

Why it matters:

Because Markdown is everywhere in the tech world, and it takes literally 5 minutes to learn. If you're writing on GitHub, Discord, Slack, Notion, Reddit, or countless other platforms -- knowing Markdown makes your text look better with zero effort.

No-Code / Low-Code

Building without writing code

TL;DR: Tools that let you build apps by dragging, dropping, and clicking instead of writing code. Like Squarespace but for apps.

You know how Squarespace lets you build a website without knowing HTML? Drag a photo here, add some text there, choose a template, done. No-code tools do the same thing but for full applications -- databases, user accounts, workflows, and all.

Why it matters:

Because no-code tools are genuinely powerful for MVPs and simple apps. If your idea doesn't need custom code, no-code might get you to launch faster than anything else.

Open Source

Free code anyone can use

TL;DR: Software where the code is public and free. Anyone can use it, modify it, and learn from it. Most of the internet runs on it.

Imagine if Coca-Cola published their secret recipe and said "here, anyone can make this. Modify it if you want. Just don't say it's the original." That's open source, but for software.

Why it matters:

Because as a builder, open source is your superpower. Need a database?

Repository

Your project's time-traveling folder

TL;DR: A project folder that remembers every change ever made. Like Google Docs version history, but for your entire codebase. Usually called a 'repo' for short.

A regular folder on your computer is dumb. Save a file, and the old version is gone forever. Delete something by accident? Tough luck. A repository (repo for short) is a smart folder. It remembers everything -- every change, every addition, every deletion. And you can go back to any point in time.

Why it matters:

Because if you ever build anything with code -- even a simple website -- a repository will save your sanity. Made a mistake?

Serverless

There are still servers (you just don't care)

TL;DR: There ARE still servers -- you just don't manage them. Like taking an Uber instead of owning a car. Someone else handles the maintenance, insurance, and parking.

The name is a lie. There are absolutely servers. You just don't have to think about them.

Why it matters:

Because serverless is why small teams can build apps that serve millions of users without a dedicated infrastructure team. If you ever build something with AI (or any web tool), serverless platforms like Vercel, Cloudflare Workers, or AWS Lambda let...

SSL/TLS

The padlock in your browser

TL;DR: The padlock icon that means your connection is secure. Without it, sending your password would be like shouting your credit card number across a crowded room.

See that little padlock icon in your browser's address bar? That means SSL (or technically TLS, its newer version) is protecting your connection. Without it, everything you send to a website -- passwords, credit card numbers, messages -- travels in plain text that anyone snooping on the network could read.

Why it matters:

Because SSL is your first line of defense for online privacy. Before entering any sensitive information on a website, check for that padlock.

Vector Database

A database that understands meaning

TL;DR: A special database that finds things by meaning instead of exact matches. Like a librarian who finds books by vibes instead of just by title.

A regular database is like a filing cabinet. You look something up by its exact label. Search for "apple pie recipe" and it finds documents with those exact words. If the recipe is titled "Grandma's Famous Apple Tart," too bad -- the regular database misses it.

Why it matters:

Because vector databases are what make AI actually useful for YOUR stuff. Want AI to answer questions about your company's documents?

Version Control

A time machine for your code

TL;DR: A system that tracks every change to your project so you can undo mistakes. Like a time machine: 'Take me back to Tuesday when everything still worked.'

Have you ever saved over a file and immediately regretted it? Or had a project going great, then made a bunch of changes that broke everything, and wished you could just go back? Version control is the "undo button" for your entire project -- but infinitely more powerful.

Why it matters:

Because version control isn't just for professional developers anymore. Anyone building a website, writing a book, or managing documents can benefit from never losing work.

Webhook

Automatic notifications between apps

TL;DR: An automatic notification from one app to another when something happens. Instead of checking your mailbox every 5 minutes, the mailman rings your doorbell.

You're waiting for a package. Option A: walk to your mailbox every 5 minutes to check if it arrived. Option B: the delivery driver rings your doorbell when it shows up. A webhook is the doorbell.

Why it matters:

Because webhooks are the glue that connects modern apps. If you've ever used Zapier, IFTTT, or any automation tool -- they rely heavily on webhooks.

Buzzwords

AGI

Artificial General Intelligence

TL;DR: AI that can do everything a human can. We don't have it yet. Anyone who says we do is selling something.

Right now, AI is really good at specific things. ChatGPT is great at writing but can't drive a car. A self-driving AI can navigate roads but can't write a poem. Each AI is a specialist -- amazing at one thing, useless at others.

Why it matters:

Because people throw "AGI" around like it's right around the corner, and it shapes investment decisions, policy debates, and a lot of fear. When you see a headline about AGI, you should know it's still theoretical.

Deep Learning

Neural networks with many layers

TL;DR: Neural networks with lots of layers stacked up. The 'deep' just means there are many layers. That's literally it.

Remember neural networks? Layers of artificial neurons that process data? "Deep learning" is just what happens when you stack a LOT of those layers together. That's it. The "deep" refers to the depth of the network -- more layers = deeper = deep learning.

Why it matters:

Because "deep learning" is the buzzword people use when they want to sound impressive about AI. Now you know it just means "a neural network with lots of layers." You can nod knowingly at dinner parties.

Neural Network

Loosely inspired by brains

TL;DR: A computer system loosely inspired by how brain cells connect. It's how AI learns patterns. Don't worry -- it's not actually a brain.

Your brain has about 86 billion neurons connected by trillions of synapses. When you learn something -- like recognizing your mom's face or catching a ball -- your neurons form patterns. Do something enough times, and the pattern gets strong.

Why it matters:

Because neural networks are the foundation of basically all modern AI. When someone says "machine learning" or "deep learning" or "AI model," they're almost always talking about some flavor of neural network.

Money & Business

Churn

When customers leave

TL;DR: The rate at which customers cancel their subscriptions. The silent killer of subscription businesses.

Imagine you have a bucket full of water (your customers). You're pouring new water in (new signups). But there's a hole in the bottom (cancellations). The water leaking out? That's churn.

Why it matters:

Because churn determines whether a subscription business grows or dies. If you're thinking about building anything with recurring revenue, reducing churn is just as important as getting new customers.

MRR

Monthly Recurring Revenue

TL;DR: How much money your subscription business makes every month like clockwork. The number that makes investors excited.

Imagine you have 100 customers each paying \$10/month for your software. Your MRR is \$1,000. Next month, if nobody cancels, that \$1,000 shows up again automatically. That's the beauty of recurring revenue -- you don't have to re-sell anything.

Why it matters:

Because if you're building any kind of subscription product -- even a simple newsletter with a paid tier -- MRR is how you measure success. It's also the number that tells you whether you can quit your day job.

MVP

Minimum Viable Product

TL;DR: The 'good enough to test' version of your product. Ship the skateboard, not the half-built car.

Say you want to build a ride-sharing app like Uber. You could spend two years building every feature -- scheduling, ratings, payment splitting, corporate accounts, driver analytics. Or you could build the simplest possible version: a button that connects a rider to a driver. That's your MVP.

Why it matters:

Because this concept will save you massive amounts of time. Whether you're building a SaaS, a content site, or a physical product -- start with the MVP.

Product-Market Fit

When people actually want what you built

TL;DR: The magical moment when people want your product so badly they'd be upset if you took it away. The hardest thing in startups.

Imagine you set up a lemonade stand. If people walk by and shrug, you don't have product-market fit. If people are lined up around the block fighting for your lemonade -- you've got it.

Why it matters:

Because finding product-market fit is the single most important thing for any new product. You can have the best marketing, the prettiest design, and the most advanced technology -- but if people don't actually need what you built, none of it...

SaaS

Software as a Service

TL;DR: Software you pay for monthly instead of buying once. Netflix for apps. The holy grail of tech businesses.

Remember when you used to buy software in a box at Best Buy? You'd pay \$200 for Microsoft Office, install it from a CD, and that was it. You owned it.

Why it matters:

If you're thinking about building anything in tech, SaaS is the business model everyone chases. The magic word is "recurring revenue" -- money that shows up every month without you having to re-sell anything.

Tools & Platforms

ChatGPT

The AI tool that started it all

TL;DR: OpenAI's chatbot that made AI mainstream. It's the one your coworkers won't shut up about.

ChatGPT is the reason you're reading this site. Before it launched in November 2022, AI was something that existed in research labs and sci-fi movies. Then OpenAI made a website where anyone could talk to an AI, and the entire world lost its mind.

Why it matters:

Because ChatGPT is probably the most practical AI tool you can start using right now. It's free (the basic version), it runs in your browser, and it can save you hours every week if you learn to use it well.

Claude

Anthropic's AI -- the thoughtful one

TL;DR: Anthropic's AI assistant. Like ChatGPT's more careful, thoughtful cousin who actually reads the whole document.

If ChatGPT is the fast-talking salesman who always has an answer, Claude is the thoughtful friend who actually pauses to think before responding. Made by a company called Anthropic (founded by former OpenAI researchers), Claude is an AI assistant that's known for being careful, nuanced, and really good at long, complex tasks.

Why it matters:

Because having options matters. ChatGPT is great, but it's not the only game in town.

Cloudflare

The internet's bodyguard

TL;DR: A company that protects websites, makes them faster, and hosts them. They sit between your site and the internet like a bouncer.

Imagine a nightclub with a bouncer at the door. The bouncer lets in regular customers, blocks the troublemakers, and keeps the line moving smoothly. Cloudflare is that bouncer for websites.

Why it matters:

Because Cloudflare is one of the best deals in tech. Free SSL certificates, free website hosting, free DDoS protection, free CDN (content delivery network that makes your site fast worldwide).

GitHub

Where code lives

TL;DR: Social media for code. It's where developers store, share, and collaborate on software projects. Owned by Microsoft.

Think of GitHub as Google Docs for code. When you write a document in Google Docs, it's saved online, you can share it with others, track changes, and go back to previous versions. GitHub does the same thing for code.

Why it matters:

Because if you start building anything with code -- even with vibe coding tools -- you'll end up on GitHub sooner or later. It's where you'll store your projects, deploy your websites (many hosting services connect directly to GitHub), and find...

IDE

Integrated Development Environment

TL;DR: A fancy text editor for writing code. Like Microsoft Word but for programmers, with autocomplete, error checking, and 47 features you'll never use.

You COULD write code in Notepad. Technically. You could also cut your lawn with scissors. Both technically work and both are terrible ideas. An IDE is the proper tool -- a text editor on steroids, purpose-built for writing code.

Why it matters:

Because if you ever touch code -- even just editing a website or following a tutorial -- an IDE makes the experience 100 times better than using a plain text editor. And with AI-powered IDEs like Cursor, you barely need to know how to code.

Supabase

Open-source Firebase alternative

TL;DR: A service that gives you a database, authentication, and storage in one package. The backend you don't have to build.

Building a web app? You're going to need a place to store data, a way to handle logins, and probably file storage too. You could build all of that from scratch... or you could use Supabase and get it all in about 10 minutes.

Why it matters:

Because Supabase eliminates the hardest part of building an app -- the backend infrastructure. If you're vibe coding your first app, Supabase is one of the tools that makes it actually possible for a non-expert to build something real.

Vercel

Where Next.js apps live

TL;DR: A hosting platform that makes deploying websites stupidly easy. Push your code, boom -- it's live.

Remember how deploying used to mean renting servers, configuring them, and praying nothing breaks? Vercel said "what if we just... made that easy?"

Why it matters:

Because Vercel removes one of the scariest parts of building a website -- getting it online. If you're vibe coding a project, Vercel means you can focus on building features instead of figuring out servers.

You now speak nerd.

But this is just the beginning. These 75 terms are your foundation. Now it's time to start building.

Keep learning at speaknerd.ai

[Free weekly newsletter](#) | [AI glossary](#) | [Courses](#)

*"I woke up with an idea. By that night I had a live site."
- Steve, building in public*