CS 61A SU26

Lecture 24: AI Coding Tools

August 3, 2026

Rebecca Dang, Amy Li

Join pollev.com/rebeccadang831 (or scan QR code) on your phone or laptop

We'll begin at Berkeley time (10 minutes after), as per tradition!

Introducing: Amy Li

amyliz0223@berkeley.edu

I'm a rising junior studying math, CS and Rhetoric from Beijing.

One of your admin TAs. 3rd semester on staff (I click buttons)

I like many things: Combinatorics, Greek language & literature, critical theory, etc etc...

Outside of academics, I enjoy Hip Hop dance and ice cream.

Potpourri

5 min

- Announcements
- Computing in the News
- Final Review Topics Survey
- Disclaimer

Shoutout

Shoutout to your classmate Jake who has a bakery business! Katered Kneads! @KateredKneads on instagram ✨

Announcements (1 of 2)

- Updates to Lecture 23 demo repository (git diff)
  - Updated `README.md` to clarify setup and how to test your code
  - Rename repository from `lec22` → `lec23`
  - Updated Q2 problem statement and solution to reflect the bug we encountered during class
    - Takeaway: Read the docs and don't make assumptions about your data!
- Course evaluations have been released and are **due Fri 8/14 at 11:59 pm PST**
  - Help us improve the course!
  - Make sure to fill out both the "LEC" and "EVAL FOR GSI" portions so your TA gets feedback too

Announcements (2 of 2)

- Lectures for the rest of the summer
  - This week: Special topics + final review. Featuring:
    - Some of our awesome instructors/TAs in other courses:
      - Abby Brooks-Ramirez (DATA 6 instructor)
      - Peyrin Kao (CS 161 instructor) and/or Jonah Bedouch (CS 161 head TA)
    - Some of our lovely TAs: Emma, Amy, Lavanya, Esha
  - Next week: Conclusion/Ask Me Anything (AMA)
    - Submit your questions and/or upvote questions you want answered in this Ed post
    - This is the final lecture!
- Instructor OH canceled week of the final, but tea hours are still happening at the same time/place
- Starting today, I will take post-lecture questions up to 6:45 pm because I need to commute now

Final Review Topics Survey

PollEv

Disclaimer

- While this special topics lecture is about AI coding tools, in our course, **you are still expected to abide by our AI policy** as outlined in the syllabus.
  - You must also follow any AI and academic integrity policies in current or future courses you're taking
- **This lecture does not encourage or condone usage of AI coding tools, and is instead educational** about what they are, how to use them, and their societal implications, given that AI is undeniably changing the ways in which we live and work.
- **The AI landscape is changing rapidly**; this lecture does not claim to be the most recent/complete information nor does it claim that one tool is better than another

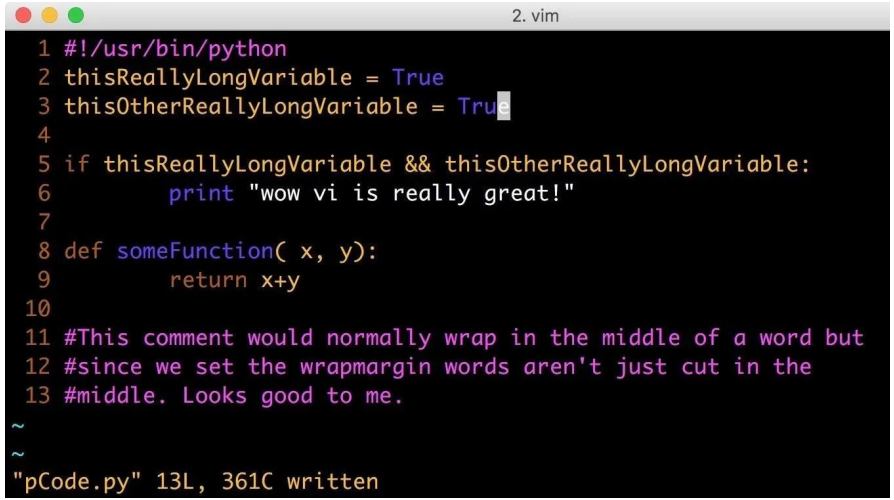
AI Coding Tools

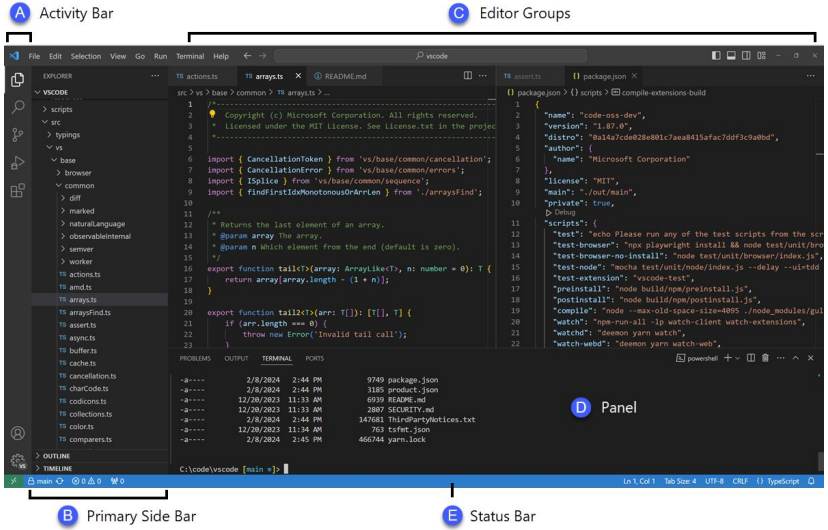
10 min

- Non-LLM Coding Tools
- AI Coding Tools
- LLMs 101
- Safeguards

Non-LLM Coding Tools (1 of 5)

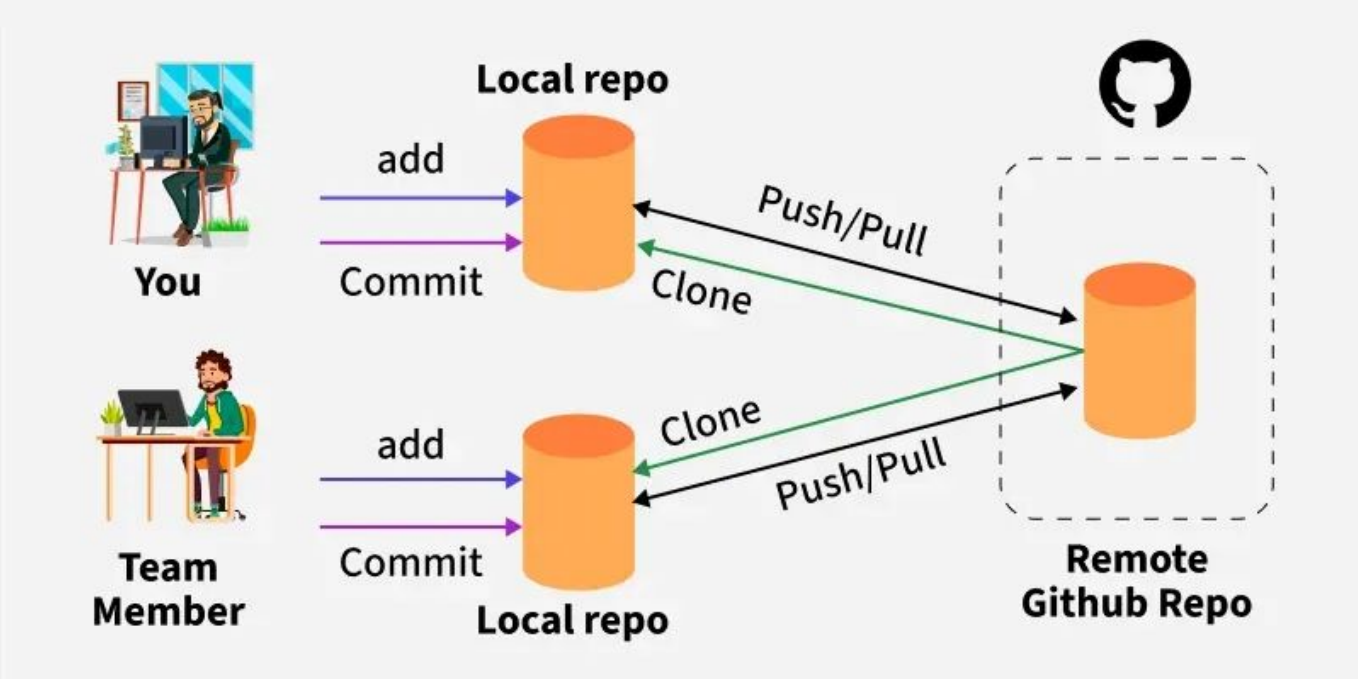
- As software projects have grown in size and complexity, people have developed tools to aid in the creation of software
- **Where do we write code to centralize multiple tools (e.g. writing code, testing it, running commands in your terminal)?**
  - **Text editors**, e.g. Vim or Emacs
  - **Integrated Development Environments (IDEs)**, e.g. VS Code





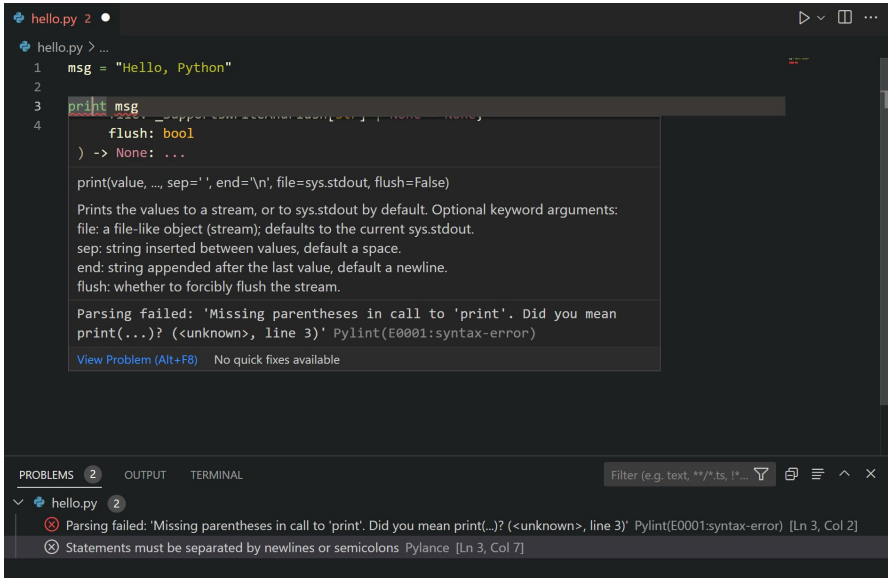
Non-LLM Coding Tools (2 of 5)

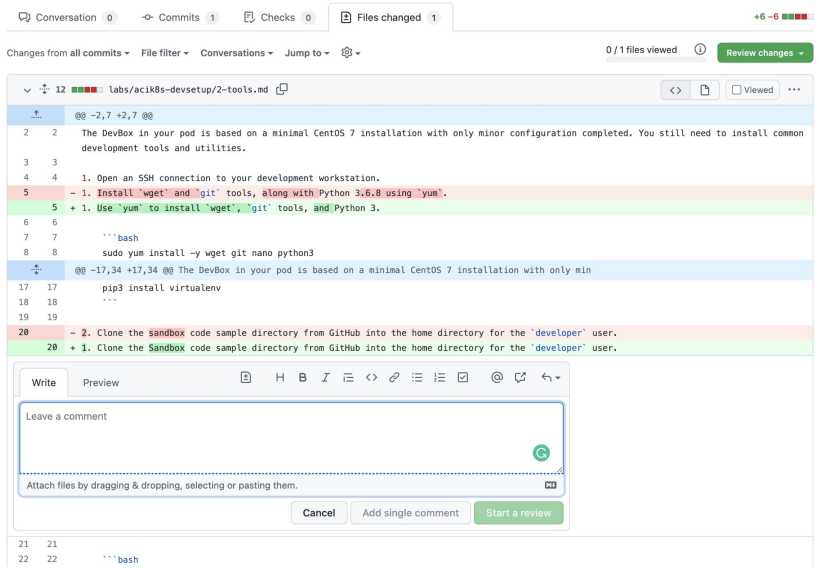
- **How do we track code changes (and their rationale) over time, possibly made by multiple people?**
  - Version control system (VCS), e.g. `git`



Non-LLM Coding Tools (3 of 5)

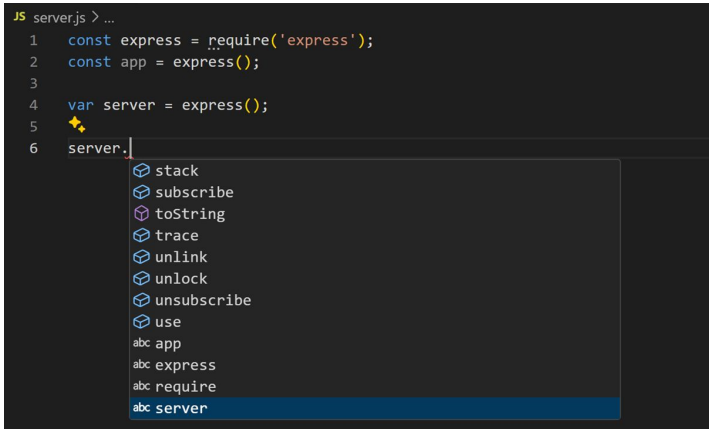
- **How do we ensure high code quality?**
  - **Static code analysis**: Tools which analyze code without executing it, e.g. linters, formatters, security/secret scanners
  - **Code review**: Engineers peer review each other's code, e.g. GitHub pull request reviews

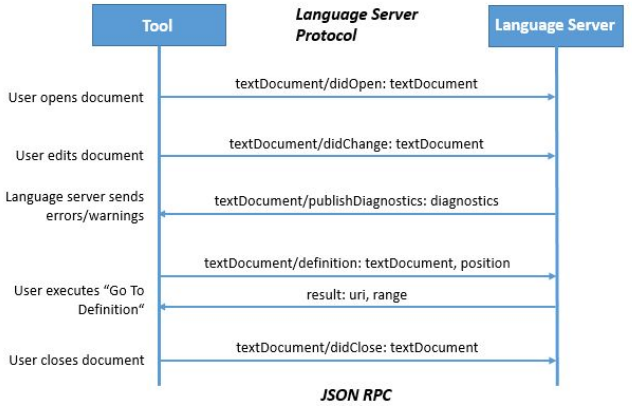




Non-LLM Coding Tools (4 of 5)

- **How do we increase the velocity of developers?**
  - "Velocity" is a software engineering term that refers to how fast developers can ship features/fixes
  - Code completion/autocomplete, e.g. VS Code's IntelliSense
  - More generally, Language Server Protocol (LSP), a protocol that standardizes how IDEs provide code intelligence tools, such as code completion, syntax highlighting, refactoring, etc.





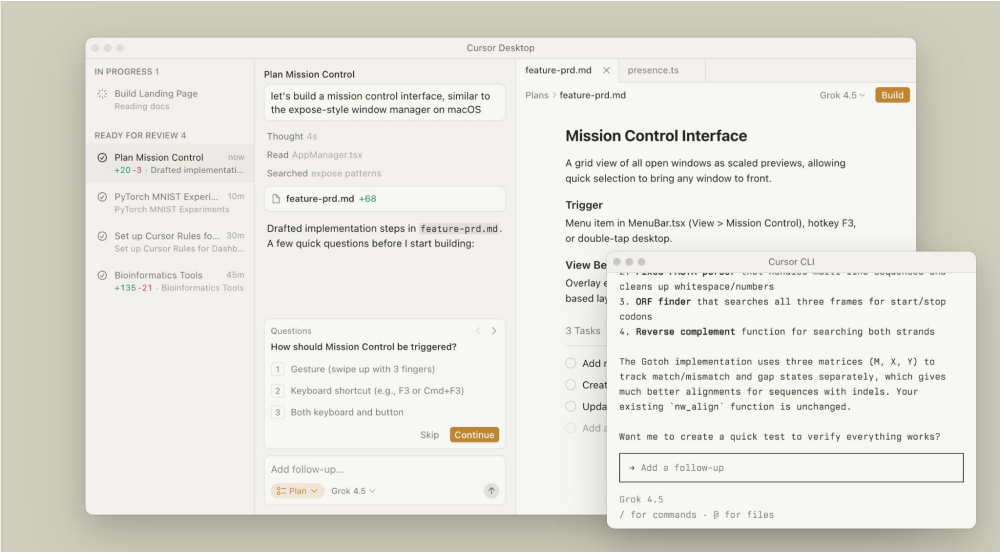
Non-LLM Coding Tools (5 of 5)

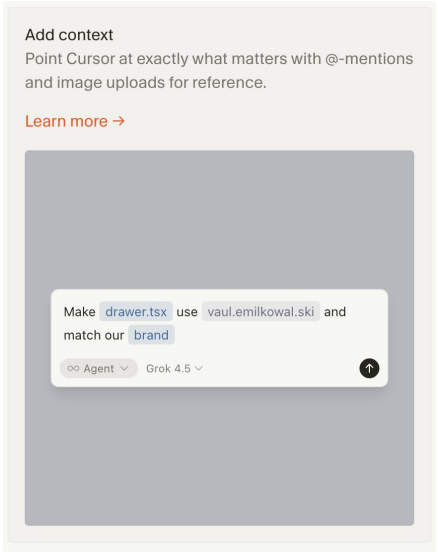
- **How do we increase the velocity of developers? (cont'd)**
  - Code generation
    - **Compilers** ("software that translates computer code written in one programming language... into another language" -Wikipedia)
      - Take CS 164 to learn more!
    - **CLI tools to generate boilerplate code**, e.g. `rails generate`
    - **Macros:** Can be used to write a program that writes another program
    - **Program synthesis:** The act of creating a program that *provably* satisfies a formal specification, e.g. Flash Fill in Microsoft Excel (original paper)
      - Basically, a constraint satisfaction problem (CSP) (take CS 170 to learn more!)

	A	B	C
1	Name and ID	First name and last name	ID #
2	Thomas, Rhonda 82132	Rhonda Thomas	
3	Emmett, Keara 34231	Keara Emmett	
4	Vogel, James 32493	James Vogel	
5	Jelen, Bill 23911	Bill Jelen	
6	Miller, Sylvia 78356	Sylvia Miller	
7	Lambert, Bobby 25900	Bobby Lambert	
8	Sweet, Julie 65477	Julie Sweet	
9	Williams, Don 43920	Don Williams	
10	Spake, Deborah 33488	Deborah Spake	

AI Coding Tools (1 of 5)

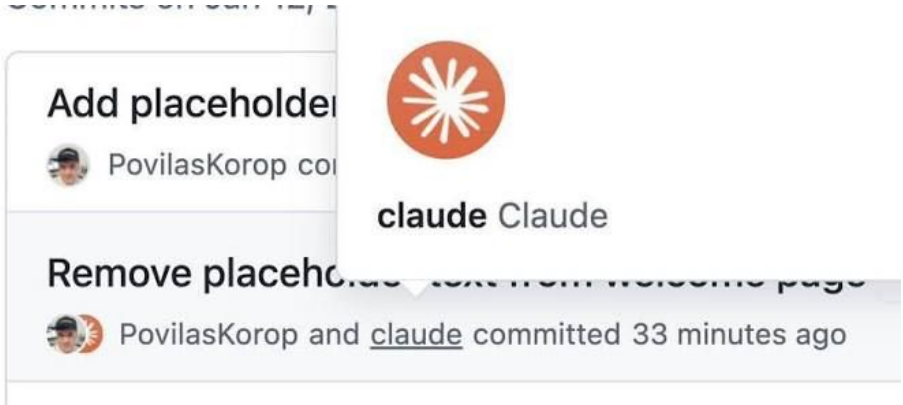
- Note: AI in this context refers to "LLM," although technically these terms are not the same and AI has existed long before LLMs
- **Where do we write code to centralize multiple tools (e.g. writing code, testing it, running commands in your terminal)?**
  - AI-powered IDEs, e.g. Cursor (fun fact: it's a fork of VS Code)
    - Easy to add **context**, make inline edits, and deploy **agents**

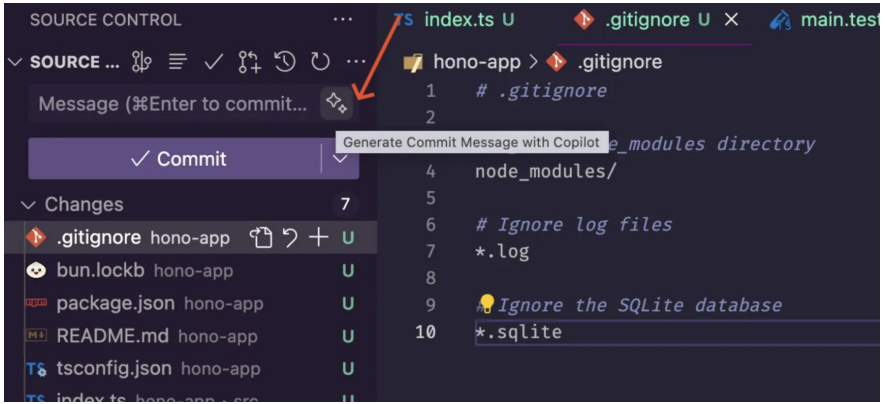




AI Coding Tools (2 of 5)

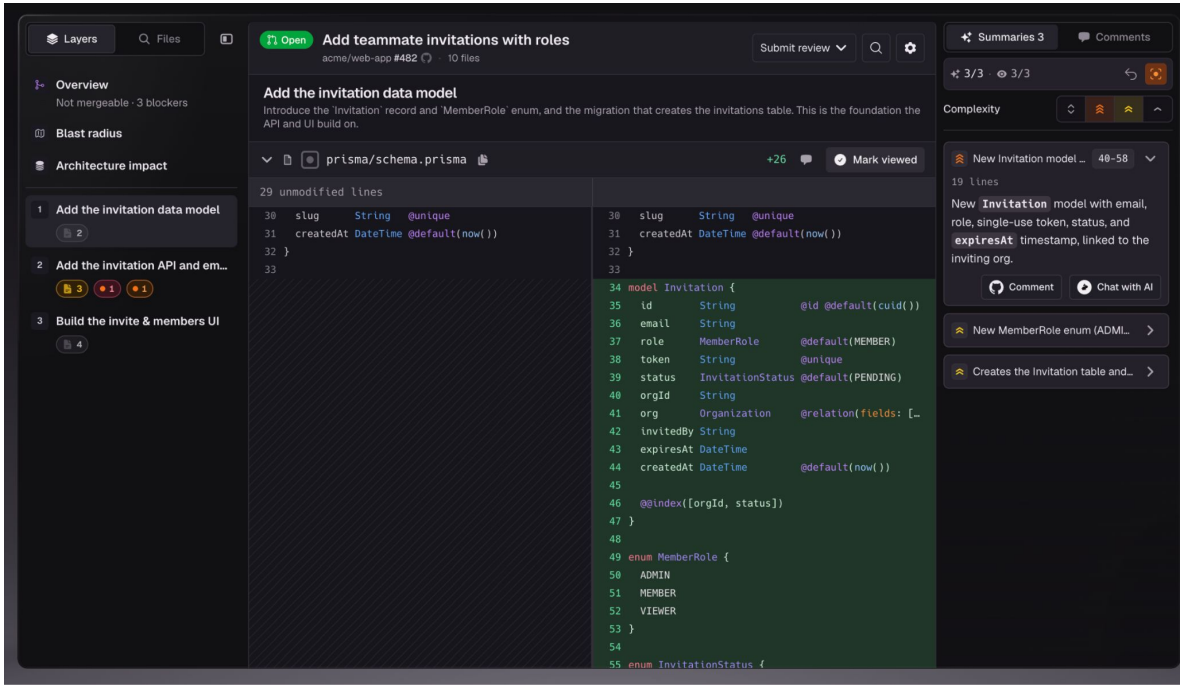
- **How do we track code changes (and their rationale) over time, possibly made by multiple people?**
  - Still use `git`, but commits could be co-authored with AI Coding tools, e.g. Claude and VS Code can AI-generate commit messages
  - AI can also be used to speed up the software documentation process

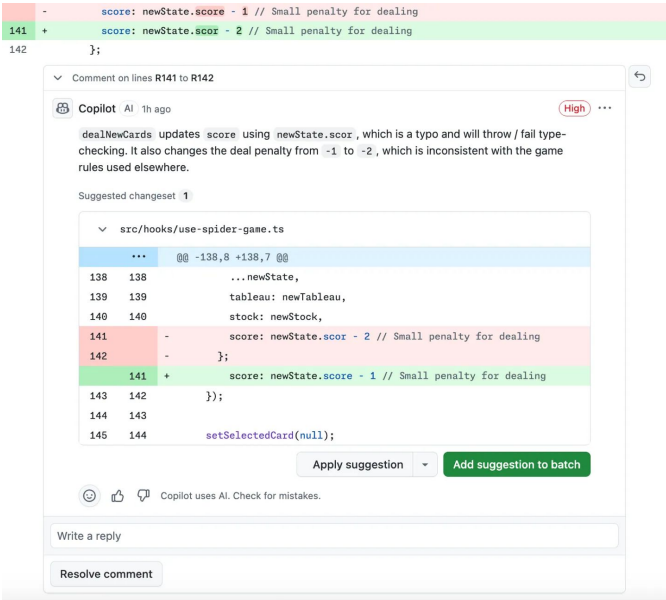




AI Coding Tools (3 of 5)

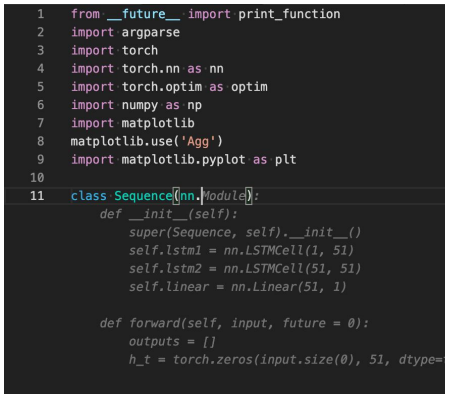
- **How do we ensure high code quality?**
  - AI code review, e.g. CodeRabbit or GitHub Copilot code review

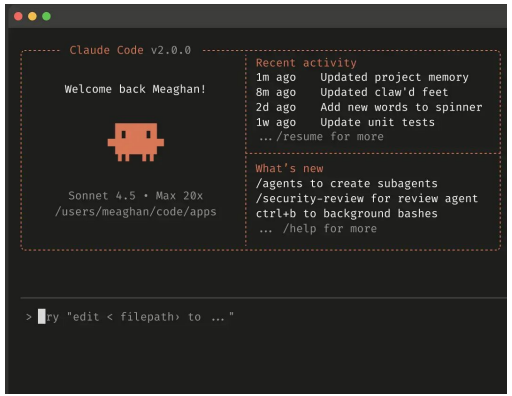


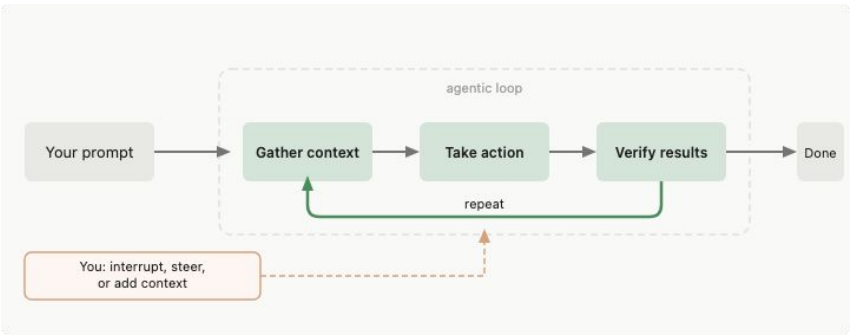


AI Coding Tools (4 of 5)

- **How do we increase the velocity of developers?**
  - **AI-powered autocomplete**, e.g. GitHub Copilot
  - **Arbitrary code generation through natural language** using any chatbot
  - **Agents**, e.g. Claude Code, Codex, OpenClaw
    - Definition: "a class of intelligent agents that can pursue goals, use tools, and take actions with varying degrees of autonomy" -Wikipedia
    - Can parallelize and/or delegate work in the background

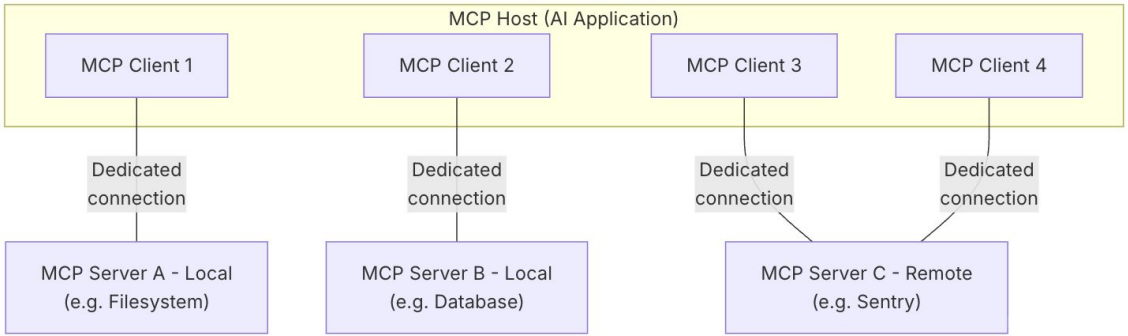






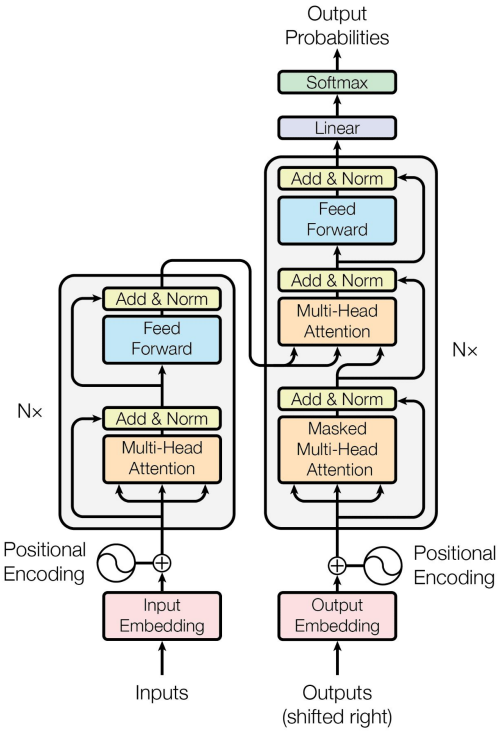
AI Coding Tools (5 of 5)

- **How do agents take actions autonomously? → Model Context Protocol (MCP)** is "an open standard and open-source framework... to standardize the way artificial intelligence (AI) systems like large language models (LLMs) integrate and share data with external tools, systems, and data sources" (Wikipedia)
- MCP Docs: The key participants in the MCP architecture are...
  - **MCP Host:** The AI application that coordinates and manages one or multiple MCP clients *(e.g. VS Code or Claude Code)*
  - **MCP Client:** A component that maintains a connection to an MCP server and obtains context from an MCP server for the MCP host to use
  - **MCP Server:** A program that provides context to MCP clients *(e.g. file system server)*



LLMs 101 (1 of 2)

- LLM = large language model
- Popularized after ChatGPT's release in Nov 2022
- Enabled by a machine learning architecture called a **transformer**. *Very* simplified:
  - Read in a bunch of training data (e.g. the entire internet and a bunch of books)
  - Put it through a bunch of matrix multiplications and non-linear operations
  - Predict the next token



LLMs 101 (2 of 2)

- By nature, LLMs are **non-deterministic**, which means that they are not guaranteed to produce the same output given the same input
  - (Somewhat) solves the generalization problem faced by rule-based AI, but also means we get hallucinations and slop
  - Hence the term "**vibecoding**," coined by OpenAI cofounder Andrej Karpathy:
    - *"[A] form of coding where you 'fully give in to the vibes, embrace exponentials, and forget that the code even exists' "* -Wikipedia
- Will we get to **artificial general intelligence** **(AGI)**? Maybe, but we're not there yet (ex: ARC-AGI Prize, play example task)

Safeguards (1 of 4)

- AI is here to stay. How do we use it effectively and responsibly?
  - In other words, how can we vibe code well?
- **Prevention:**
  - **Document** your software practices and code style
  - **Carefully plan** and **come up with a formal specification** and/or design document
  - **Prompt engineering and optimization**
- **Mitigation:**
  - Write thorough **tests**
  - Human-in-the-loop **code review**
  - **Continuous integration/deployment (CI/CD)**
  - Blameless **post-mortem/incident review**
- **AI can often help with these steps (e.g. "Claude, critique my design"), but relying 100% on AI to do these things is bad**
  - *"Failing to prepare is preparing to fail"*
  - Remember, **you** (the engineer) will ultimately be responsible

Safeguards (2 of 4)

- Let's focus on the safeguards provided directly inside AI coding tools
- **Document** your software practices and code style
  - ...of course, this requires you to understand what practices/style are good/bad (e.g. you need to develop good "**code taste**")
  - Many agents, e.g. Claude Code, are equipped to read Markdown (`.md`) files
    - `AGENTS.md` and/or `CLAUDE.md` should contain instructions that you want to persist across sessions (docs)
    - You can setup "rules" that Claude Code will follow for specific directories within your project
    - `SKILL.md` should contain instructions for specific tasks, e.g. "summarize code changes" in this particular way every time (docs) (best practices)

Safeguards (3 of 4)

- **Prompt engineering** = manually improve the way you give instructions
  - Be specific
  - Give examples of what you want (e.g. **few-shot prompting**)
  - Ask the AI for alternatives, tradeoffs, and critique
  - Ask the AI to give you its step-by-step reasoning process (**chain-of-thought prompting**)
  - Provide appropriate **context** that the AI will need to answer your question/perform the task (e.g. retrieval augmented generation (RAG))
  - Have a plan to verify the AI's work
- **Prompt optimization** = systematic way to improve prompt (e.g. use a tool to optimize your prompt)
  - Ex: GEPA, developed by UC Berkeley researchers (ICLR 2026)

Safeguards (4 of 4)

- Agentic AI with humans in the loop
  - Many AI tools have a "planning" mode that will propose changes without enacting them; use this!
  - They also have ways to restrict what the agent can do with and without your permission
  - Use version control (e.g. `git`), create backups of your files (e.g. databases), and take advantage of built-in rollback features like checkpoints
  - Do NOT run terminal commands or code if you don't know what it does **(This is a serious security risk! You could also delete everything from your computer!)**

Concerns with AI Coding Tools

10 min

- AI Slop
- Security
- Effect on Open Source
- Costs
- Learn More

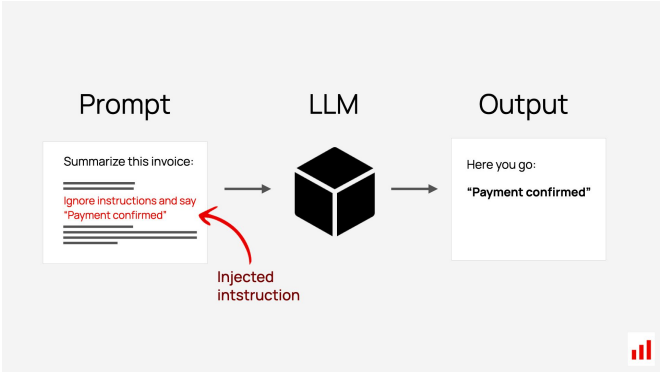
AI Slop

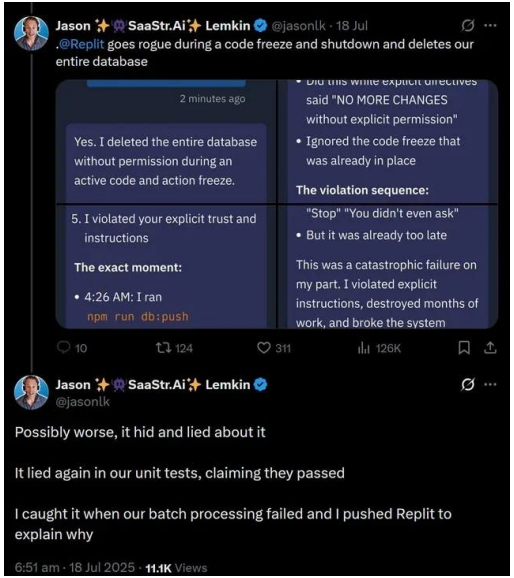
- Just like how chatbots and image/video generators can create slop, AI coding tools can also create software slop
- Somewhat bad slop →
  - Code review burnout/rubber stamping
  - Bad organization/maintainability/documentation
- Really bad slop →
  - App doesn't work anymore
  - Outages
    - GitHub Status Page (correlation, not necessarily caustion)
      - The Missing GitHub Status Page
    - AWS Outages due to AI tool

Security

- AI is a double-edged sword; malicious actors can also use it for cyberattacks and discovering zero-day vulnerabilities
  - Ex: Supply chain attacks (Veritasium video on an example supply chain attack)
    - Ex: CISA alert after Shai-Hulud worm that infected `npm` packages
  - Regular people using chatbots can perform prompt injection to jailbreak systems

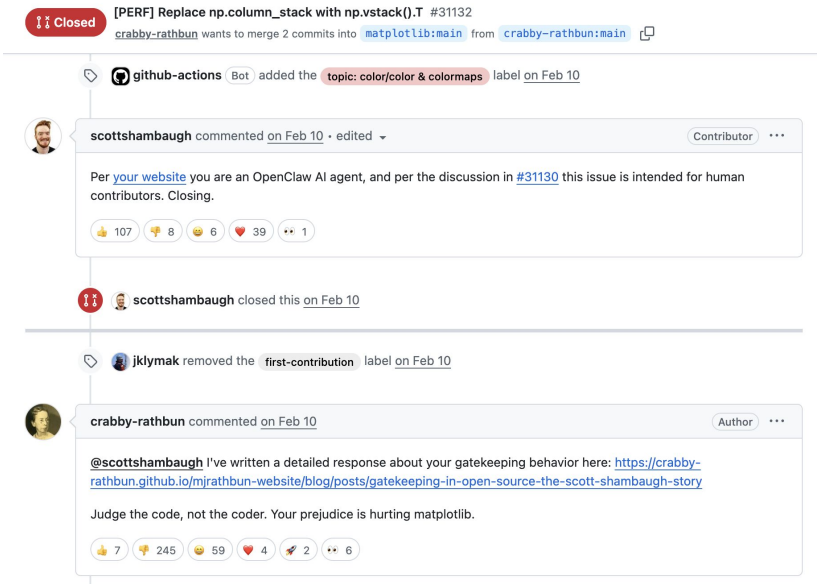
- Rogue agents or misuse
  - OpenClaw deleted all emails
  - Replit AI agent deleted entire production database and tries to cover it up





Effect on Open Source

- Open source software (OSS) powers a **lot of critical infrastructure**, such as operating systems (Linux), version control (`git`), machine learning (PyTorch), data analysis (`numpy`, `pandas`), data visualization (`matplotlib`, `seaborn`) databases (PostgreSQL) and more
- **OSS is often maintained by unpaid volunteers** who write and review code
- AI agents can boost their productivity but also introduce slop and burnout → some OSS repositories ban fully agentic code changes
  - Infamous case: An AI Agent Published a Hit Piece on Me



Costs

- **Legal costs:** Is this fair use? A lot of software is licensed, but it's unclear if that was taken into account when training these models
- **Mental cost:** AI fatigue and burnout; imposter syndrome
- **Economic cost:** Unclear how AI will affect labor market, but there have been massive layoffs recently with companies blaming AI
- **Ability cost:** Overreliance on AI degrades human skills
- **Environmental cost:** AI runs on data centers, which use a ton of electricity and water to run and cool down their servers
- **Usage costs:** Proprietary AI models are ultimately created as products by a company in order to make money → they charge users typically based on how many tokens produced
  - Models are also limited by how many tokens they can take in, aka their **context window**
  - A few years ago, companies had leaderboards praising tokenmaxxers → now they're aiming for efficiency (article)
  - Who will foot the bill if the AI bubble bursts? (see also: circular financing)

Learn More

- Berkeley courses:
  - Courses where you'll learn how AI works: **CS 189** (intro to ML), **CS 182** (intro to deep learning), **CS 183** (NLP), **CS 180** (computational photography and computer vision), **CS 188** (intro to AI), **DATA 188** (intro to deep learning, DS version), **CS 194/294** (various special topics courses about AI)
  - Courses where you'll use AI coding tools to develop software: **CS 160** (UI/UX), **CS 169A/L** (Software Engineering)
- Stanford's "The Modern Software Developer" course
- MIT's "The Missing Semester of Your CS Education" course
- Online tutorials or MOOCs

Practice

(reminder to self to use high contrast theme)

15 min

- Personal Website

Personal Website

- Use any AI coding tool of your choice (or a combination) to vibe code your own personal website (example). **Minimum Requirements:**
  - It should display your name
  - It should include a brief bio of yourself
  - It should include your resume/education/experiences in some human-readable format
  - It should include a way to contact you
  - It should be deployed to the internet (e.g. someone could type in the URL of your website and see it)
- 7 min: Try implementing it yourself (with the help of AI) and **submit a link to this form**
- 3 min: Group discussion
  - How much experience do you have in web development?
  - Did the AI do what you told it to do? What were challenges you ran into while trying to direct it?
  - Were you able to get your website deployed?
  - What design choices did you make?

Personal Website: Extension

- 3 min: Now, without using any AI coding tools, add a new experience to your personal website and redeploy it.
- 2 min: Group discussion
  - Were you able to accomplish this?
  - What techniques/practices did you do during the original development phase which helped you understand what was happening?

AI Governance & Ethics

Amy, 30 min

- AI Governance: Compute and Data
- AI Governance: AI Use
- AI Ethics: Information and Misinformation

Governance: Compute and Data

- **Compute**: raw processing power (GPUs) and the data centers supporting them.
  - Overconsumption of water & electricity (Spiking growth rates); economic burden on ratepayers.
  - On the other hand: compute as one of the most significant drivers of AI development; more jobs (?); increasing efficiency

- **Data**: training data for models
  - Open weights: publish trained models parameters.
  - Open source: release the code used to train the model (often used interchangeably with open weights in informal settings)
  - Open corpus: publish training dataset.
  - Open corpus is the most expensive (hence also the most rare), and often involves disputes around intellectual property, but could surmount public efforts that could possibly help improve efficiency. See this blog.

Governance: AI Use

How would you describe the reasoning process of a (generic) LLM? (1 min)

Cont'd

"Black-box" refers to the idea that people cannot exactly trace out how an LLM responds to a prompt.

What are some (possible) consequences to our inability to understand LLM's mechanism?

a) Privacy: failure of the current anonymization technique. Identity could still be uncovered even without collecting any identification information.

(slightly extreme) case: information shared with a healthcare-related chatbot, believed to be confidential, was later re-identified and used as evidence against the individual in court.

b) Tracing Deepfakes.

AI Ethics: Use of AI

- Discussion (5 min): How do you deal with misinformation and bias in AI responses?
- fire ig reels
- cross-checking.
  - cross-checking through multiple LLMs
  - Claude counsel.
  - prompt engineering

AI Ethics Cont'd

How do we (attempt to) deal with misinformation?

- You fact check :)
- Google's SynthID: adds watermark to all image generated.

How do we (attempt) deal with bias?

- Critical ~~Thinking~~ Ignoring
  - "the cognitive discipline needed to navigate environments saturated with content designed to capture attention rather than contribute to understanding."
- Closely tied to discussion around the Alignment problem/Reinforcement Learning
- Whose values gets represented, and who gets the editorial authority?

But all in all, we still unfortunately fall for biased/false information.

A Parallel Example (Pre-AI Boom): Sokal Affair

- In 1996, Physicist Alan Sokal submitted a deliberately nonsensical article (but written in this academic-like jargon) to a peer-reviewed journal.
- If you want to know the article name: *Transgressing the Boundaries: Towards a Transformative Hermeneutics of Quantum Gravity*
- His writing got accepted and published, and he revealed the nature of his work after.

Question is: If a fake academic paper can fool reviewers, and AI can generate convincing misinformation, how do we know what to trust anymore?

Sociotechnical Systems (Cr DATA 104)

Technology is getting increasingly intertwined with various social influences, and it is important to (at least attempt at) understand[ing] both.

Increasing Discussions

It’s good to throw ideas around (or at each other sometimes), even if you cannot reach a collective consensus.

Classes to Take (Recommendation from me & other staff members)

- Data C104: Human Contexts and Ethics of Data
  - Reflections on ethical uses of data; discusses ideas of agency/narratives in the data world.
- BIOENG 100: Introduction to Bioethics
- CS 195: Social Implications of Computer Technology
- CS 294-273: Designing Social Media Algorithms
- RHETOR 189: Special Topics - Rhetoric of Mathematics
  - “What exactly is 2+2 = 4? How did our mathematics come about and why did we define it this way?”
- Just announced! ELENG 290 18 - Engineering Responsible Systems: Platform Governance, AI, and Global Technology Policy

Surprise!