

- Zed (not going to use but keep an eye for the windows release date)
 - Coming to windows eventually
 - Designed for collaboration with the human programmer and AI
 - Entire Zed code editor is open source under GPL version 3, and scratch-built in Rust down to handcrafted GPU shaders and OS graphics API calls
 - Zed's AI capabilities are also open-source, just like the rest of the editor
 - Can use multiple ai applications (Claude 3.7, GPT, openAI, etc)
 - Secure, no data used for training
 - Paywall for ai however
- Cursor
 - Available for download on microsoft
 - Another ai code editor
 - 3 categories
 - Agent
 - Complete tasks end to end. Does this quickly while keeping programmers in the loop.
 - Finds context using custom retrieval models, cursor can understand a codebase. This reduces the need to manually add context
 - Runs commands by automatically write and run terminal commands. By default, you'll be asked to confirm all commands
 - Loops on errors - cursor can detect lint errors automatically and apply fixes, reducing the need for manual debugging
 - Tab
 - Cursor includes a powerful autocomplete that predicts your next edit. Once enabled, it is always on and will suggest edits to your code across multiple lines, taking into account recent changes
 - Multi-line Edits - Cursor can suggest multiple edits at once, saving time
 - Smart Rewrites - type carelessly, and Cursor will fix your mistakes
 - Cursor Prediction - Cursor predicts your next cursor position so you can seamlessly navigate your code
 - Chat
 - Lets you talks with an AI that sees your codebase.
 - Chat can always see current file and cursor, so you can ask it things like "Is there a bug here?"
 - Can add particular blocks of code to the context with Ctrl+Shift+L or "@"
 - Instant apply
 - Apply the code suggestions from chat back into your codebase by clicking the play button on top of any chat codeblock
 - Codebase Answers

- Use @Codebase or Ctrl Enter to ask questions about codebase
 - Cursor will search codebase to find relevant code to your query
- Reference your code
 - Reference code with @ symbols to be used as context for the AI. Just type @ to see a list of all the files and code symbols in your folder
- Use images
 - Hit the image button under chat to drag an image into the input box to include visual context into chat
- Ask the web
 - Get up-to-date information from the internet with @ Web
 - Cursor will search the web for you and use the latest information to answer your question
- Use documentation
 - Reference popular libraries using @LibraryName, or add your own using @Docs -> Add new doc
- Ctrl K
 - Lets you edit and write code with the AI
 - To edit, select some code, click “Edit” and describe how the code should be changed
 - To generate completely new code, just type Ctrl K without selecting anything
 - Terminate Ctrl K
 - Write terminal commands in plain english.
 - Cursor will convert them into the terminal command you need
 - Quick questions
 - Quick questions can be asked by clicking “quick question” to get your answer right away
- Security Concerns
 - Cursor depends on subprocessors for infrastructure security
 - AWS - sees the code data. Primary host for the infrastructure. Most servers in US, with some in latency critical servers located in AWS regions like Tokyo and London
 - Microsoft Azure - sees the code data. Secondary infrastructure hosted on Azure. All servers in US
 - Google Cloud Platform (GCP) - sees code data. Some more secondary infrastructure on GCP. All servers in US
 - Fireworks - sees code data. Custom models hosted with Fireworks, servers in US, Asia, Europe. May store code data if privacy mode is disabled to speed up inference for models. Zero data retention agreement with Fireworks

- OpenAI - gives AI response. Requests may be sent to OpenAI even if you have Anthropic (or someone else's) model selected in chat (e.g. for summarization) Zero data retention agreement
 - Anthropic - Sees code data. Rely on many of anthropic's models to give AI responses. Requests may be sent to Anthropic even if you have an OpenAI (or something else's) model selected in chat. Zero data retention agreement with Anthropic
 - Google Cloud Vertex API - sees code data. Rely on some Gemini models offered over GCP Vertex API to give AI responses. Requests might be sent to GCP API even if you have another model selected in chat. Zero data retention agreement with vertex
 - xAi - sees code data. Rely on some grok models offered over xAI API to give AI responses. Zero data retention agreement with xAI
 - Turbopuffer - Stores obfuscated code data. Embeddings of indexed codebases, as well as metadata associated with embeddings (obfuscated file names), are stored with Turbopuffer on Google Cloud's server servers in the US. Users can disable codebase indexing
 - Exa and SerpApi - Sees search requests (potentially derived from code data) Used for web search functionality. Search requests potentially derived from code data (e.g. when using @ web in the chat, a separate language model will look at the message, conversation history and current file to determine what to search for, and Exa/SerpApi will see resulting search query)
 - Rest of subprocessors see no code data
- Github Copilot
 - Integrates with leading editors, including Visual Studio Code, Visual Studio, JetBrains IDEs, and Neovim, and, unlike other AI coding assistant, is natively built into GitHub
 - Deals with issues - when assigned issues, GitHub Copilot plans, writes, tests and iterates-using GitHub Actions to run code and deliver ready-to-review pull requests
 - Codes like an insider - GitHub Copilot hooks into MCP servers to draw on data from your repositories and external resources-working like an onboarded team member from day one
 - Human and Agent in the loop - Comment to guide Github Copilot, polish your code for merge, or take over locally in your IDE
 - Agent mode helps make sweeping changes by analyzing code, proposing edits, running tests, and validating results across multiple files
 - Can swap between models like Claude 3.7 Sonnet, OpenAI o1, and Google Gemini 2.0 Flash to crush coding tasks fast or go deep when it counts
 - Plans/Pricing
 - Free
 - 50 agent mode or chat requests per month

- 2,000 completions per month
 - Access to Claude 3.5 Sonnet, GPT-4.1, and more
 - Pro
 - Unlimited completions and chats with access to more models
 - Pro+
 - Maximum Flexibility and model choice
 - Access to all models, including GPT-4.5
 - 30x more premium requests than Copilot free to use latest models, with option to buy more
 - Coding agent (preview)
 - Individual
 - Designed for individual developers, probably not best for this internship
 - Business
 - Features copilot in the coding environment
 - IDE, CLI, and GitHub Mobile
 - Enterprise
 - Includes everything in GitHub Copilot
 - Added layer of customization for organizations and integrates into [GitHub.com](https://github.com) as a chat interface to allow developers to converse with copilot throughout the platform
- Ollama
 - Stands for Omni-Layer Learning Language Acquisition Model
 - Democratizes access to Large Language Models by enabling users to run them locally on their machines
 - Developed with a vision to empower individuals and organizations, Ollama provides user-friendly interface and seamless integration capabilities, making it easier than ever to leverage power of LLMs for various applications and use cases
 - Key Features
 - Local Execution: One of the distinguishing features of Ollama is its ability to run LLMs locally, mitigating privacy concerns associated with cloud based solutions. Bringing AI models directly to users' devices, Ollama ensures greater control and security over data while providing faster processing speeds and reduced reliance on external servers
 - Extensive Model Library: offers access to an extensive library of pre-trained LLMs, including popular models like Llama 3. Users can choose from a range of models to different tasks, domains, and hardware capabilities, ensuring flexibility and versatility in their AI projects
 - Seamless Integration: Ollama seamlessly integrates with a variety of tools, frameworks, and programming languages, making it easier for developers to incorporate LLMs into their workflows.
 - Customization and Fine-Tuning: With Ollama, users have the ability to customize and fine-tune LLMs to suit their needs and preferences. From

prompt engineering to few-shot learning and fine-tuning processes, Ollama empowers users to shape the behavior and outputs of LLMs, ensuring they align with desired objectives.