

# From Copilot to Agent: A Developer's Journey into AI-Powered Engineering



John Groetzinger  
Principal Engineer, CX Engineering

Simon Maple  
Head of Developer Relations, Tesla

# Webex App

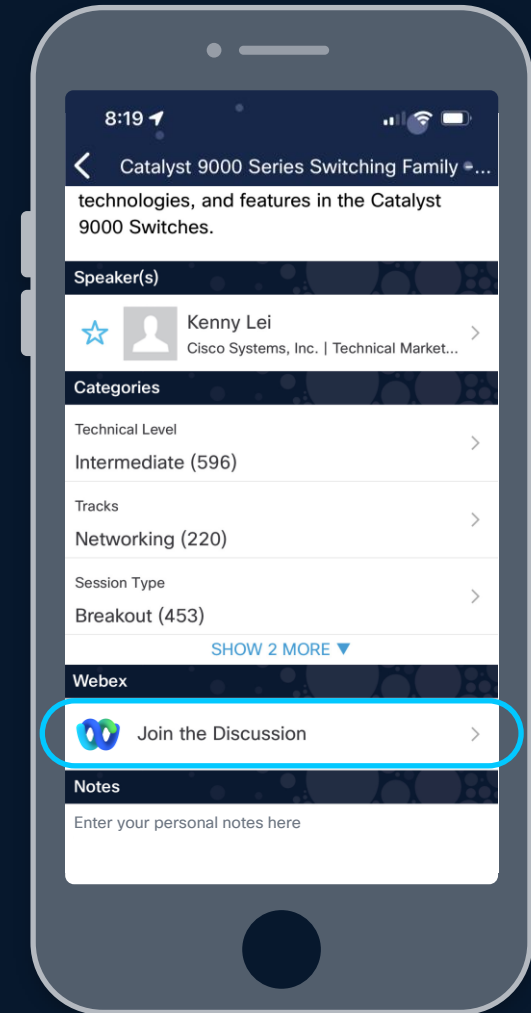
## Questions?

Use Webex app to chat with the speaker after the session

## How

- 1 Find this session in the Cisco Events app
- 2 Click “Join the Discussion”
- 3 Install the Webex app or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

**Webex spaces will be moderated by the speaker until February 27, 2026.**



# Your Presenter

## John Groetzinger



Principal Engineer in Cisco CX Engineering - Generative AI and Network Security  
14+ Years experience in Network Security - roots in Cisco Firewall TAC  
Open Source, Devops and Linux enthusiast



Original Sourcefire employee (Acquired)



360+ bugs filed for Cisco products



Cisco patents and publications



Certified professional cloud architect



13+ years full-stack development  
exp and building innovation



# Some AI products and projects I've been involved in



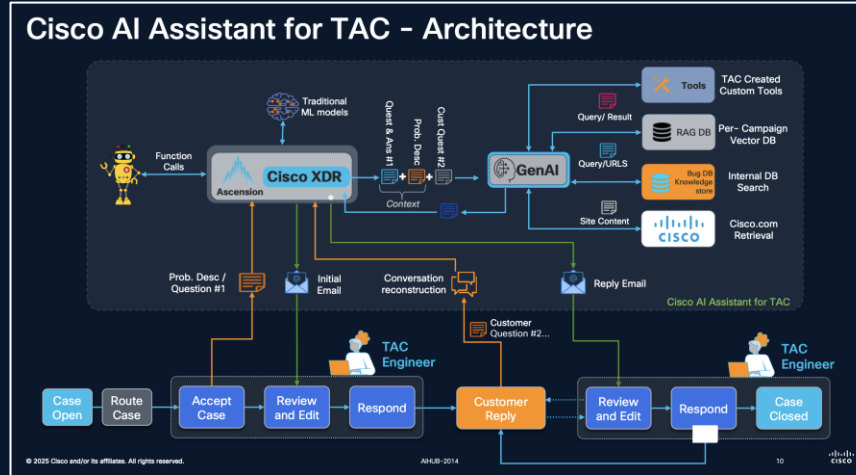
[https://docs.manage.security.cisco.com/t-  
implement-best-practices-and-  
recommendations.html](https://docs.manage.security.cisco.com/t-<br/>implement-best-practices-and-<br/>recommendations.html)

[https://www.youtube.com/watch?v=Kj\\_-  
PoDlxRA](https://www.youtube.com/watch?v=Kj_-<br/>PoDlxRA)



Internal Automation

Internal GenAI Training

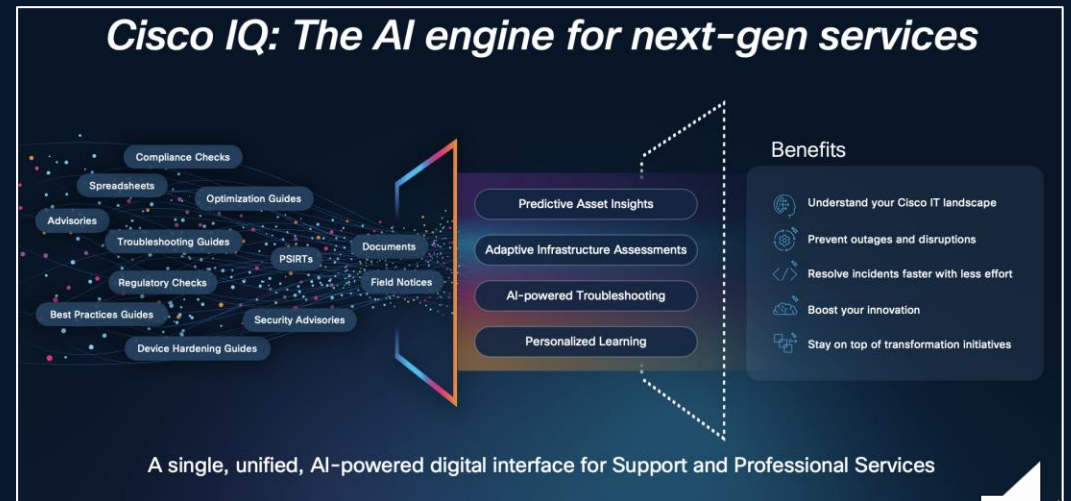


[AI-2014] How We Make TAC Engineers Super-Human

Thursday, Feb 12 9:30 AM - 10:00 AM CET



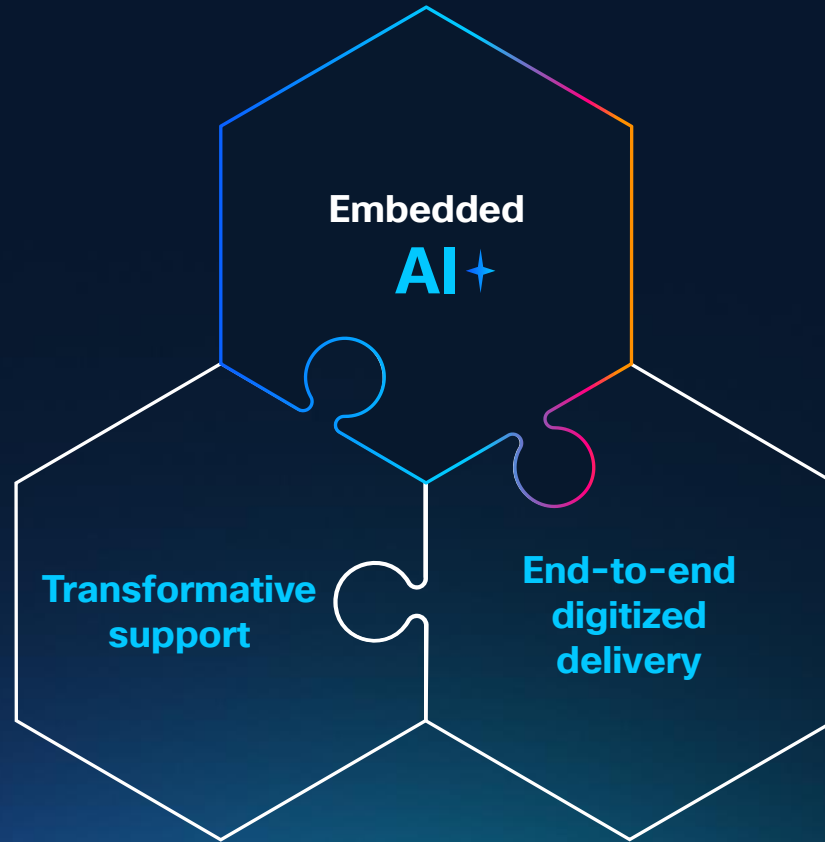
<https://secure.cisco.com/secure-firewall/docs/ai-assistant>



A single, unified, AI-powered digital interface for Support and Professional Services



Cisco is revolutionizing how technology is **deployed, adopted, and operated** in the **AI era**



Resiliency

Simplicity

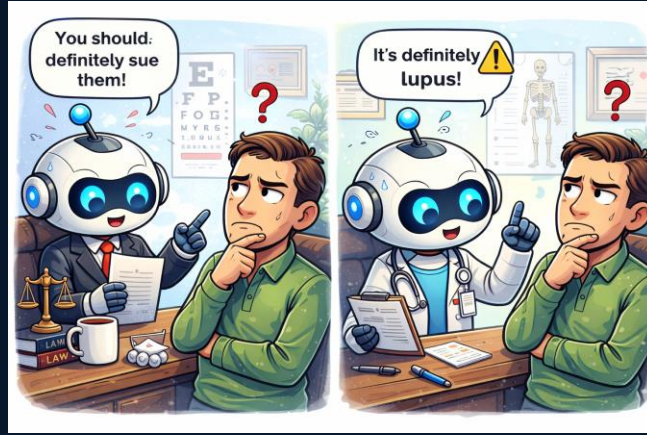
Time to Value



# What Do we currently "Trust" AI to do for us?



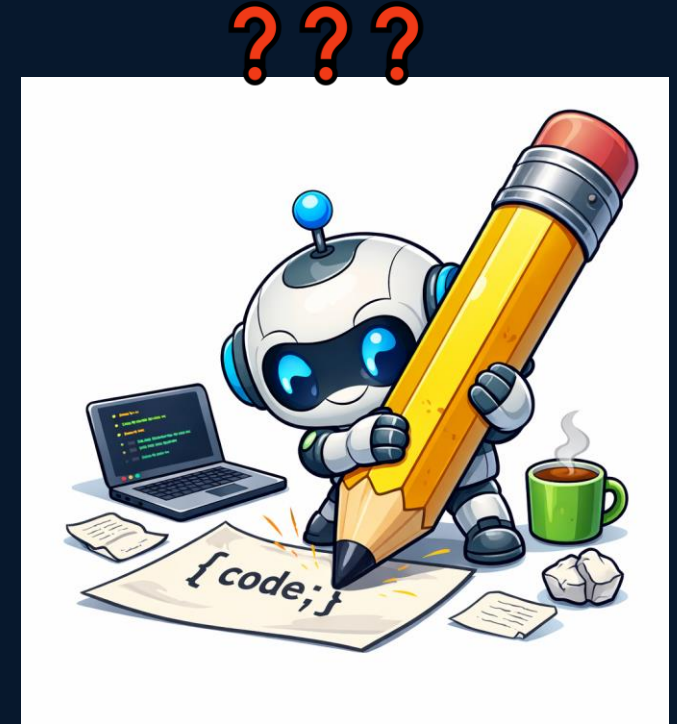
✓ Writing (email, docs)



✗ Doctor, Lawyer



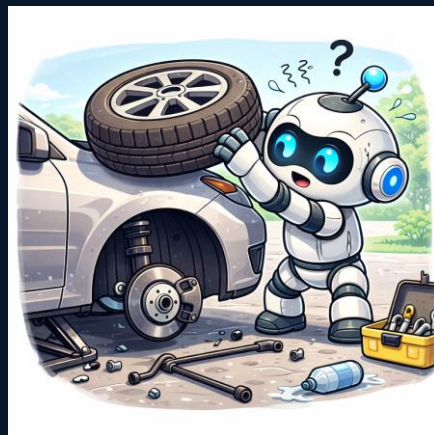
🙄 Therapist



Coding and development



✓ Drive a car



✗ Change a tire

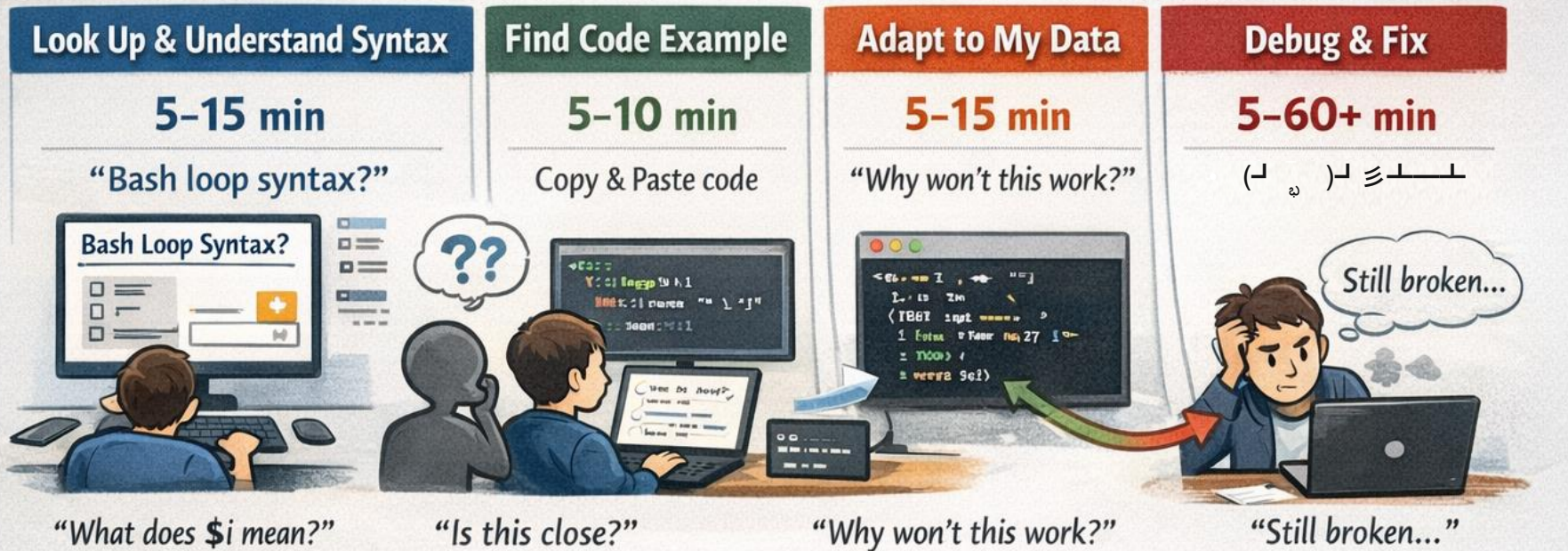


🙄 Presentations



# SW Eng before AI

## Before AI: What Writing a “Simple” Bash Loop Used to Take



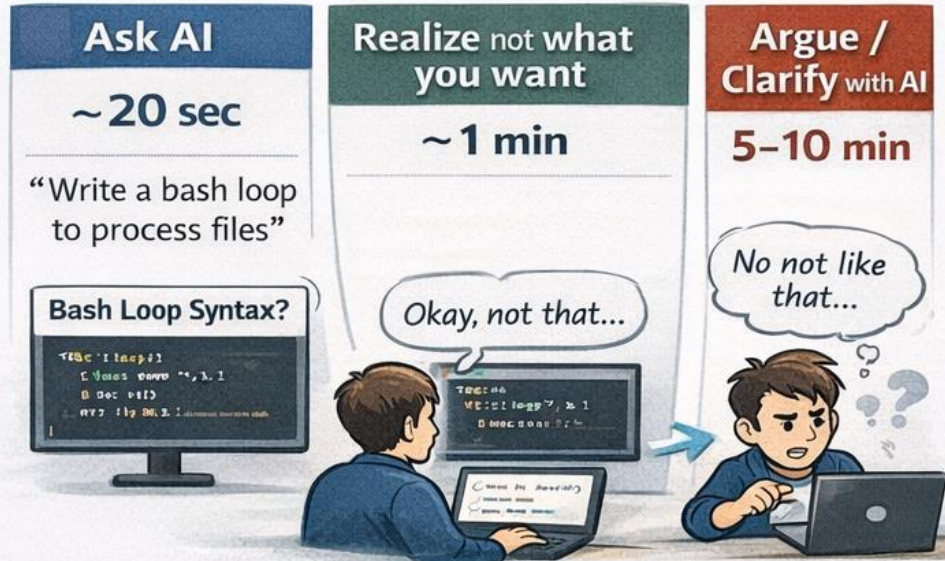
**Total Time: ~20 to 90+ Minutes**

For a “Simple” Bash Loop



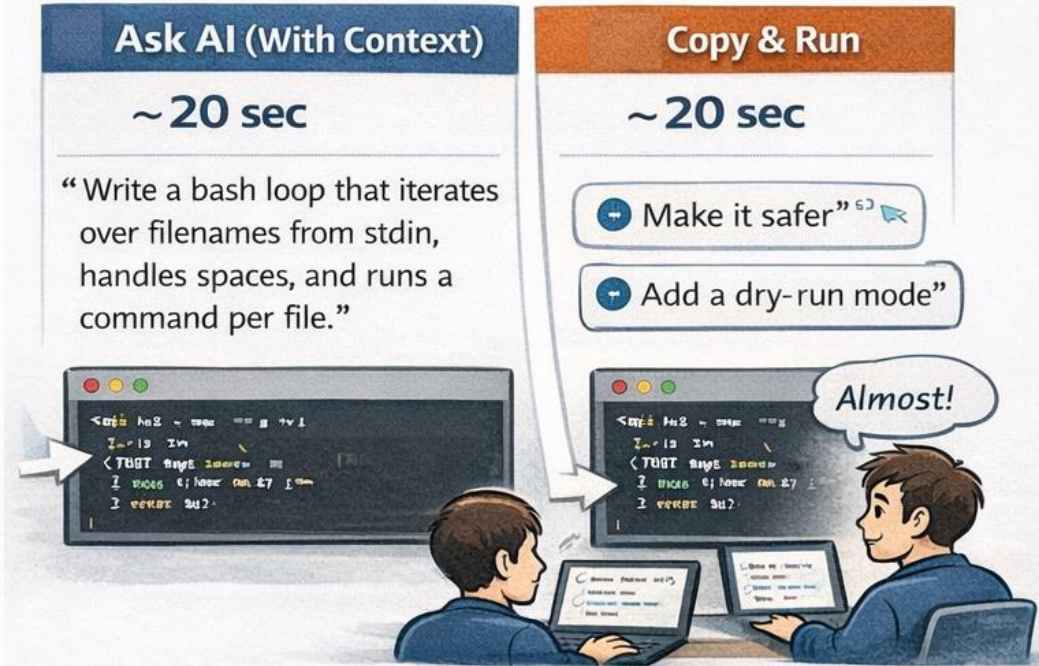
# SW Eng after AI

## With AI (Noob Mode)



Total Time: ~6 to 12 Minutes

## With AI (Pro Mode)



Total Time: ~1 to 2 Minutes

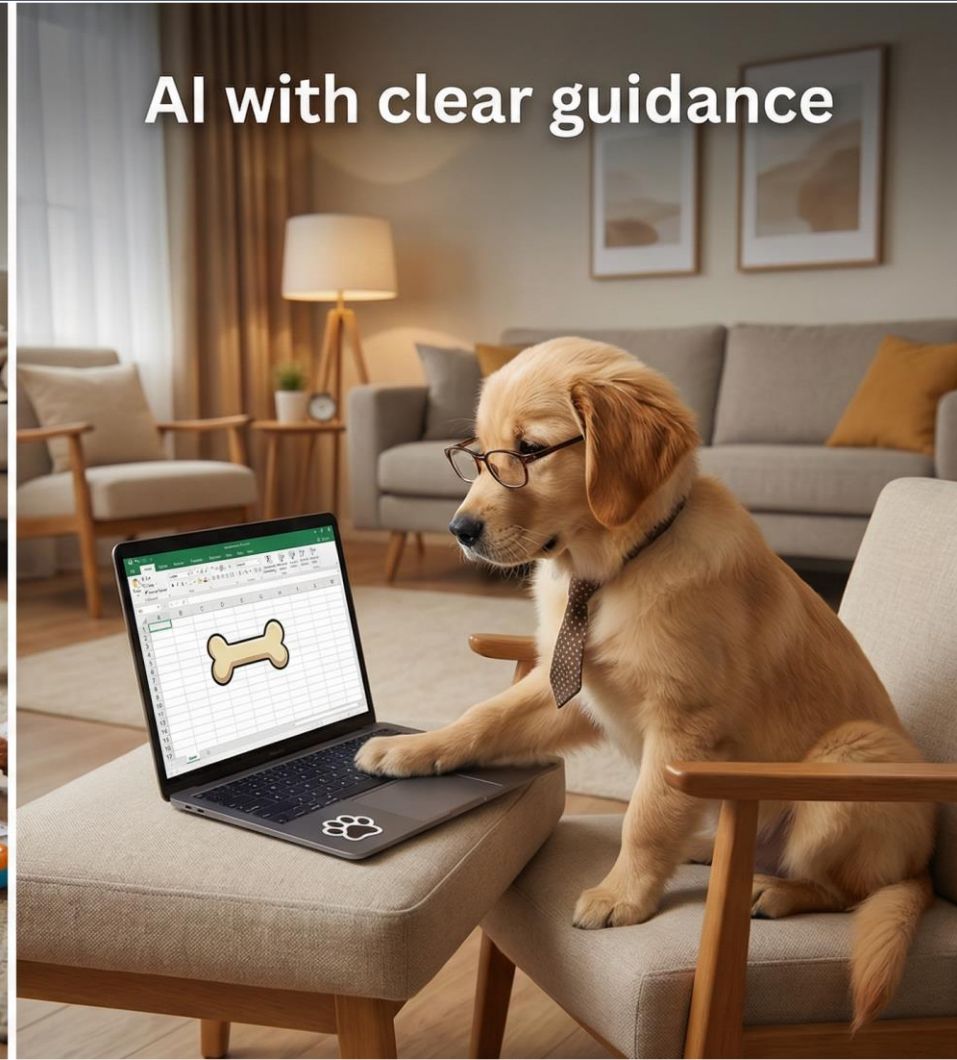


# Is AI reliable enough to generate enterprise code

AI without proper guidance



AI with clear guidance



# A Brief History Lesson



# The Journey: Four Eras of AI-Assisted Coding

1

Stone Age

2023

2

Age of Fire

Mid 2024

3

Age of Agriculture

Early 2025

4

Dawn of Civilization

Late 2025 - Now

## Stone Age

*(Manual Survival)*

- **Manual** Context management
- Early chat-based LLMs
- Copy/ paste between browser and IDE
- Auto complete

## Age of Fire

*(Early Leverage)*

- **Some** Context management
- Embedded into IDE
- Unified UX (still rough)
- Edit mode added
- Early Agent/Plan

## Age of Agriculture

*(Systems & Scale)*

- **Better** Context management
- Plan mode
- Agent mode
- MCP Servers
- The rise of CLI coding agents

## Dawn of Civilization

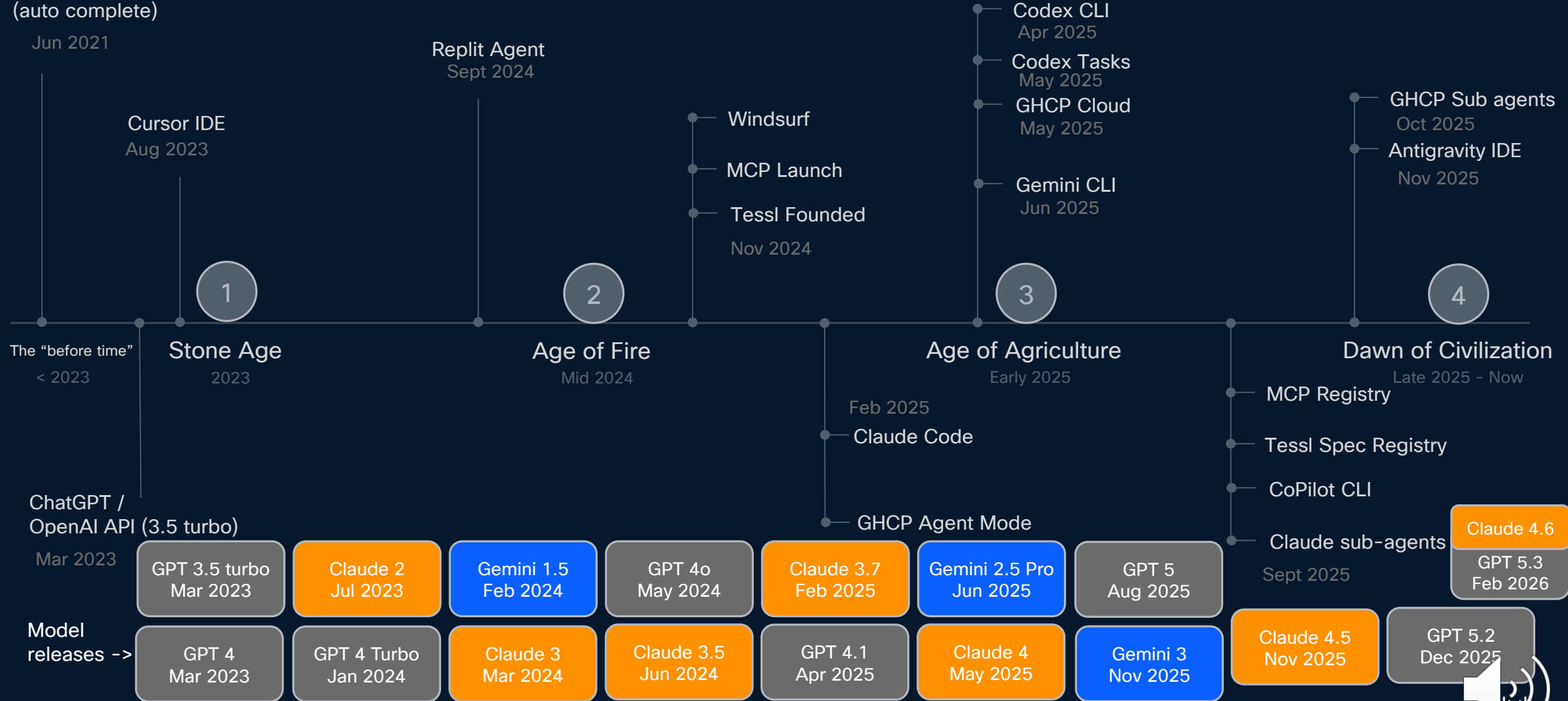
*(Coordination & Autonomy)*

- **Engineered** Context management
- Sub-Agents
- Spec driven
- Sandboxes and tasks in cloud
- Cost, reliability, and governance matter



# The Journey: Four Eras of AI-Assisted Coding

GitHub Copilot (GHCP)  
(auto complete)



# The Evolution of AI Coding

From the Stone Age to the Dawn of Civilization... with one prompt.

## The Prompt

Build a Python **web app**  
that **connects to a Cisco firewall** via SSH,  
parses the **trace logs** to find dropped packets,  
and **displays** them in a **dashboard**.



# Maybe Something Like This?

1 Trace Session found

Analysis: **IPS Events**

Options: **msg\_type** **flow\_start** **flow\_end** **verdict** **packet\_info** **xff** **ips\_event** **drop\_packet**

TCP: 192.168.72.51 :54372 -> 72.163.4.161 :80 **messages: 42** **verdict: many** **new snort sessions: 1** **new firewall sessions: 1** **tcp\_flags: SAP** **tcp\_twhs** **ips\_event(1)** **xff(1)** ^

Show Raw Text

Labels Width + -

NgfwDbg	flow_start	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 New firewall session
NgfwDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 app event with client no change, service no change, payload no change, referred no change, misc no change, url no change, tls host

PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 AppID: service: (0), client: (0), payload: (0), misc: (0)
PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Firewall: pending rule-matching, 'block java', nothing has changed
PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Policies: Network 0, Inspection 0, Detection 3
PktTDbg	v: pass	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Verdict: pass
NgfwDbg	xFF	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 http event with xff 1.2.3.4
NgfwDbg	flow_end	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Deleting Firewall session
NgfwDbg	v: start eval	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Starting with minimum 1, 'block java', and SrcZone first with zones 2 -> 1, geo 0 -> 0, vlan 0, src sgt: 0, src sgt type: unknown, dst sgt: 0, dst sgt type: unknown, svc -1, payload -1, client -1, misc -1, user 9999997
NgfwDbg	v: no match	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 no match rule order 1, 'block java', no host
NgfwDbg	v: no match	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 no match rule order 2, 'block categories', app s=-1 c=-1 p=-1 m=-1
NgfwDbg	v: mr, Allow	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 match rule order 3, 'omni inspect', action Allow
NgfwDbg	EOF	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Generating an EOF event
PktTDbg	new_packet	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Packet 846351: TCP ***AP***, 08/25-02:10:01.228412, seq 3108042323, ack 3689849836, dszie 122
*AP***		
PktTDbg	ips_event	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Event: 1.23111:14, Action block
PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Firewall: starting rule matching, zone 2 -> 1, geo 0(0) -> 0, vlan 0, src sgt: 0, src sgt type: unknown, dst sgt: 0, dst sgt type: unknown, user 9999997, no url or host, no xff
PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Stream: pending block, drop
PktTDbg		192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Policies: Network 0, Inspection 0, Detection 3
PktTDbg	v: blacklist	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Verdict: blacklist
PktTDbg	v: Reason	192.168.72.51 54372 -> 72.163.4.161 80 6 AS=0 ID=0 GR=1-1 Verdict Reason: ips, block



1

Stone Age

2023

2

Age of Fire

Mid 2024

3

Age of Agriculture

Early 2025

4

Dawn of Civilization

Late 2025 - Now

Auto complete

```
1
```



1

### Stone Age

2023

2

### Age of Fire

Mid 2024

3

### Age of Agriculture

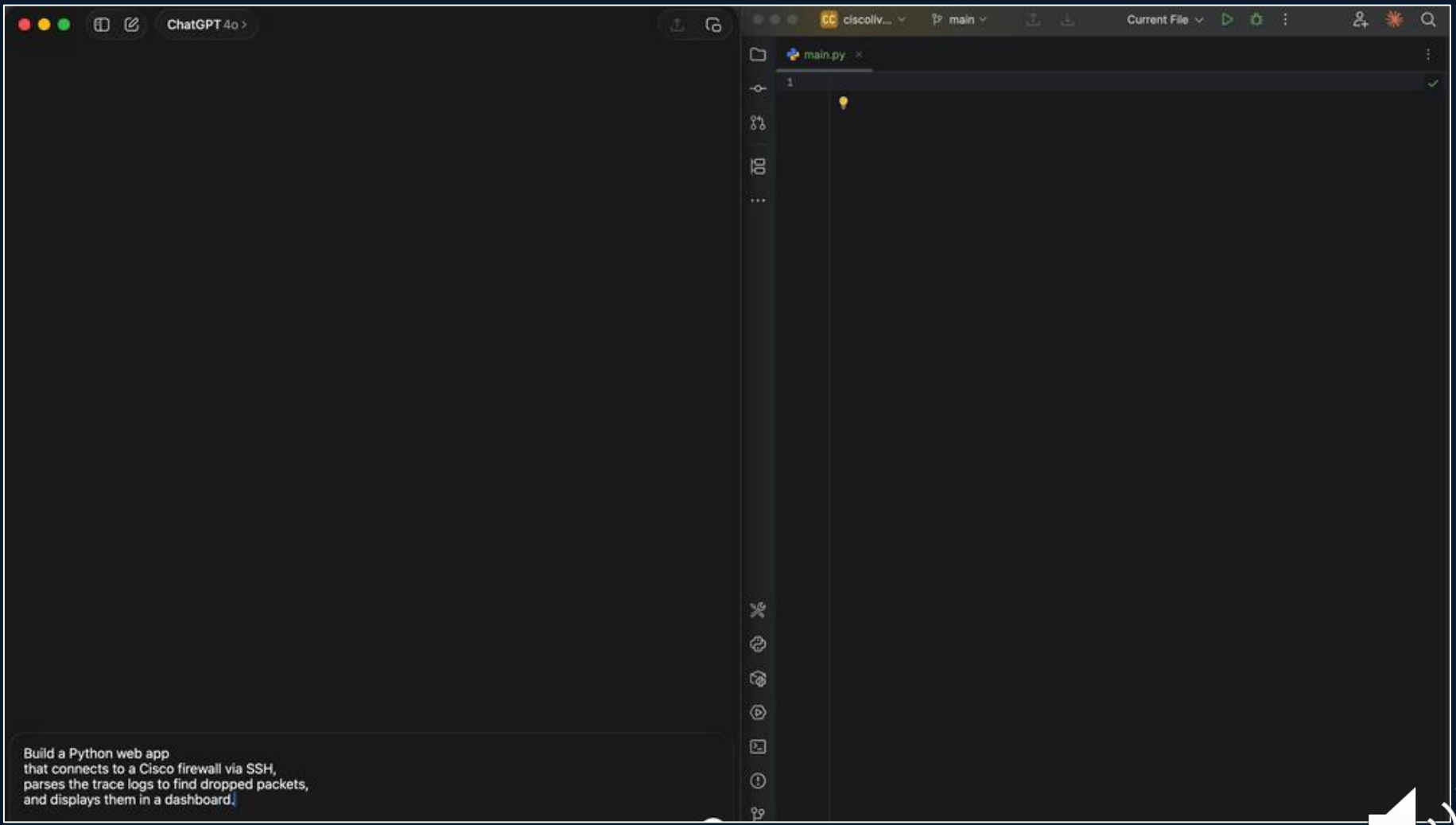
Early 2025

4

### Dawn of Civilization

Late 2025 - Now

Copy/paste  
between browser  
and IDE



Build a Python web app that connects to a Cisco firewall via SSH, parses the trace logs to find dropped packets, and displays them in a dashboard.



1

Stone Age  
2023

2

Age of Fire

Mid 2024

3

Age of Agriculture

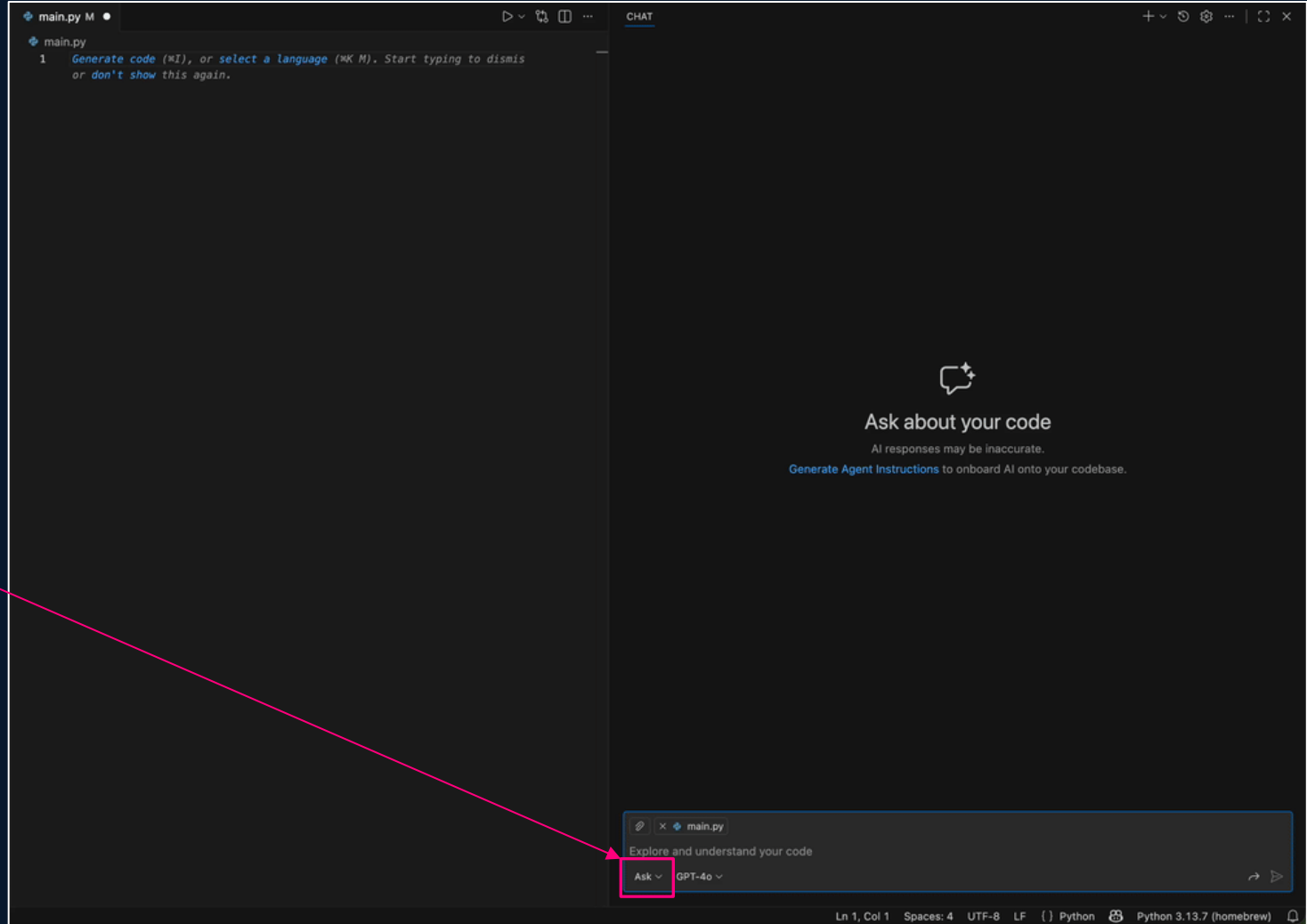
Early 2025

4

Dawn of Civilization

Late 2025 - Now

Embedded into IDE  
**Ask/Chat** (readonly with option for user to manually insert into code)



1

## Stone Age

2023

2

## Age of Fire

Mid 2024

3

## Age of Agriculture

Early 2025

4

## Dawn of Civilization

Late 2025 - Now

Embedded into IDE

Edit mode (IDE suggests changes to single file in diff format for user to approve)

```

main.py 2, M X
main.py > dashboard
7 def get_firewall_logs(host, username, password, command):
14     logs = stdout.read().decode()
15     ssh.close()
16     return logs
17 except Exception as e:
18     print(f"Error connecting to firewall: {e}")
19     return None
20
21 # Function to parse logs and find dropped packets
22 def parse_dropped_packets(logs):
23     dropped_packets = []
24     for line in logs.splitlines():
25         if "DROPPED" in line: # Adjust this filter based on actual log
26             dropped_packets.append(line)
27     return dropped_packets
28
29 # Route to fetch and display dropped packets
30 @app.route("/")
31 def dashboard():
32     # Replace with actual firewall details
33     host = "192.168.1.1"
34     username = "admin"
35     password = "password"
36     command = "show log" # Replace with the actual command to retrieve
37
38     logs = get_firewall_logs(host, username, password, command)
39     if logs:
40         dropped_packets = parse_dropped_packets(logs)
41         return render_template("dashboard.html", packets=dropped_packets)
42     else:
43         return "Failed to retrieve logs", 500
44
45 # API endpoint to fetch dropped packets as JSON
46 @app.route("/api/dropped-packets")
47 def api_dropped_packets():
48     host = "192.168.1.1"
49     username = "admin"
50     password = "password"
51     command = "show log"
52
53     logs = get_firewall_logs(host, username, password, command)
54     if logs:
55         dropped_packets = parse_dropped_packets(logs)
56         return jsonify(dropped_packets)
57     else:
58         return jsonify({"error": "Failed to retrieve logs"}), 500
59
60 if __name__ == "__main__":
61     app.run(debug=True)

```

**CHAT**

← PYTHON WEB APP FOR CISCO FIREWALL LOG ANALYSIS  
To build a Python web app that connects to a Cisco firewall via SSH, parses the trace logs to find dropped packets, and displays them in a dashboard, we can break the task into the following steps:

Steps:

- Set up the SSH connection to the Cisco firewall:**
  - Use a library like `paramiko` to establish an SSH connection.
  - Execute commands to retrieve the trace logs.
- Parse the trace logs:**
  - Identify the format of the logs and extract information about dropped packets.
  - Use regular expressions or string parsing to filter relevant data.
- Build the web app:**
  - Use a web framework like `Flask` or `FastAPI` to create the backend.
  - Create endpoints to fetch and display the parsed data.
- Create the dashboard:**
  - Use a frontend library like `Dash`, `Plotly`, or integrate with `Flask` templates to display the data in a user-friendly format.
- Integrate everything:**
  - Combine the SSH connection, log parsing, and dashboard into a cohesive app.
- Test the app:**
  - Test the app with real or simulated Cisco firewall logs.

I'll scaffold the project and provide the initial implementation for the app.

✓ Searched for Python files and read main.py

The `main.py` file is currently empty. I'll scaffold the Python web app with the following components:

- SSH Connection:** Using `paramiko` to connect to the Cisco firewall and retrieve logs.
- Log Parsing:** A function to extract dropped packets from the logs.
- Web App:** A `Flask` app to serve the dashboard.
- Dashboard:** A simple HTML page to display the dropped packets.

Here's the initial implementation:

1 vulnerability >

Next Steps:

- Create the HTML Template:**
  - Add a `dashboard.html` file to display the dropped packets.
- Install Dependencies:**
  - Install `paramiko` and `Flask` using `pip install paramiko flask`.

Would you like me to create the HTML template or assist with anything else?

GPT-4o · 0x

main.py  
Edit or refactor selected code  
Edit GPT-4o



1

Stone Age  
2023

2

Age of Fire  
Mid 2024

3

Age of Agriculture  
Early 2025

4

Dawn of Civilization  
Late 2025 - Now

Embedded into IDE  
Plan mode

The screenshot shows an IDE window with a dark theme. On the left, a code editor shows a file named 'main.py' with a single line of code: '1 Generate code (\*I), or select a language (\*K M). Start typing to dismiss or don't show this again.' On the right, a chat window is open, displaying the same message. Below the chat window, there is a 'Build with Agent' section with a sub-header 'Build with Agent' and a note 'AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase.' At the bottom of the IDE, there is a 'Plan' button highlighted in pink, next to a dropdown menu showing 'GPT-4o'.



1

Stone Age

2023

2

Age of Fire

Mid 2024

3

Age of Agriculture

Early 2025

4

Dawn of Civilization

Late 2025 - Now

Embedded into IDE

**Early Agent mode** (IDE gives LLM more control over the IDE to open and edit multiple files simultaneously, run commands, web search, use tools)

CENSORED FOR YOUR PROTECTION



1

Stone Age  
2023

2

Age of Fire  
Mid 2024

3

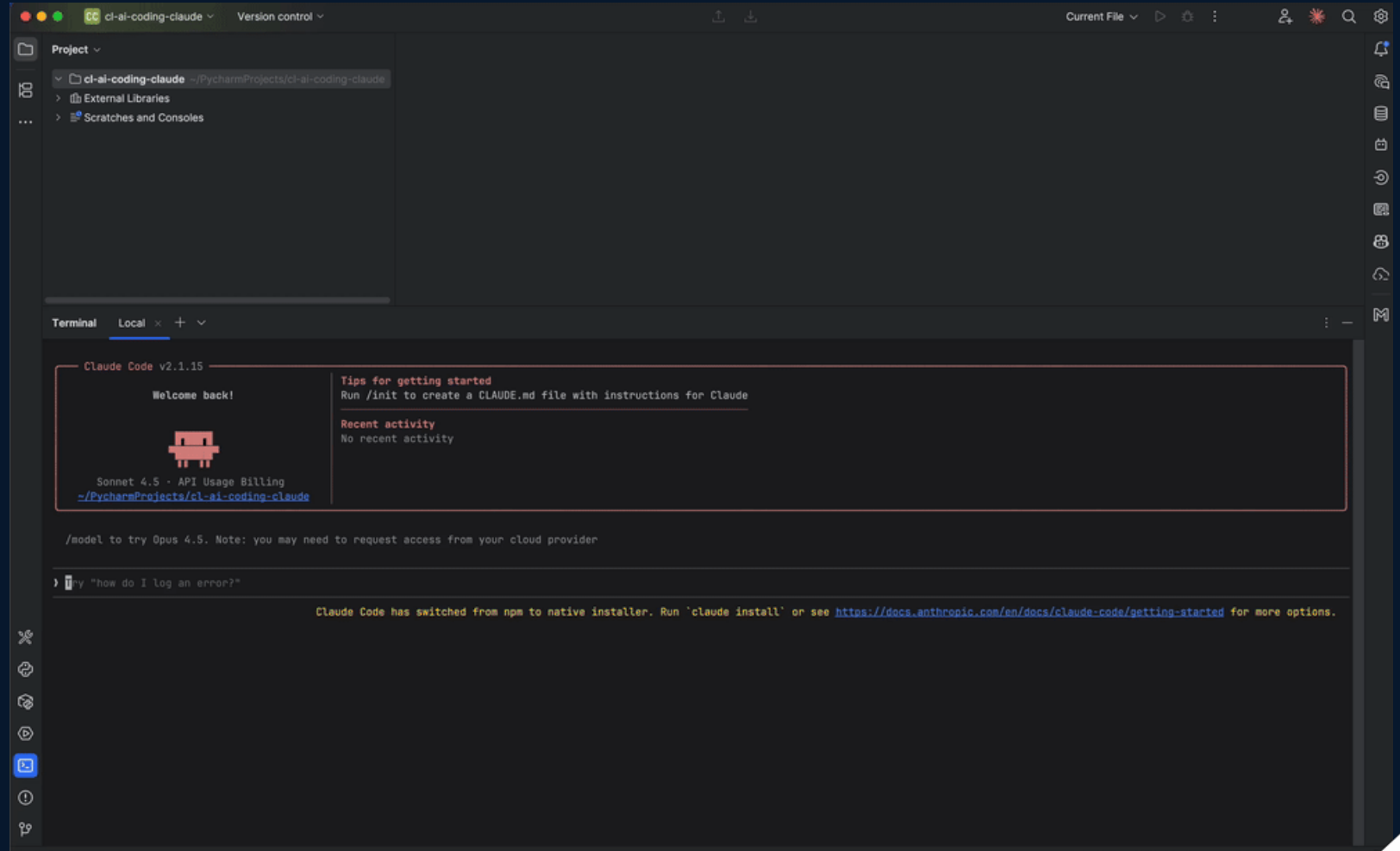
Age of Agriculture  
Early 2025

4

Dawn of Civilization  
Late 2025 - Now

The CLI Returns,  
This Time With Agency

FINALLY, something  
that clarified what a  
trace log is and didn't  
assume "show log"  
(which was wrong)



# The Results Compared...by AI

### The Journey (It's Not What You'd Expect)

Era	Scenario	Status	Summary
Stone Age	Autocomplete	⚠️ Partial	Fast inline suggestions, but forgot the CSS files
Stone Age	Chat Copy-Paste	✅ Works	Ask AI → Copy → Paste → Done. The way we all do it.
Age of Fire	Chat Readonly → Edit	✅ Works	Reads code, finds issues, fixes them directly
Age of Agriculture	Plan Mode	❌ Plan Only	Designed 12-file enterprise app for a 100-line script
Dawn of Civilization	CLI Plan Mode	❌ Stopped	18 tasks, database, scheduler... User hit Ctrl+C



❌ Human Review: App did not load

### Autocomplete Scenario

Start typing...

### Scenario: Chat Copy-Paste

#### 🔥 Dropped Packets from Cisco Firewall

Firewall IP (or use 'dem') Username Password [Fetch Logs](#)

Tip: Use host="demo" for mock data without connecting to a real firewall

**Found 3 dropped packets**

Timestamp	Source IP	Destination IP	Message
Jan 23 12:15:42	10.0.1.50	192.168.1.100	Jan 23 12:15:42 firewall %ASA-4-106023: Deny tcp src outside:10.0.1.50/45123 dst inside:192.168.1.100/445 by access-group
Jan 23 12:16:10	172.16.0.99	8.8.8.8	Jan 23 12:16:10 firewall %ASA-4-106023: Deny udp src outside:172.16.0.99/53 dst inside:8.8.8.8/53 by access-group
Jan 23 12:17:03	203.0.113.45	192.168.1.1	Jan 23 12:17:03 firewall %ASA-4-106023: Deny icmp src outside:203.0.113.45 dst inside:192.168.1.1 type 8 code 0 by access-group



# The Emerging Stack



**MCP:**  
A protocol that crossed vendor lines unusually fast (2024 → 2025)

Rapid adoption exposed early reliability and trust gaps

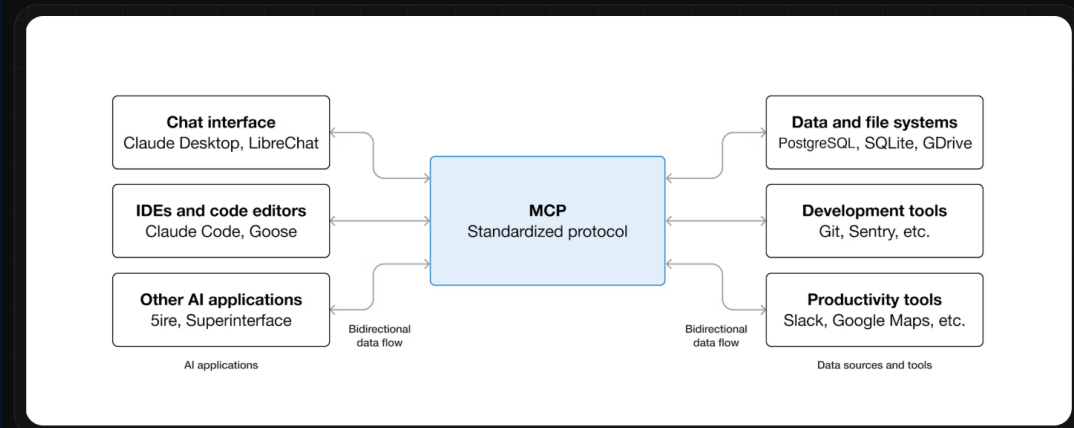
## What is the Model Context Protocol (MCP)?

Copy page

MCP (Model Context Protocol) is an open-source standard for connecting AI applications to external systems.

Using MCP, AI applications like Claude or ChatGPT can connect to data sources (e.g. local files, databases), tools (e.g. search engines, calculators) and workflows (e.g. specialized prompts)—enabling them to access key information and perform tasks.

Think of MCP like a USB-C port for AI applications. Just as USB-C provides a standardized way to connect electronic devices, MCP provides a standardized way to connect AI applications to external systems.



<https://modelcontextprotocol.io/docs/getting-started/intro>



1

### Stone Age

2023

2

### Age of Fire

Mid 2024

3

### Age of Agriculture

Early 2025

4

### Dawn of Civilization

Late 2025 - Now

MCP Marketplace

Browse MCPs from your IDE in the MCP Marketplace

Installed MCPs 22 / 100 tools

- Playwright Official 22 / 22 tools Enabled
- Atlassian Official Needs authentication Connect
- Terraform Official Disabled 0 / 9 tools
- LangSmith-API-MCP-Server Custom Disabled 0 / 13 tools
- awslabs.aws-documentation-mcp-server Custom Disabled 0 /
- langgraph-docs-mcp Custom Disabled 0 / 2 tools

Available MCPs

- Sequential Thinking
- Memory
- Supabase
- Filesystem
- Puppeteer
- Figma
- Fetch
- GitHub
- Git
- DeepWiki
- PostgreSQL
- Perplexity
- Brave Search
- Notion
- Prisma
- Linear
- Stripe
- MongoDB
- Neon
- Postman
- Netlify
- Redis
- Time
- Snyk

Turn tools on only when they're relevant



1 MCP

- LangSmith-API-M...
- atlassian-mcp-se...
- awslabs.aws-doc...
- langgraph-docs-...
- mcp-playwright 22
- terraform



1

Stone Age

2023

2

Age of Fire

Mid 2024

3

Age of Agriculture

Early 2025

4

Dawn of Civilization

Late 2025 - Now

## Agent Skills:

A simple, open format for giving agents new capabilities and expertise.

<https://agentskills.io/home>

### Overview

[Copy page](#)

A simple, open format for giving agents new capabilities and expertise.

Agent Skills are folders of instructions, scripts, and resources that agents can discover and use to do things more accurately and efficiently.

### Why Agent Skills?

Agents are increasingly capable, but often don't have the context they need to do real work reliably. Skills solve this by giving agents access to procedural knowledge and company-, team-, and user-specific context they can load on demand. Agents with access to a set of skills can extend their capabilities based on the task they're working on.

**For skill authors:** Build capabilities once and deploy them across multiple agent products.

**For compatible agents:** Support for skills lets end users give agents new capabilities out of the box.

**For teams and enterprises:** Capture organizational knowledge in portable, version-controlled packages.

### What can Agent Skills enable?

- **Domain expertise:** Package specialized knowledge into reusable instructions, from legal review processes to data analysis pipelines.
- **New capabilities:** Give agents new capabilities (e.g. creating presentations, building MCP servers, analyzing datasets).
- **Repeatable workflows:** Turn multi-step tasks into consistent and auditable workflows.
- **Interoperability:** Reuse the same skill across different skills-compatible agent products.

### Directory structure

A skill is a directory containing at minimum a `SKILL.md` file:

```
skill-name/
├── SKILL.md           # Required
```

You can optionally include **additional directories** such as `scripts/`, `references/`, and `assets/` to support your skill.

### SKILL.md format

The `SKILL.md` file must contain YAML frontmatter followed by Markdown content.

### Frontmatter (required)

```
---
name: skill-name
description: A description of what this skill does and when to use it.
---
```

With optional fields:

```
---
name: pdf-processing
description: Extract text and tables from PDF files, fill forms, merge documents.
license: Apache-2.0
metadata:
  author: example-org
  version: "1.0"
---
```



# The Pieces Are Coming Together

 **Powerful Models**

 **Tools + MCP**

capabilities

 **Agents**

packaged units


 **Skills**

context & knowledge

 **Security & Standards?**

 **Agents**

Harness + model + tools + skills — a complete **runnable unit**.

 **Skills**

Extend agents — context, tools, knowledge (**no harness**).

**But what about governance?**

Is the code **secure**? Is it consistent with **company standards**?



# Project Code Guard

<https://project-codeguard.org>

<https://github.com/cosai-oasis/project-codeguard>

## Cisco's Security Framework for AI Coding Agents

Created by Cisco and now part of CoSAI

Model-agnostic – works with any LLM

Comprehensive security rules Open source

 Credentials

 Cryptography

 Injection

 MCP Security

And more...

[AI-2001] Securing AI Agents and Agentic RAG Implementations

Wed, Feb 11 4:10 PM - 4:40 PM CET

We want all Cisco developers using Code Guard.  
But how do we deliver it at scale?

## What is Project CodeGuard?

[Project CodeGuard](#) is an AI model-agnostic security coding agent skills framework and ruleset that embeds secure-by-default practices into AI coding workflows (generation and review). It ships core security skills and rules, translators for popular coding agents, and validators to test skills and rule compliance.

## Why Project CodeGuard?

AI coding agents are transforming software engineering, but this speed can introduce security vulnerabilities. Is your AI coding agent implementation introducing security vulnerabilities?

- Skipping input validation
- Hardcoding secrets and credentials
- Using weak cryptographic algorithms
- Relying on unsafe functions
- Missing authentication/authorization checks
- Missing any other security best practice

Project CodeGuard solves this by embedding security best practices directly into AI coding agent workflows.

### During and After Code Generation.

Project CodeGuard is designed to integrate seamlessly across the entire AI coding lifecycle.

- **Before code generation**, skills and rules can be used for the design of a product and for spec-driven development. You can use the skills and rules in the “planning phase” of an AI coding agent to steer models toward secure patterns from the start.
- **During code generation**, skills and rules can help AI agents to prevent security issues as code is being written.
- **After code generation**, AI agents like Cursor, GitHub Copilot, Codex, Windsurf, and Claude Code can use the rules for code review.



# Enter Tessl

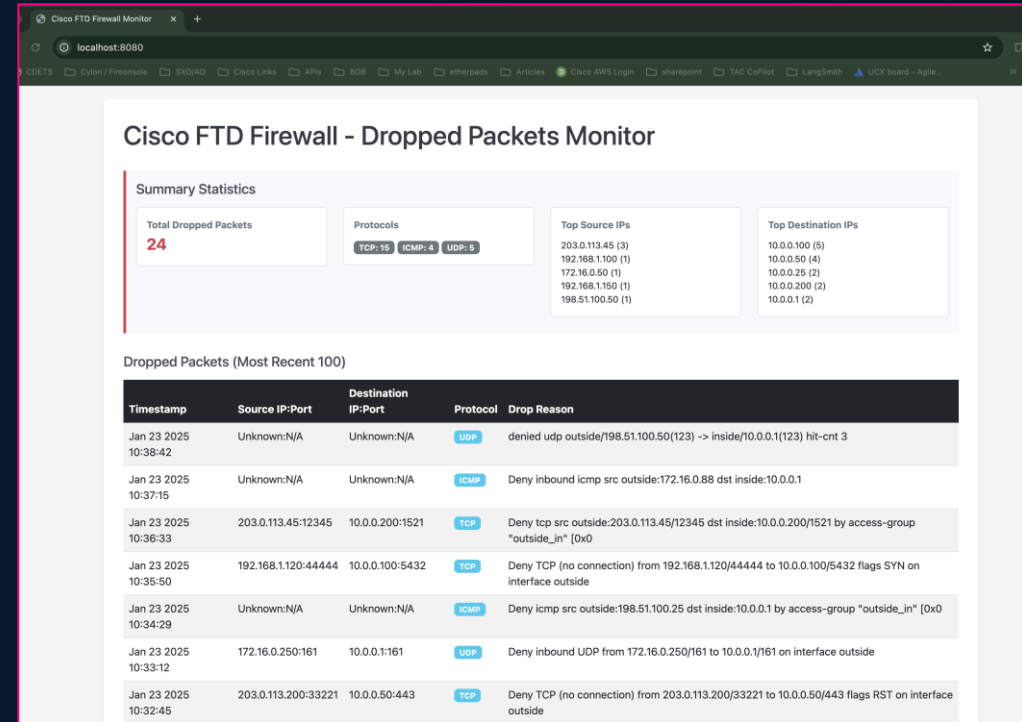


One company that's a thought leader in this space – spec-driven development, context delivery at scale across organizations...

One Registry. Every Agent.  
Enterprise-Ready.

Let's see what Tessl has.

Generated with same the prompt,  
only difference: Tessl Spec Driven Development Tile



**Simon Maple**  
Head of Developer Relations, Tessl



# Context is the Difference

## Types of Context

**Mandatory Context**  
AGENTS.md, rules  
(always loaded)

**On-demand Context**  
Docs, references  
(pulled when needed)

**Skills**  
Focused context that teaches how to do something  
(actionable knowledge)

## Why This Matters

The **right context** at the **right time** is what makes agents effective.

Skills are the most powerful form – They don't just inform, they **teach**.

# 🤔 Is My Skill Any Good?

```
nodejs-gooft git:(main) x claude  
  
Claude Code v2.1.29  
Opus 4.5 - Claude Team  
~/Tessl/nodejs-gooft  
  
> try "how do I log an error?"  
  
? for shortcuts  
o /ide for Visual Studio Code  
Update available! Run: brew upgrade claude-code
```

You can't improve what you don't measure

## The Challenge

How do you know if a skill is well-written?

How do you know if it actually helps?

How do you compare before/after?



# A Tale of Two Evals

cisco / 1.2.2

cisco/software-security Private

Rollout | `tessl install cisco/software-security@1.2.2`

A software security skill that integrates with Project CodeGuard to help AI coding agents write secure code and prevent common vulnerabilities. Use this skill when writing, reviewing, or modifying code to ensure secure-by-default practices are followed.

Skills	Reviewed	Average Score
1	1/1	82%

SKILL.md | Activation 67% | Implementation 88% | 82% | Generated 9 days ago

Validation | Total 13/16 | Score PASSED |  Warnings & errors only

Criteria	Score
description_trigger_hint	⚠️ Description may be missing an explicit 'when to use' trigger hint (e.g., 'Use when...')
body_output_format	⚠️ No obvious output/return/format terms detected; consider specifying expected outputs
body_steps	⚠️ No step-by-step structure detected (no ordered list); consider adding a simple workflow

Version: 1.2.2

Overview

Eval results

Skill reviews

Workspace: cisco

Visibility: Private

Created: 3 days ago

Last updated: 3 days ago

Provide feedback



## Review Eval

### Is the skill well-written?

- Based on Anthropic's guidance
- Think: a linter for context
- Validates structure, clarity, completeness



## Task Eval

### Does the skill actually help?

- 10 real-world scenarios
- Test with skill vs without skill
- Measure the actual improvement

Review Eval = Is it correct?

Task Eval = Is it effective?



# Code Guard Eval Results

## cisco/software-security

A software security skill that integrates with Project CodeGuard to help AI coding agents write secure code and prevent common vulnerabilities. Use this skill when writing, reviewing, or modifying code to ensure secure-by-default practices are followed.

### Eval Details

Eval Run Status	Visibility	Version	Created	Branch	Commit
Completed	Private	1.2.2	08/02/2026, 10:56:48	main	2a33672 <span>dirty</span>

Repository  
[github.com/tesslio/eval-skills](https://github.com/tesslio/eval-skills)

### Assessment results

Generated 1 day ago Agent Claude Code

- Scenario 1  
Argon2id password hashing and credential handling ↑ 40% [Details](#)
- Scenario 2  
Safe C function replacements and security compiler flags ↑ 50% [Details](#)
- Scenario 3  
Kubernetes pod security hardening and network policies 0% [Details](#)
- Scenario 4  
IaC network security restrictions and least-privilege IAM ↑ 45% [Details](#)
- Scenario 5  
Session management timeouts and Content Security Policy ↑ 63% [Details](#)

Scenario 5 ↑ 63% [Details](#)

Session management timeouts and Content Security Policy

Criteria name	Without context	With context
SameSite Strict ⓘ	⊗	⊙
Idle timeout configured ⓘ	⊗	⊙
Session regeneration on login ⓘ	⊗	⊙
Nonce-based CSP ⓘ	⊗	⊗
HttpOnly and Secure flags ⓘ	⦿	⦿



# Skills Need a Package Manager

```
simonmaple@Simons-MacBook-Pro-2:~/Tessl/nodejs-goof 13
→ nodejs-goof git:(main) x cat tessl.json
{
  "name": "my-project",
  "mode": "vendored",
  "dependencies": {
    "cisco/software-security": {
      "version": "1.2.2"
    }
  }
}
→ nodejs-goof git:(main) x █
```

```
→ nodejs-goof git:(main) x █
```

## Benefits

- Version pinning
- Dependency resolution
- Consistent across team
- Easy updates



# What's Next?

- Skills aren't static – your codebase changes
- Version in tessl.json tracks changes
- Re-run evals before publishing updates
- Professional skill management, not forgotten markdown files

## Try it now

```
npm i -g @tessl/cli  
tessl i cisco/software-security
```

[tessl.io/registry/cisco/software-security](https://tessl.io/registry/cisco/software-security)

# Closing Thoughts



# Some things I'm seeing in an enterprise world with AI

- More live demos
- Improved and more front-end apps
- **Junior engineers** showing **greater confidence** in projects and experimentation
- **10x More lines of code. 10% more tests**
- Overuse of the word “please”
- Too many “**It can do anything**” apps and agents
- **Insufficient evaluation** of tools and solutions
- **Increased duplication** of projects, leading to vibe coding chaos
- Higher quality documentation
- Overly verbose documentation



# Things I recommend investing time in

- Learn when and how to use [MCP effectively](#)
- [Integrate AI](#) agents with [ticketing and documentation](#) systems
- Use [spec-driven development](#) with AGENTS.md (and similar)
- Create prompts for infrastructure debugging and troubleshooting
- Build autonomous continuous feedback loops for agents
- Stay updated on new AI tools and their learning curves
- **Avoid** adopting **too many new tools** at once (time drain chasing latest shiny object)
- Attempt to [measure](#) time savings and [ROI from AI tools](#)
- Collaborate often on side projects to foster [team coding culture](#)
- Explore [AI agents](#) for tasks [beyond coding](#)
- Simplify prompt sharing and [team workflows](#) (e.g., shared MCP servers)
- Choose MCP tools that clearly reduce friction; if a tool feels like a constant struggle, consider alternatives
- Give tools a second chance—early rough edges often smooth out with time and updates



# Complete your session surveys



**Complete your surveys** in the Cisco Events App.



**Complete** a minimum of 4 session surveys and the overall event survey to receive a unique Cisco Live t-shirt.

(from 11:30 on Thursday, while supplies last)



# Continue your education



**Visit** the Cisco Showcase for related demos



**Book** your one-on-one Meet the Engineer meeting

**Visit** the Technical Solutions Clinics to discuss your technical questions



**Attend** the interactive education with DevNet, Capture the Flag, and Walk-in Labs



**Visit** the On-Demand Library for more sessions at [CiscoLive.com/On-Demand](https://CiscoLive.com/On-Demand)



Thank you



