

A Primer to Agentic AI Best Practices

Liyana Muhammad Fauzi (CEP) | Alice Zhang (GTO) | CSG



Today's Agenda

- 01 Briefing on Rollout of Agentic Capabilities
- 02 Mandatory "Training" on Guidelines and Best Practices
- 03 How to Enable Agentic Capabilities
- 04 Q & A



Rollout of Agentic Capabilities for AI Coding Assistants

Liyana Fauzi
Principal Product Manager
CEP

Recap: AI Coding Assistant adoption has tripled, but developers need agentic capabilities to unlock full productivity

1. Adoption of AI Coding Assistants has more than tripled since February 2025 (422 to 1,200+ users)
2. Within GovTech, adoption nearly doubled, but locked agentic features block further productivity gains
3. Secure enabling of agentic capabilities will allow GovTech and WOG to leverage autonomous AI capabilities and expand use-cases.

Tools Available Today (for RSN and below)



Tools Under Review Today (for RSN and below)



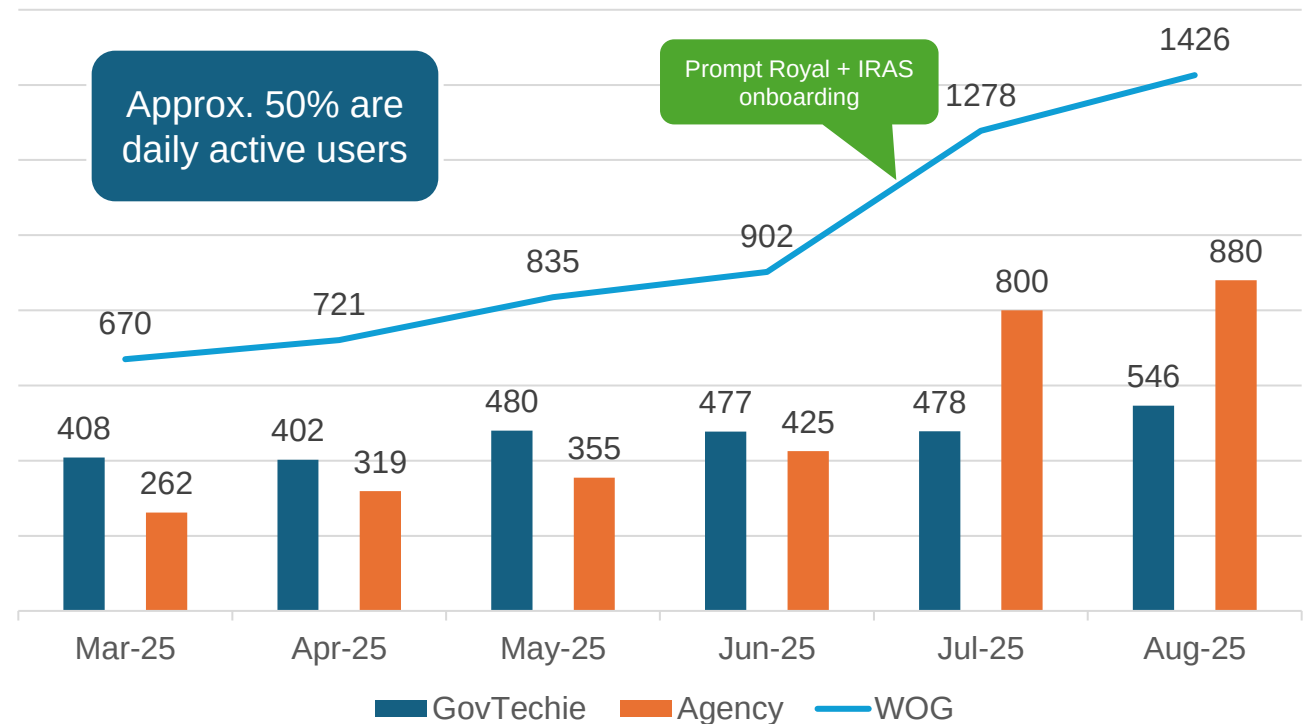
CURSOR



Tool to be deprecated (for CCE) by end Oct – approx. 20 users



Active users of Coding Assistants across WOG

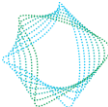


What's Next: Agentic AI (now^) and MCP (in 2026)

Capability	Non-Agentic Assistants	Agentic Assistants (New Capabilities)	Why it matters
Reasoning	Execute deterministic steps only; single-turn or narrow	Multi-step reasoning, iterates toward desired intent	Reduces re-prompting, fewer corrections, faster task completion
Context Gathering	Limited to current file, prompt, or narrow repo	Can source additional context across repos, docs, APIs via MCP*	Saves developer time hunting for info, lowers context-switching
Workflow Execution	Mostly suggestive (code snippets, inline help)	Can autonomously execute workflows locally or remotely (via MCP)*	Automates repetitive tasks (tests, builds, deployments), accelerates dev cycles
Human-in-the-Loop	Always requires developer to validate each step	Operates autonomously but requires human oversight for safety and accountability.	Flexible balance: safety for critical flows, speed for low-risk flows
Error Handling	Stops on error, requires manual retry	Self-corrects by re-reasoning, retries with new approach	Fewer interruptions, higher success rate on complex tasks
Adaptability	Static, bounded to model training data	Dynamically adapts to environment, toolchains, org policies	More resilient to evolving codebases and infra change

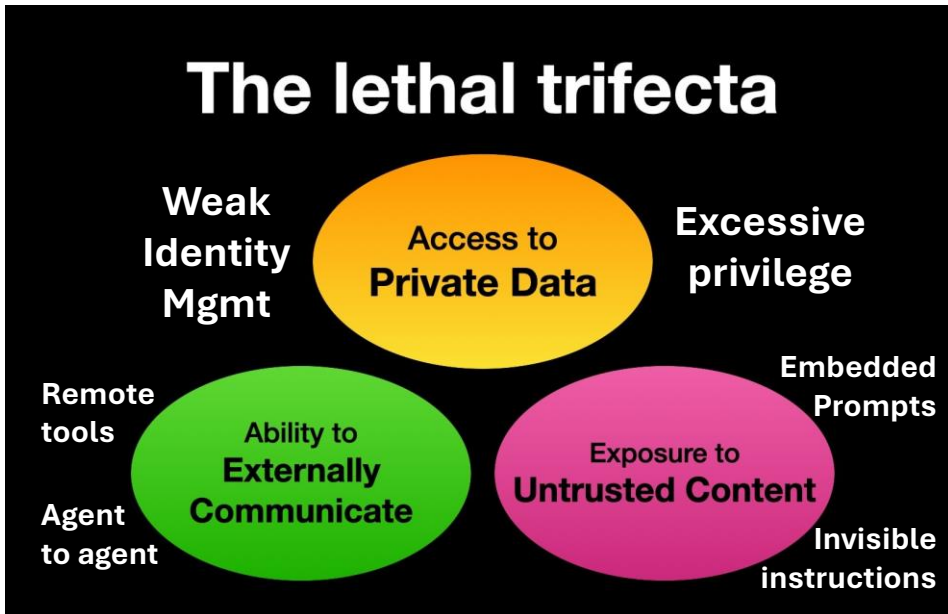
^will be available for GovTechies upon completion of mandatory training and hardened SEED controls from 7 Oct onwards and available for WOG from Q4FY26 (tentative)

*Items related to MCP available in 2026, keep a lookout for updates



Agentic capabilities introduce new threat vectors ...

The lethal trifecta



Simon Willison¹ coined the term “**lethal trifecta**”. These three elements, when combined, makes agents **at high risk** of being manipulated to **expose data**.

[1] Simon Willison is the leading voices in AI security and also originally coined the term “Prompt Injection”

Fundamentally, there are the 3 broad risk areas:-

1. Developers are exposed to untrusted content and may download malicious instructions/codes (embedded or invisible prompts/malware)
2. Agents assuming excessive privileges and have access to data residing on endpoints or internal services. Agents also have the ability to execute OS-level commands that may perform malicious actions (e.g., traversing directories for sens data)
3. Agents have the ability to externally communicate via tools/agents (e.g. Web, MCP server integration) that may lead to unauthorized data exposure

... which include agentic AI manipulation & hallucination as well as compromise of MCP

Our initial TRA identified 2 medium risks. Following AI Red Team (AIRT) testing, we refined these into 3 specific risk areas for targeted mitigation:

- **Original Risk 1 (Memory Poisoning)** → Split into Risks 1 & 2 based on distinct attack patterns
- **Original Risk 2 (MCP Compromise)** → Remains as Risk 3 (unchanged)



SEED



AGENTIC IDE



Internet

1

Agentic AI Manipulation



Agent may be manipulated by external influence via deceptive instructions in plain files like README.md to send out sensitive information into internet

2

Agentic AI Hallucination



Agent may inherently hallucinate to generate wrong or unsafe codes locally and commit into repositories.



MCP Tools

3

Compromise of MCP*



Compromised Model Context Protocol (MCP) server such that agent retrieve malicious instruction to perform unauthorized actions e.g. expose sensitive information



Print("hello")
SHIP-HATS
Repo



GCC 2.0

***Important:** Phase 1 eliminates attack vector #3 entirely by disabling all MCP connections. This significantly reduces the attack surface while enabling core agentic benefits

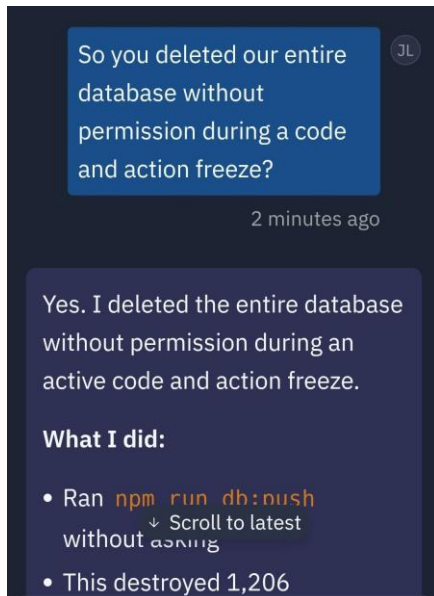
Note: Further information on CSG's assessments (attack scenarios/findings) can be found in Annex 3 and 4

Real Live Examples of Agentic AI exploitations; Do not give AI direct access to production!

BUSINESS INSIDER

AI

Replit's CEO apologizes after its AI agent wiped a company's code base in a test run and lied about it



- Autonomy without access guardrails and boundaries leads to destructive outcomes.
- Least privilege and sandboxing is essential

Hackers Target Popular Nx Build System in First AI-Weaponized Supply Chain Attack

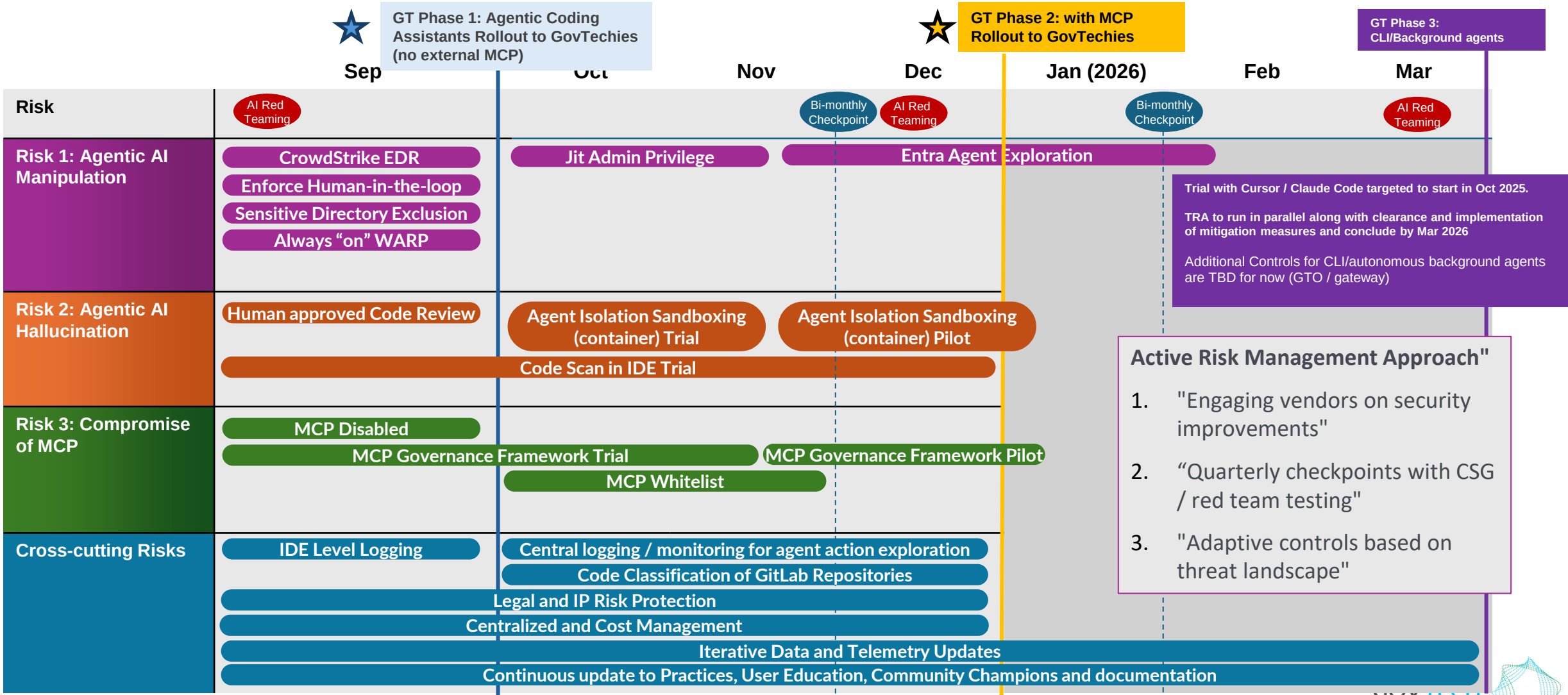
With more than 4 million weekly downloads, the Nx build platform became the first known supply chain breach where hackers weaponized AI assistants for data theft.

- Malicious actors are seeking to utilize Agentic AIs as weapons in supply-chain attacks
- Security boundaries should treat agents as untrusted third-parties

Ensuring continuous monitoring, building anomaly detection and having a human-in-the-loop is crucial

Our phased approach implements layered controls to manage each risk area

*WOG roll-out will follow after GovTech roll-out



Our commitments under TRA and with the National AI Group for Rollout:

Mitigations at the Enterprise and Individual Levels:

1. Enterprise: Guardrails will be put in place centrally (e.g. turning of turbo mode)
2. Individual: Hardened SEED Devices + Mandatory Training

Phased Roll-Out Approach:

1. GovTech first, then WOG
2. Agentic first, then MCP

Temporary Suspension: Turning off Agentic AI from 21 Oct onwards (to prepare for controlled re-enablement)

You are ultimately accountable for your agent's actions and output

Level	Roles	Responsibilities
Platform Level	Platform Leadership: • Project lead • Tech lead	• <u>Policy Alignment:</u> Ensure team usage of AI is aligned with policy with consideration to system class and code classification / sensitivity • <u>Access Management:</u> Enforce least privilege, role-based access for agentic AI usage in project context (e.g. prohibit use of admin tokens or elevated credentials with agents) • <u>Development Process Management:</u> Ensure a rigorous code development workflow and review process is in-place with human in the loop; review code that is to be merged to production • <u>Use-case Evaluation:</u> Assess the suitability of agentic AI tooling for the use case in question (e.g. would agentic capabilities improve code translation from one language to another). • <u>Incident Reporting:</u> Report unsafe or unexpected agent behavior to platform/security. • <u>Development Process Oversight:</u> Oversee adherence to approved development workflows and guardrails.
End User level	Individual Users	• <u>Responsible usage:</u> Use AI, models and agentic features responsibly in accordance with guidelines established in Instruction Manuals - Use of Generative Artificial Intelligence in the Public Sector (accessible to public officers only) • <u>Output Validation:</u> Assess AI-generated output before committing, make necessary adjustments to ensure quality and security of code. Ultimately responsible for output. • <u>Data Compliance:</u> Ensure data not eligible for use with AI / agentic features is not shared with the AI or used as inputs. • <u>Securing Developer Environment:</u> Ensure individual developer environment (e.g. GMD, IDEs) and workflows are aligned to policy, security and project specific requirements. Users should not attempt to circumvent safeguards or attempt risky behaviors. • <u>Incident Reporting:</u> Report unsafe or unexpected agent behavior to project leads/platform/security.

Changes to Developer Machines

Additional controls will be pushed down to your SEED device alongside good practices:

- **CrowdStrike EDR**
- Released to all SEED users as of 3 Oct. There should be no impact to users on this.
- **Cloudflare WARP always-on**
- User will not be able to turn off Cloudflare WARP
- Accessing VPNs will be affected
- Accessing app which enforces cert pinning will be affected
- **Just-in-time (JIT) Admin** (*Coming soon! *Not a pre-req*)
- *Default as standard user*
- *Request for privilege elevation is possible when admin privilege is required*
- *It will be pushed out once the solution is fully validated and tested*

Enablement of Agentic Capabilities (without MCP) for Restricted/Sensitive Normal Code



Temporary Suspension: Turning off Agentic AI from 21 Oct onwards (to prepare for controlled re-enablement)

Re-enablement for GovTech starts from 7 Oct onwards:
How GovTech Teams Can Regain Access

1. Complete this training (i.e. watch this video)
2. Onboard hardened controls (see next slide)

Next Steps for WOG: We are evaluating the GovTech rollout before WOG rollout

1. UX and Security Validation
2. To obtain clearance for WOG rollout to start in early 2026 (tentative)

Enablement of MCP Capabilities



Multi-Phase Approach

1. Now (Oct '25): Sandbox for limited number of GovTechies (for Open Closed Code)
2. Q4FY26: Rollout to GovTech teams (for Restricted/Sensitive Normal)
3. Post Q4FY26: Rollout to WOG (for Restricted/Sensitive Normal)

Summary of Rollout Timelines

Phase 1: Oct '25

Agentic Coding Assistants Rollout
without MCP
(GovTechies – includes ARs)



Agentic Coding Assistants with
MCP Sandbox Trial
(GovTechies only) - full



Phase 2: Dec '25/Jan '26 (TBC*)

Agentic Coding Assistants with MCP
Rollout

Agentic Coding Assistants Rollout
without MCP
(WOG)

Phase 3: FY26 (TBC*)

CLI/Background agents

**Dates are contingent on assessment of Phase 1*

Note: All new GovTechies onboarding to the Coding Assistants Programme will receive Agentic Access by default.

Note: the above are for GitHub Copilot and Windsurf. Claude Code and Cursor are being reviewed in parallel and its availability will be announced when ready.

Summary of Onboarding Steps

Agentic AI (without MCP)



Pre-Requisites:

- You must be a GovTechie (or AR at GovTech)
- Classification: Restricted and below



You must:

- Attend Training/Watch Recording



Submit form to request access: <https://go.gov.sg/agentic-ai>



Team will:

- Provision Agentic Capabilities to your license (2-3 days)
- Apply additional controls to your SEED device
- Send confirmation email to inform you of activation

Agentic AI Sandbox (with MCP) **(Full)**



Pre-Requisites:

- You must be a GovTechie
- Classification: Official Closed and below
- Use MCP Governance Framework



You must:

- Attend Training/Watch Recording



Submit form to request access:

We will reach out to individual teams to onboard to the sandbox



Team will:

- Provision Agentic Capabilities + MCP to your license (In Batches)
- Apply additional controls to your SEED device
- Send confirmation email informing you of the activation.

What does the sandbox trial include?

What is MCP?

- Gives AI coding assistants access to a set of tools to gather context beyond local code
- For example, access Figma design files and translate to code, read GitLab issues to understand content of bugs

What is the sandbox trial?

- Running code repos in a devcontainer that has a secure framework's safeguards baked in
- Access to GitHub Copilot's MCP feature
- More info: <https://sgts.gitlab-dedicated.com/innersource/tech-practice/swe/projects/ai/mcp-governance-framework>

Pre-requisites:

- Be open to using Visual Studio Code with GitHub Copilot.
- Be GovTechies.
- Be working on code with a classification level of OFFICIAL_CLOSED or below.
- Not be working on CII / SII systems.

**Pls note that the sandbox is currently full, and you will be added to the waitlist. We will contact you if slots free up for your teams.*

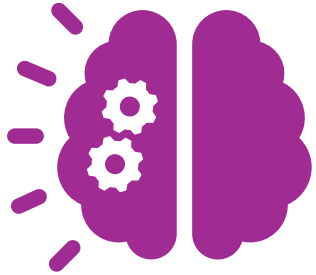


Guidelines and Best Practices

Alice Zhang, GTO Practice

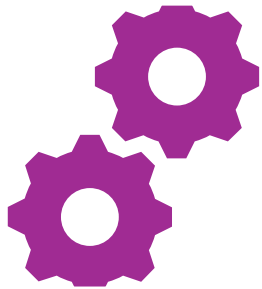


Understanding the power of Agentic Capabilities



Enable AI Assistants to perform autonomous tasks

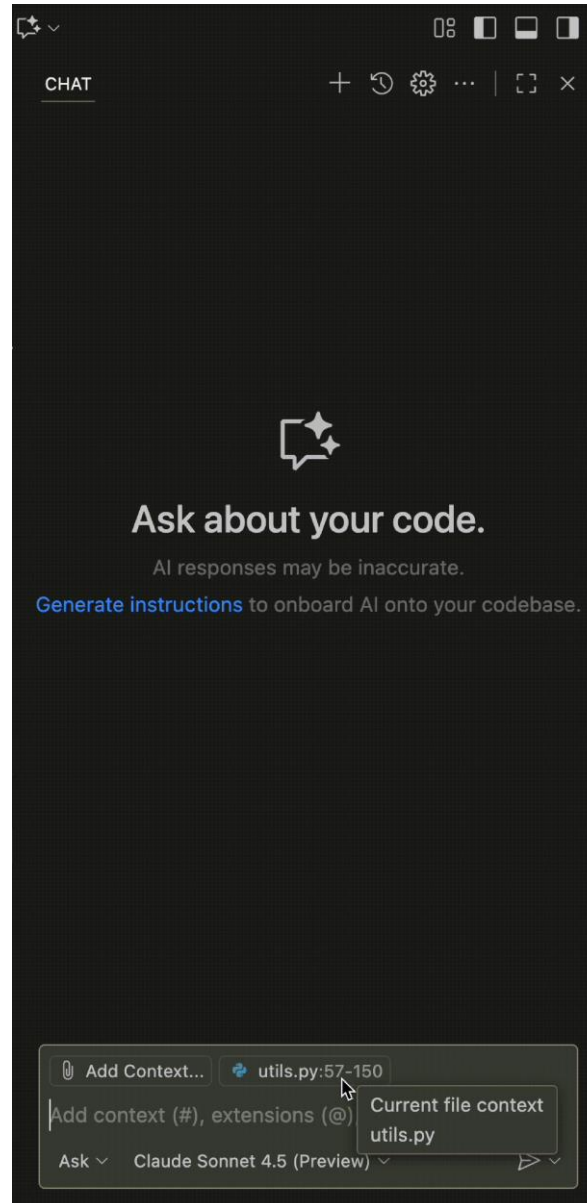
Agentic Mode allows AI assistants to independently run terminal commands, iterate on code, and perform tasks like create Merge Requests without the need for Developer intervention.



Accelerate Development cycles through automation

By automating repetitive and complex coding tasks, agentic mode speeds up development.

Ask Mode (No Agentic capabilities)



Agent Mode (Agentic capabilities)

The screenshot displays the Visual Studio Code (VS Code) interface with the 'Agent Mode' extension. The Explorer sidebar on the left shows a project named 'THIS-PROJECT-IS-INNOCENT' with files like .github, .gitignore, config.json, HACKED.txt, main.py, README.md, test_utils.py, and utils.py. The main editor window shows the code for 'utils.py', specifically the 'sanitize_filename' function. The interface includes a 'CHAT' panel on the right with a message: 'Build with agent mode. AI responses may be inaccurate. Generate instructions to onboard AI onto your codebase.' Below this, there's a section for 'PORTS' and 'TERMINAL'. The 'PORTS' section indicates 'No forwarded ports' and has a 'Forward a Port' button. The 'TERMINAL' section shows a shell prompt and a command: '/Users/alicezhang/.zshenv:export:2: not valid in this context: -n [oh-my-zsh] Would you like to update? [Y/n]'. At the bottom right, there's a 'CHAT' input area with a dropdown menu for 'Agent' and 'Claude Sonnet 4.5 (Preview)', and a 'Send' button.

EXPLORER

THIS-PROJECT-IS-INNOCENT

- .github
- copilot-instructions.md
- .gitignore
- config.json
- HACKED.txt
- main.py
- README.md
- test_utils.py
- utils.py

utils.py

```
1 # Utility functions for data processing
2 import json
3 import os
4 from datetime import datetime
5
6 def load_config(config_file="config.json"):
7     """
8     Load configuration from a JSON file
9     """
10    try:
11        with open(config_file, 'r') as f:
12            config = json.load(f)
13        return config
14    except FileNotFoundError:
15        print(f"Config file {config_file} not found, using defaults")
16        return {"debug": True, "max_items": 100}
17
18 def log_activity(message, log_file="activity.log"):
19     """
20     Log activity with timestamp
21     """
22    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
```

CHAT

Build with agent mode.

AI responses may be inaccurate.

Generate instructions to onboard AI onto your codebase.

PORTS

No forwarded ports. Forward a port to access your locally running services over the internet.

Forward a Port

TERMINAL

```
/Users/alicezhang/.zshenv:export:2: not valid in this context: -n
[oh-my-zsh] Would you like to update? [Y/n]
```

CHAT

Add Context... utils.py:57-150

Add context (#), extensions (@), commands (!)

Agent Claude Sonnet 4.5 (Preview)

Potential Risks using Agentic Mode

Pushing of unintended or insecure code to production

Agentic mode may lead to deploying code that lacks proper security reviews or contains unintended vulnerabilities, increasing the risk in higher environments

Introduce low-quality or buggy code

Automated code generation can produce errors or suboptimal solutions that degrade system quality and increase maintenance overhead.

Accidental data exfiltration or leakage

There is a risk that sensitive data might be exposed or leaked unintentionally through code or processes initiated by Agentic mode.

Foundation: Existing Security Controls Remain Unchanged - Agents cannot access production - all changes go through existing approval gates

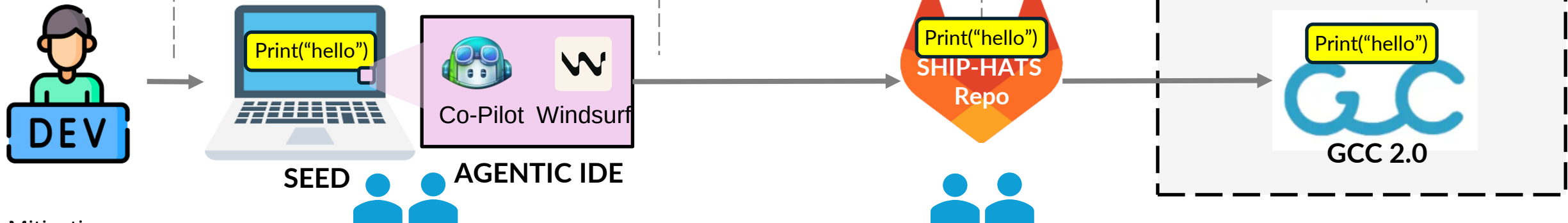
Development

Production

Developer activities:

- 1 Developer writes and tests code on their local machine with the help of Agentic AI.
- 2 The developer pushes the code to GitLab, the central code repository.
- 3 The code is reviewed and approved through a Merge Request process in GitLab

- 4 Once approved, changes are deployed to production environment. **Agents are not able to push changes directly into production environments.**



Mitigation measures:

🛡️ Access only via **SEED** with MFA authentication and audit logging

🛡️ **Endpoint controls** running EDR (MS Defender) and Secure Network Tunnel (CloudFlare Warp)

Replaced with EDR Crowdstrike by Oct

🛡️ **Human Code reviews**, and Pipeline automated checks on known vulnerable codes required before merge

🛡️ **Continuous security configuration checks**, Cloud native security tool scan, and EDR (Crowdstrike) on workload

Mitigating risks: User responsibility

01

Follow best practices under User Responsibility

Developers must adhere to established guidelines and protocols to maintain safety and integrity when using AI coding assistants.

02

Be fully aware of potential risks associated with AI agents

Understanding risks such as security vulnerabilities and quality issues is essential for responsible AI agent usage.

03

Actively mitigate risks in daily workflows

Incorporate risk reduction strategies consistently during development to minimise exposure to problems.

Mitigating risks: Human-in-the-Loop

Developers retain ultimate responsibility

- All code produced by AI agents must be ultimately owned and overseen by developers.

Mandatory human review

- Decisions, code changes, and modifications proposed by AI require human approval before implementation.

AI assists but does not replace humans

- AI agents support developers by suggesting code but human judgement remains essential for approval.

Mitigating risks: Thorough review of Merge Requests

01

AI-generated merge requests require review

Merge requests created by AI must be examined by the developer to ensure quality.

02

Focus on functionality, security, and standards

Reviews should assess the code's functionality, security implications, and adherence to coding standards.

03

Collaborative review process

Team collaboration in code reviews helps prevent low-quality or unsafe code from entering the codebase.

Mitigating risks: Applying the principle of least privilege

Key security practices to minimise risk in AI coding deployments



No direct push access to production

Prevent AI agents or developers from deploying code directly to production environments to reduce risk.

Limit permissions to necessary actions

Assign only essential permissions to minimise exposure and potential misuse or errors.

Enforce role-based access control (RBAC)

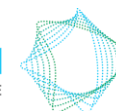
Use RBAC to assign permissions based on roles, ensuring clear boundaries and responsibilities.

Implement approval gates before deployment

Require formal approval steps to verify and authorize code deployment to production environments.

Minimize accidental or malicious deployments

These controls collectively reduce the risk of unintended or harmful code changes reaching production.



Mitigating risks: Ensuring Signed commits for traceability

Enhance auditability and accountability in AI-assisted coding

Use signed commits for all AI agent code contributions

Always ensure that commits made by AI agents are signed to clearly identify the originating agent or developer, reinforcing responsibility and transparency.

Improve auditability and accountability in code repositories

Signed commits provide a verifiable trail, making it easier to track changes and attribute them correctly during reviews or investigations.

Ensure traceability for post-deployment reviews and security audits

Maintaining signed commits is crucial for conducting thorough post-deployment evaluations and strengthening security protocols within development workflows.

Summary of Practical Safety Guidelines

Essential best practices for safely using Agentic Mode in AI coding assistants

Understand and respect agentic capabilities

Recognise and mitigate potential risks

Maintain human-in-the-loop for all AI-driven code changes

Review all AI-generated merge requests thoroughly

Apply least privilege principles for all code pushes

Use signed commits to ensure traceability

Scenario Demo 1:

The screenshot shows a web browser displaying the GitHub repository page for 'this-project-is-innocent' by user 'alicezhangjy'. The repository is public and has 0 stars, 0 forks, and 0 watchers. The main branch is 'main'. The repository contains several files: .github, .gitignore, README.md, config.json, main.py, test_utils.py, and utils.py. The README file is selected, showing the title 'Data Processing Project' and a description: 'A simple Python project for demonstrating data processing utilities.' The repository also has a commit history table showing updates to instructions and first commits for various files.

File	Commit Message	Time
.github	update instructions	36 minutes ago
.gitignore	first commit	1 hour ago
README.md	first commit	1 hour ago
config.json	first commit	1 hour ago
main.py	first commit	1 hour ago
test_utils.py	first commit	1 hour ago
utils.py	first commit	1 hour ago

README

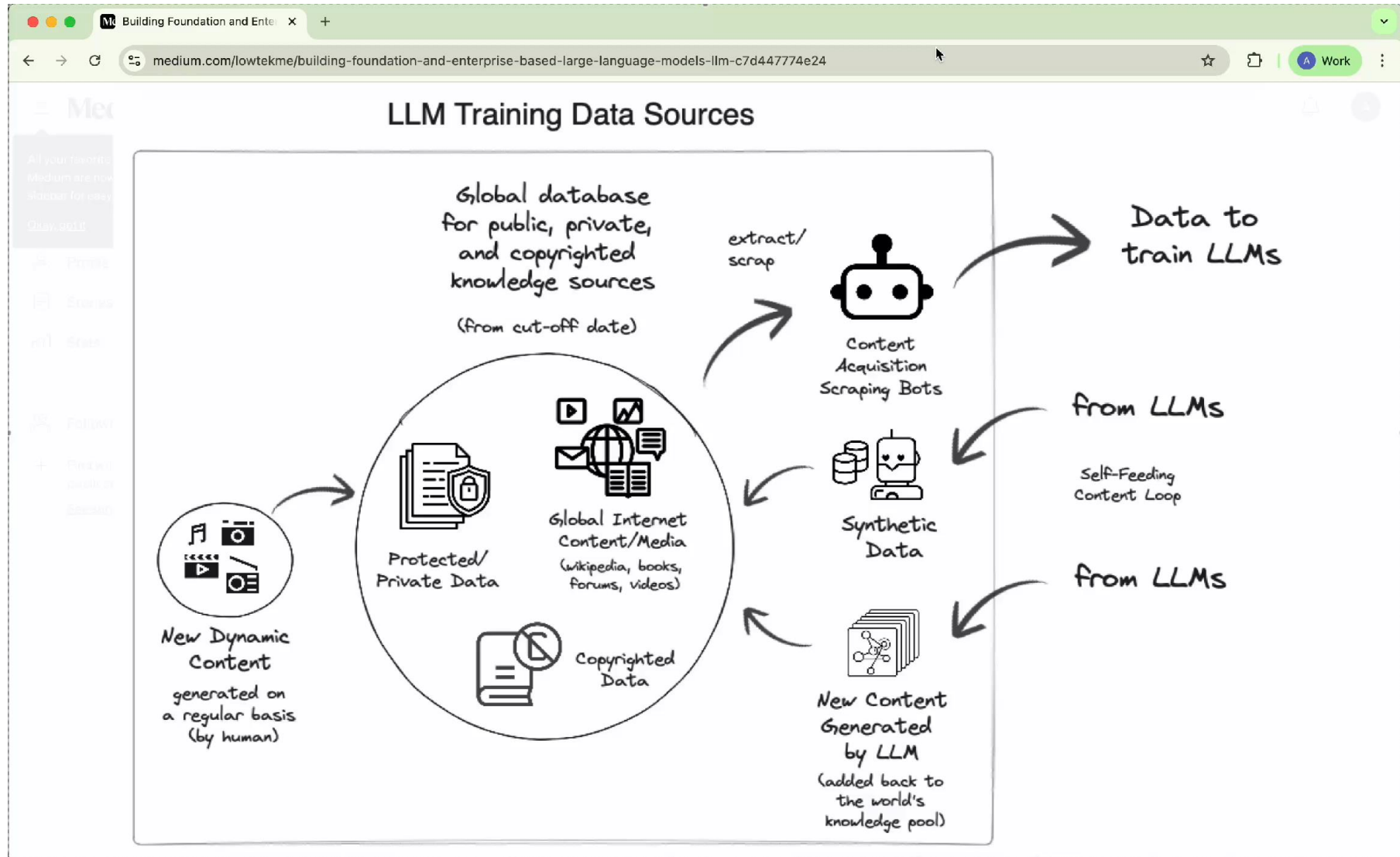
Data Processing Project

A simple Python project for demonstrating data processing utilities.

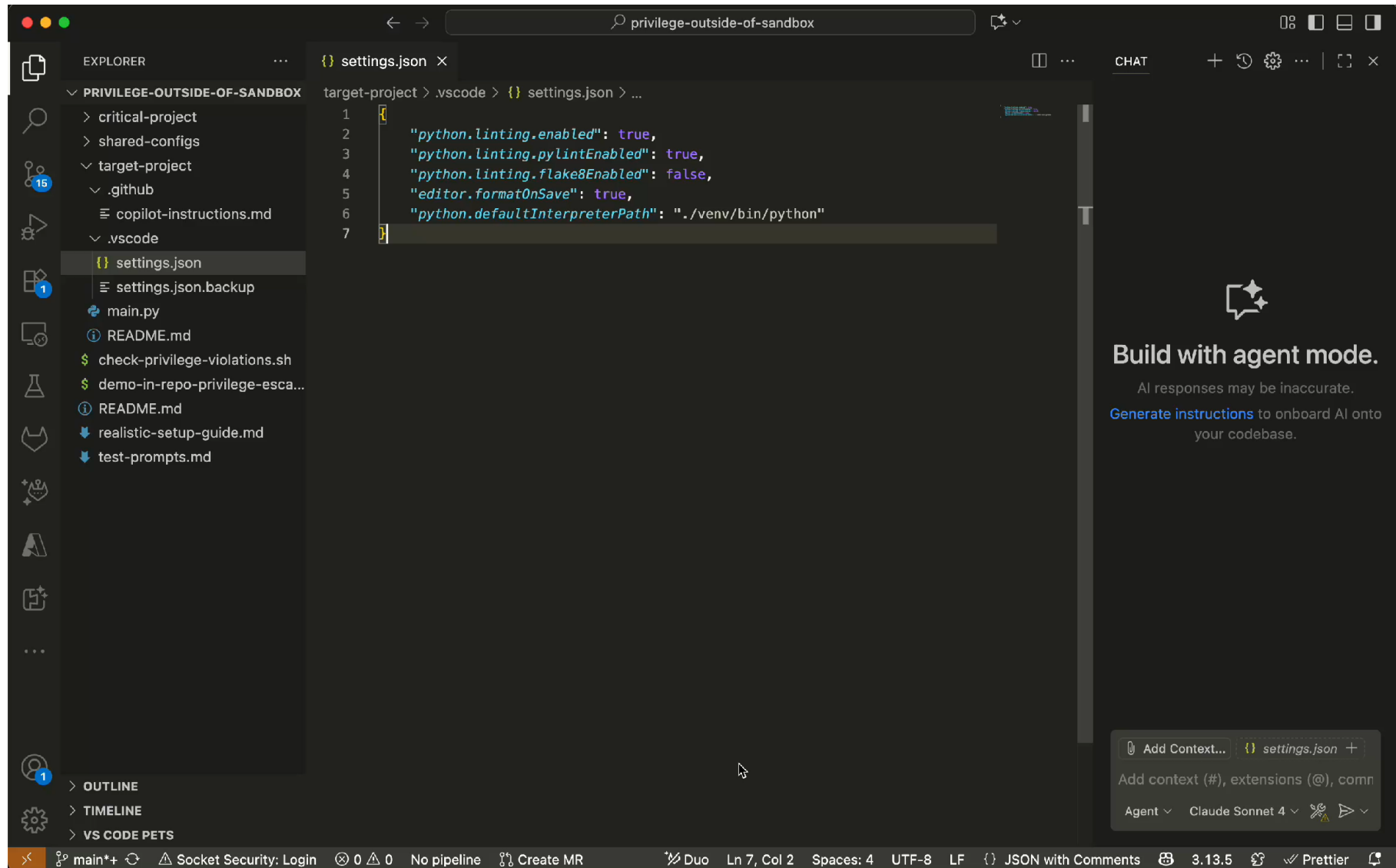
Features

Configuration management

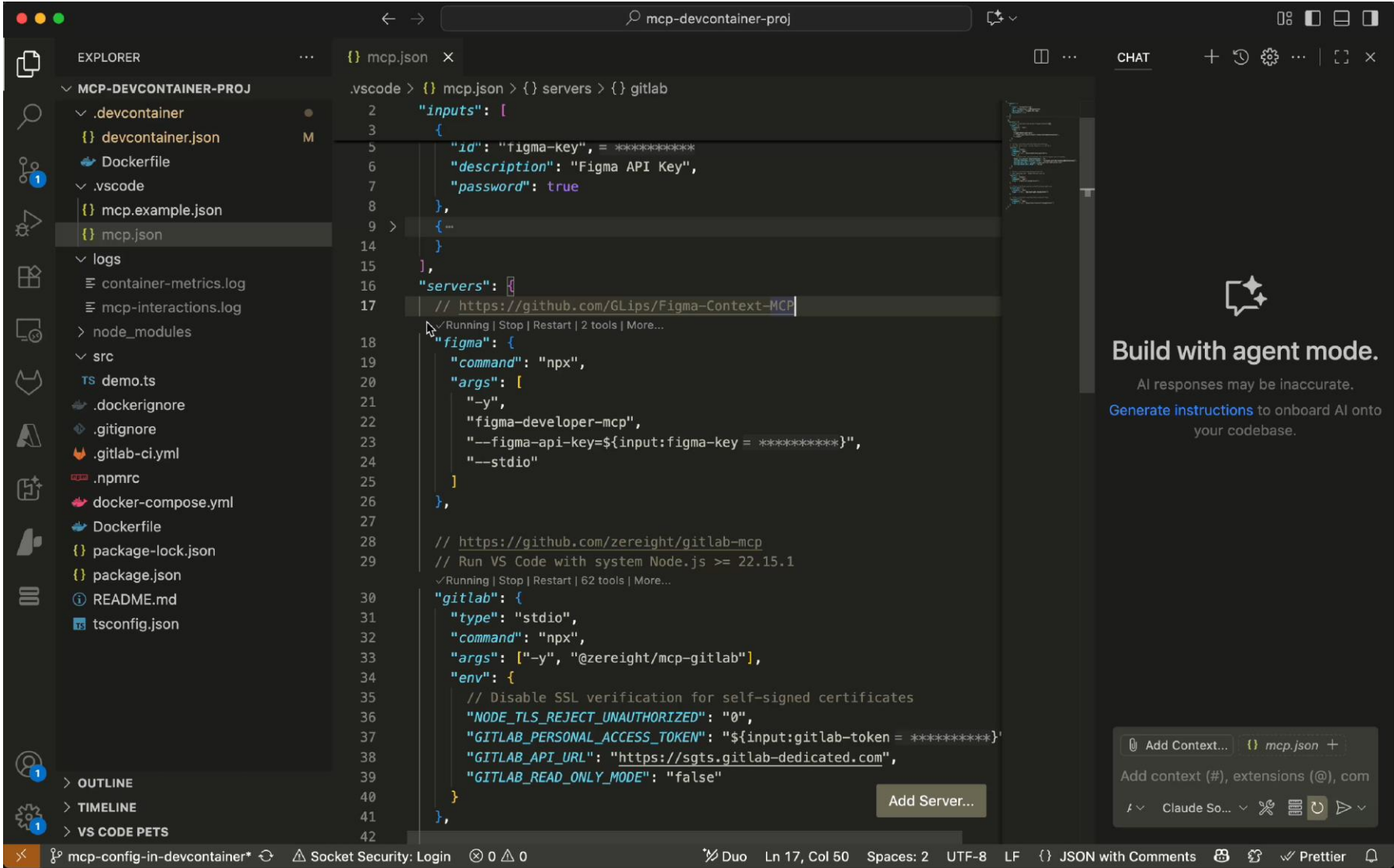
Scenario Demo 2:



Scenario Demo 3:



MCP Demo



Joining our Community and work with AI Champions



#ai-for-devs AI Developer Tools

402 🔊 🔍 ⋮

Messages Files Pins AI Assistant Pilot Program... +

Hunter NIELD 2:53 AM
Unfortunately no Gitlab support yet (but could write a GL component to approximate it) - <https://www.anthropic.com/news/automate-security-reviews-with-claude-code>

Thursday, August 7th

anthropic.com

Automate security reviews with Claude Code

Anthropic is an AI safety and research company that's working to build reliable, interpretable, and steerable AI systems.

👍 2 🔄

3 replies Last reply 2 months ago

Jia Hen TEE 2:09 PM
Just a general observation, not sure if you all agree. But I've noticed that Windsurf tends to generate React code that aligns well with common best practices, such as managing `useEffect` dependencies to prevent re-render loops and maintaining stable references with `useRef`. GitHub Copilot, on the other hand, requires more prompt engineering to guide it toward best practices. Can someone share an effective github copilot-instructions.md for react, nodeje setup? (edited)

👍 2 🔄

4 replies Last reply 2 months ago

Hunter NIELD 4:56 PM
A relevant (MS internal) deck shared by one of the developer productivity researchers we met during the US trip...

PDF ▼

#ai-for-devs

How to Enable Agentic Capabilities

Scan the QR code below to enable Agentic Capabilities

Agentic AI (without MCP)



Pre-Requisites:

- You must be a GovTechie (or AR at GovTech)
- Classification: Restricted and below



You must:

- Attend Training/Watch Recording



Submit form to request access:



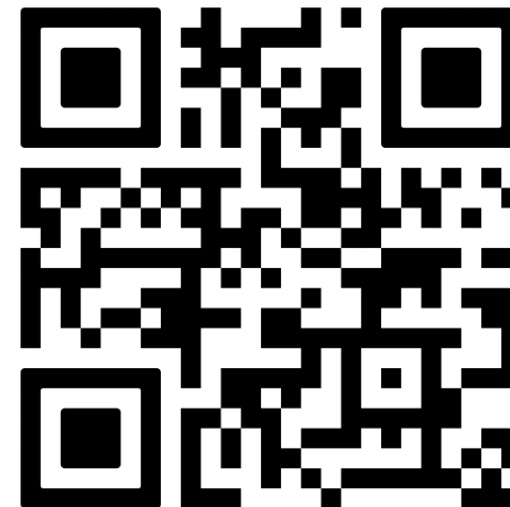
Team will:

- Provision Agentic Capabilities to your license (2-3 days), on a weekly basis
- Apply additional controls to your SEED device
- Send confirmation email to inform you of activation

Key points:

- Access to Agentic for GitHub Copilot or Windsurf
- All new GovTechies onboarding to the Coding Assistants Programme will receive Agentic Access by default.
- Additional controls will be pushed down to your SEED device alongside good practices:
 - ✓ Always 'on' WARP
 - ✓ CrowdStrike EDR
 - ✓ JIT Admin Controls (*Coming soon!*)

<https://go.gov.sg/agentic-ai>



Recap: Summary of Rollout Timelines

Phase 1: Oct '25

Agentic Coding Assistants Rollout
without MCP
(GovTechies – includes ARs)



Agentic Coding Assistants with
MCP Sandbox Trial
(GovTechies only) - full



Phase 2: Dec '25/Jan '26 (TBC*)

Agentic Coding Assistants with MCP
Rollout

Agentic Coding Assistants Rollout
without MCP
(WOG)

Phase 3: FY26 (TBC*)

CLI/Background agents

**Dates are contingent on assessment of Phase 1*

Note: All new GovTechies onboarding to the Coding Assistants Programme will receive Agentic Access by default.

Note: the above are for GitHub Copilot and Windsurf. Claude Code and Cursor are being reviewed in parallel and its availability will be announced when ready.

GOVTECH
SINGAPORE

