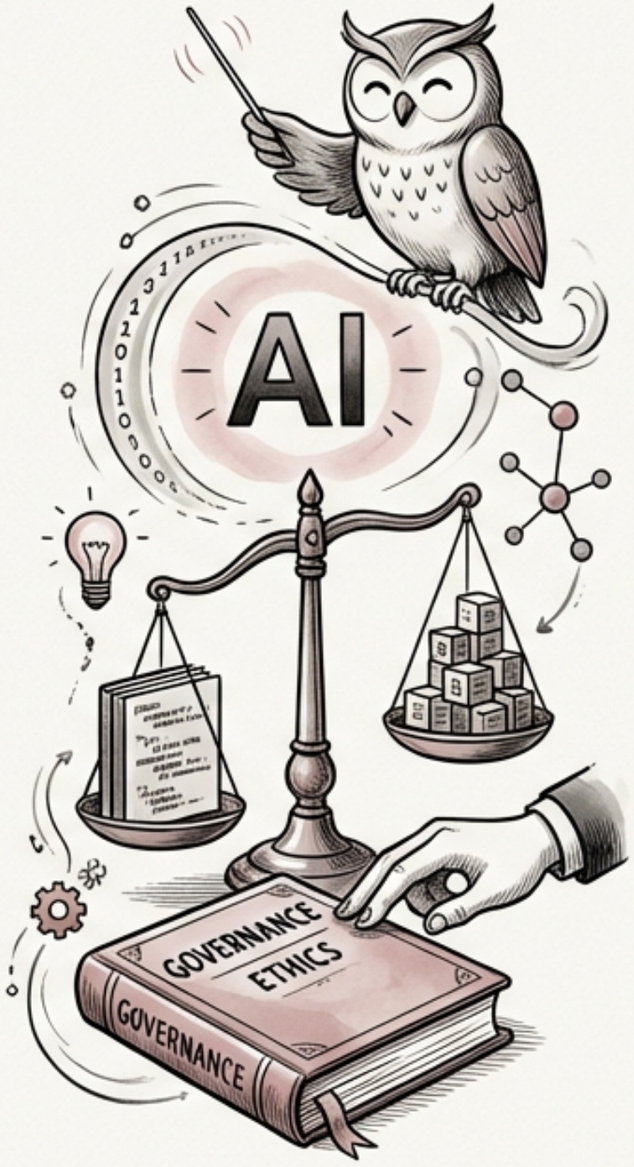
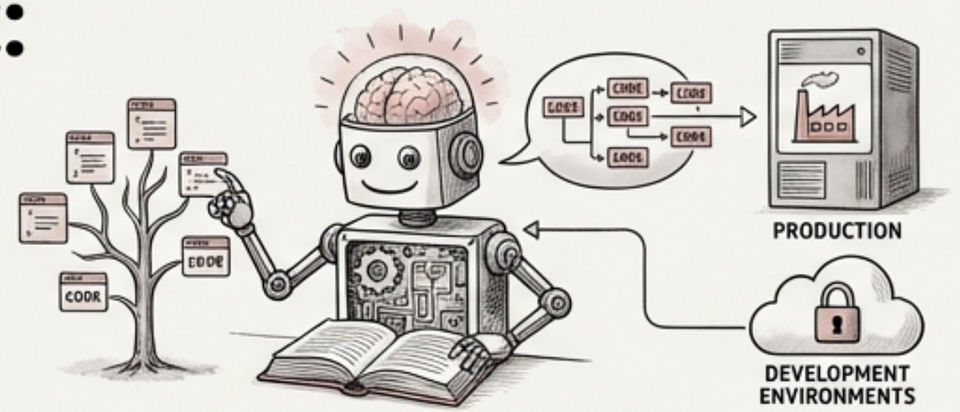
# AI Governance in Development: Why It Matters for Your Code

PRESENTATION BY [YOUR NAME] | DATE: OCTOBER 26, 2024

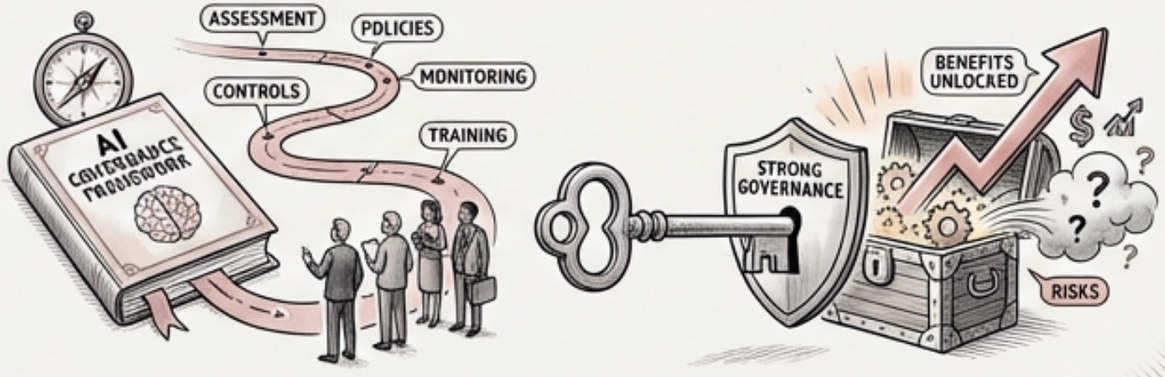


# AI Governance in Development: Why It Matters for Your Code

- AI tools in development are fundamentally different from traditional software due to their ability to process code, generate production software, and access development environments.
- AI tools can create new intellectual property, introducing complex ownership and licensing considerations.
- Without governance, AI tools can introduce security vulnerabilities, compliance violations, and legal liabilities into the software development lifecycle.
- This module provides a comprehensive guide to establishing an AI governance framework tailored for software development teams.
- Implementing strong AI governance helps to unlock the benefits of AI while mitigating potential risks.





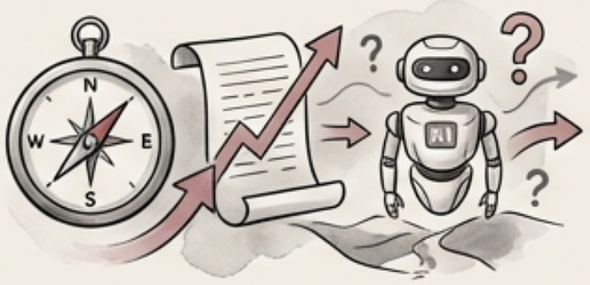



# Establishing Your AI Governance Board: Composition and Authority

- The AI Governance Board is crucial for overseeing **AI tool usage, policy changes, incident response,** and **exception reviews.**

- Recommended board **composition** includes representatives from **Security (CISO), Legal** (IP and liability), **Engineering** (productivity), **Compliance** (regulatory), and **Risk** (enterprise risk).

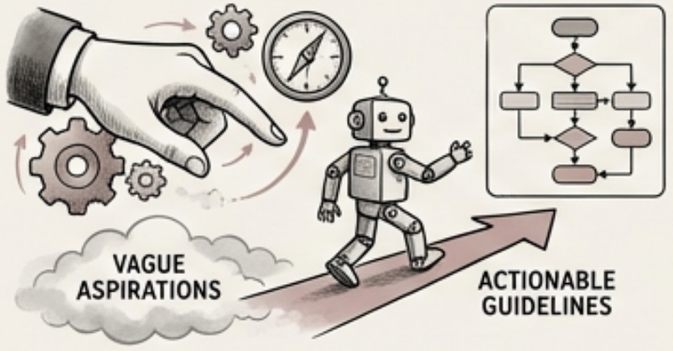
- The Board's **decision authority** includes **approving new AI tools** for use within the development environment.

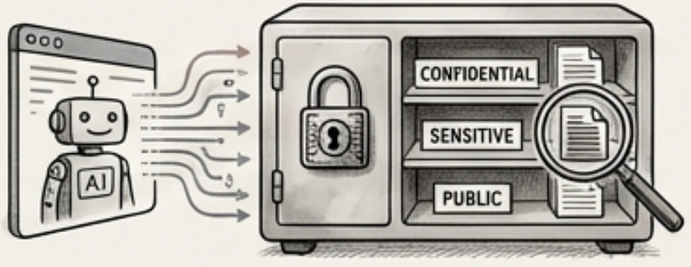


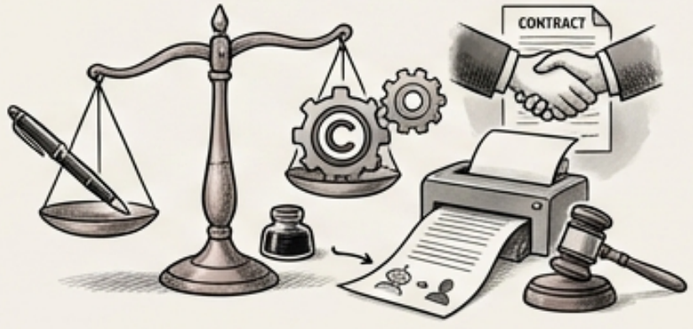

- The Board is responsible for **adapting and evolving AI use policies** in response to new technologies and emerging risks.

- A minimum **monthly meeting cadence** is required, with provisions for emergency meetings in case of critical incidents.

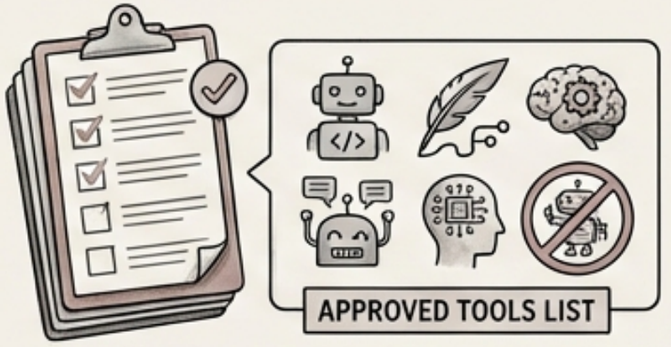
# CRAFTING AN ACTIONABLE AI ACCEPTABLE USE POLICY: EIGHT KEY DOMAINS

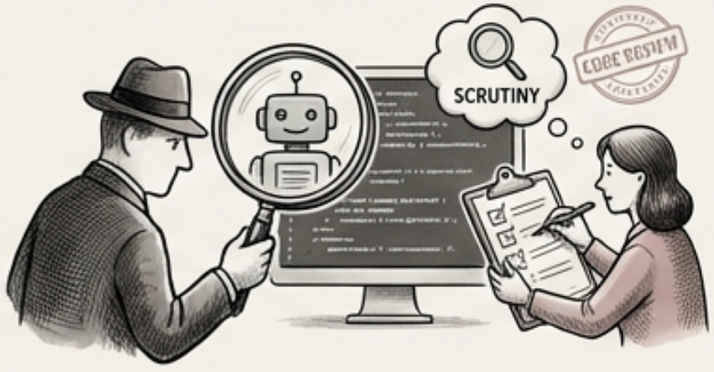



- The **Acceptable Use Policy (AUP) must be specific enough** to be **actionable** and avoid vague, aspirational statements.
- The AUP should include an **approved tools list**, clearly defining which AI coding assistants are permitted for use.
- **Data classification rules** must specify how sensitive data should be handled by AI tools.
- **Code review requirements** should outline the level of scrutiny required for AI-generated code.
- **Attribution and IP handling guidelines** need to clarify how AI-generated code is attributed and how IP rights are managed.



# Three-Tier Tool Classification: Balancing Productivity and Risk

### TIER 1 (UNRESTRICTED)

- IDE code completion on non-sensitive, non-regulated projects. Network monitoring is optional.


### TIER 2 (CONTROLLED)

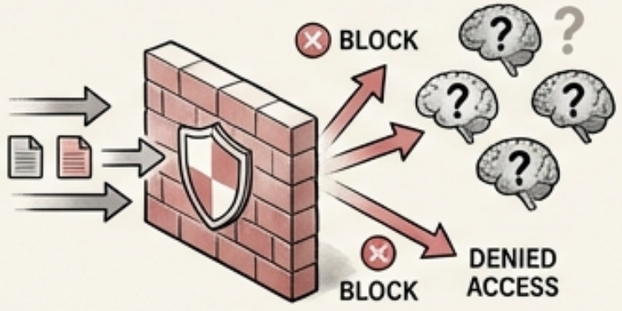
- Cloud AI on internal code requires DLP controls, approved configurations, output review, and manager approval. Network monitoring is mandatory.


### TIER 3 (PROHIBITED)

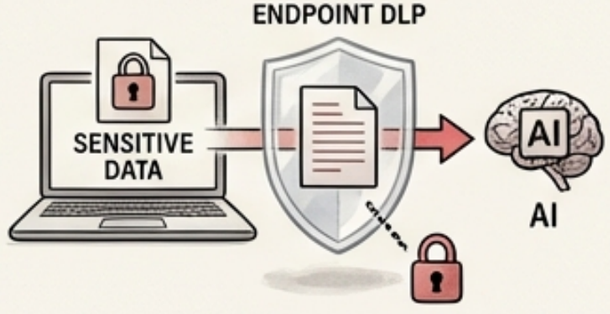
- AI on regulated data (PCI, HIPAA, ITAR), production secrets, PII, and customer data is strictly forbidden.


### TECHNICAL ENFORCEMENT IS CRITICAL

- network-level blocking prevents access to unauthorized AI services.

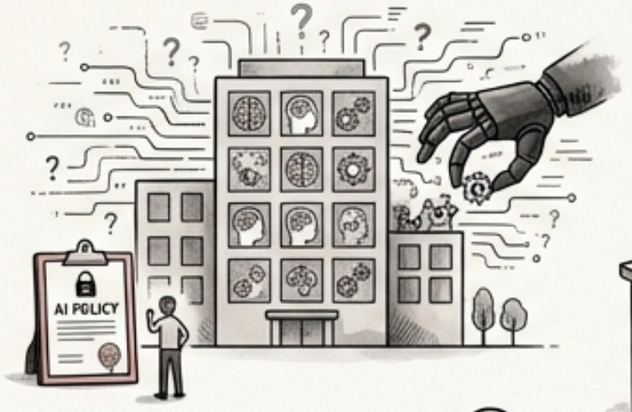



- Endpoint DLP (Data Loss Prevention) prevents sensitive data from being sent to AI tools.

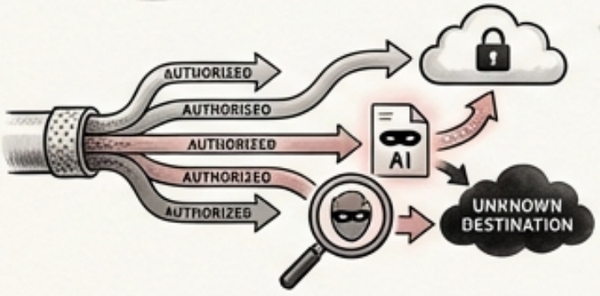


# COMBATING SHADOW AI: IDENTIFYING AND MITIGATING UNAPPROVED TOOL USAGE

- **98%** of organizations use AI tools, but only 37% have governance policies in place.
- **89%** of **developers** use unapproved AI tools, creating significant security and compliance risks.
- **Network traffic analysis** can detect unauthorized AI tool usage by monitoring outbound connections.
- **Endpoint monitoring** can identify AI tools installed on developer machines.
- **License audits** can uncover AI tools being used without proper authorization.

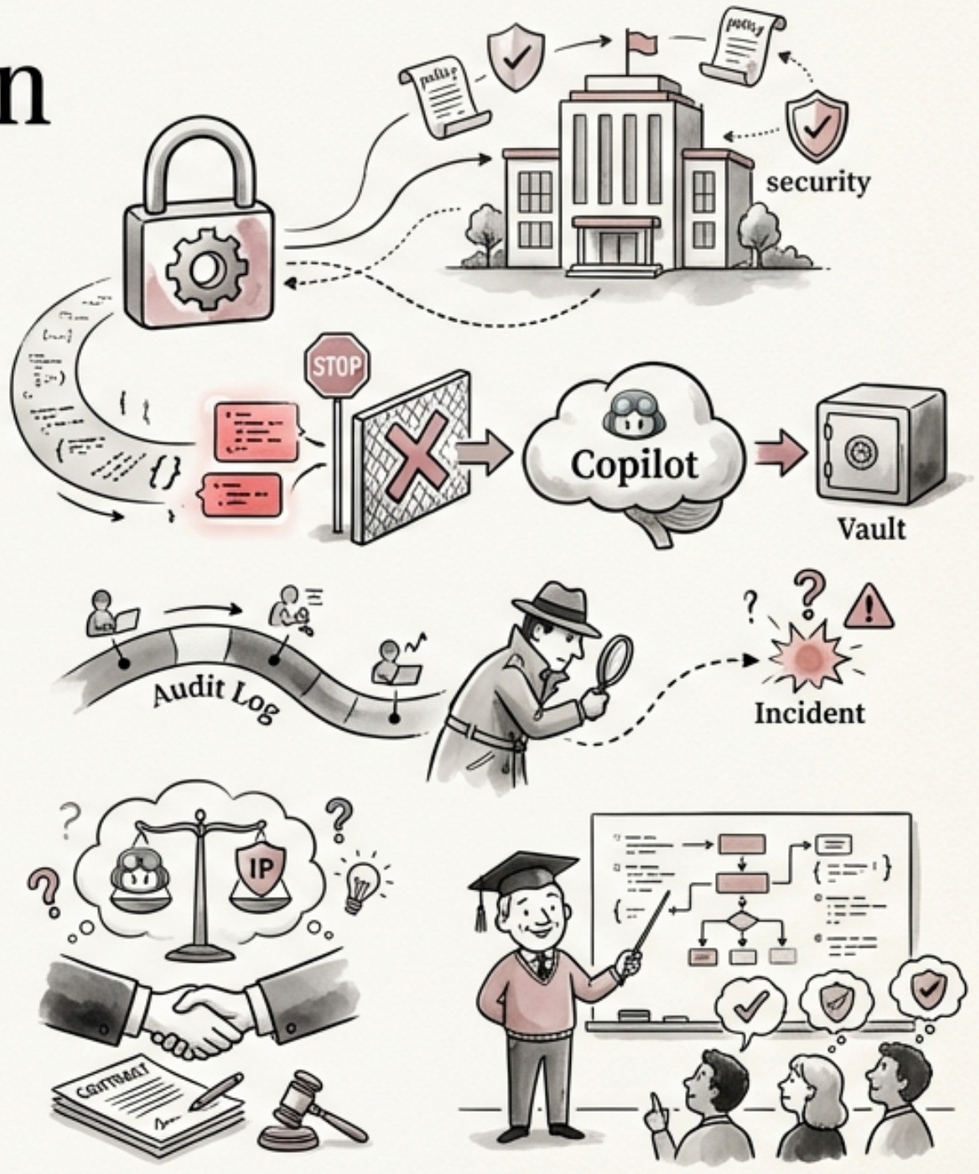





# Secure Integration Patterns: GitHub Copilot Configuration

- Implement organization-level settings within GitHub Copilot to enforce security and compliance policies.
- Use content exclusion rules to prevent sensitive code or data from being processed by Copilot.
- Enable audit logging to track Copilot usage and identify potential security incidents.
- Understand the IP indemnification requirements associated with using GitHub Copilot.
- Educate developers on how to effectively use Copilot without compromising security or compliance.



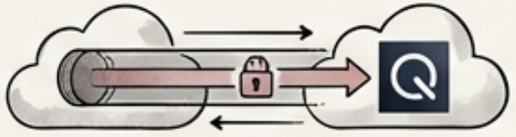
# SECURE INTEGRATION PATTERNS: CURSOR IDE CONSIDERATIONS

- Establish clear workspace trust boundaries to isolate sensitive projects within Cursor IDE.
- Carefully manage extension permissions to limit the capabilities of third-party extensions.
- Define appropriate data retention settings to control how long Cursor IDE stores code and data.
- Implement secure authentication and authorization mechanisms to protect access to Cursor IDE.
- Educate developers on the security implications of using different Cursor IDE features.

# Secure Integration Patterns: Amazon Q Integration

- Leverage IAM (Identity and Access Management) integration to control access to Amazon Q resources.

- Utilize VPC (Virtual Private Cloud) endpoints to restrict network access to Amazon Q services.



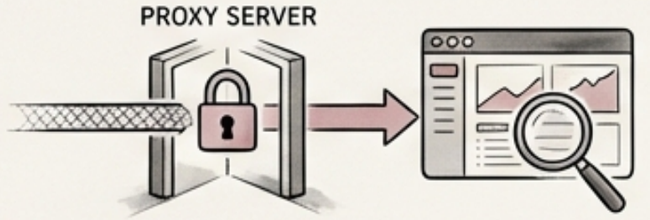

- Implement S3 bucket policies to control access to code context data stored in S3.


- Configure appropriate logging and monitoring to track Amazon Q usage and identify potential security incidents.

- Ensure that developers are trained on how to use Amazon Q securely and in compliance with company policies.

# SECURE INTEGRATION PATTERNS: ChatGPT/Claude Usage

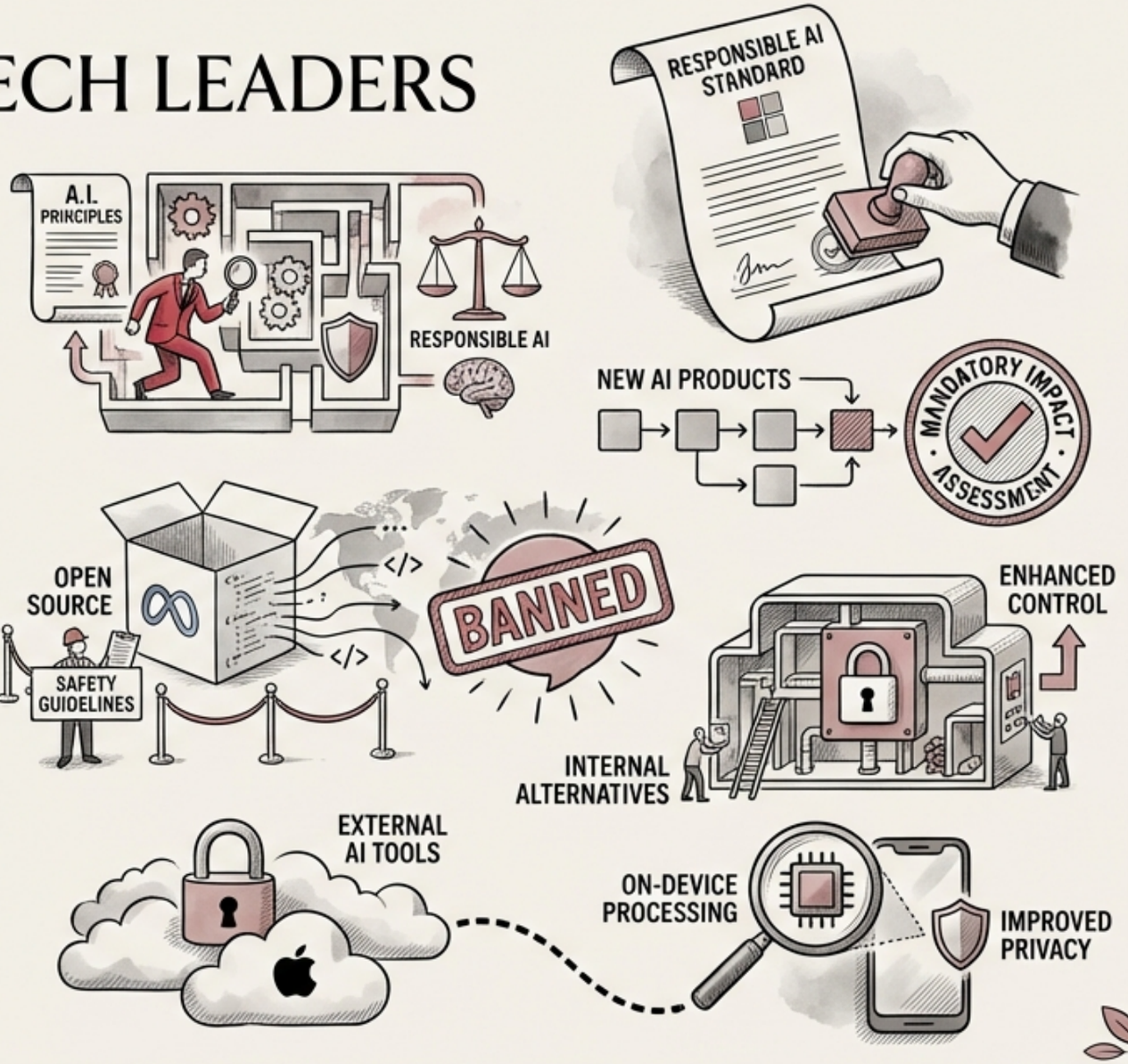
- Use API-only access with a proxy server to control and monitor traffic to ChatGPT/Claude.
- Prohibit direct browser access to ChatGPT/Claude from developer machines to prevent data leaks.
- Implement comprehensive output logging to capture all interactions with ChatGPT/Claude.
- Enforce strict data masking and anonymization policies before sending data to ChatGPT/Claude.
- Educate developers on the risks associated with sharing sensitive information with ChatGPT/Claude.





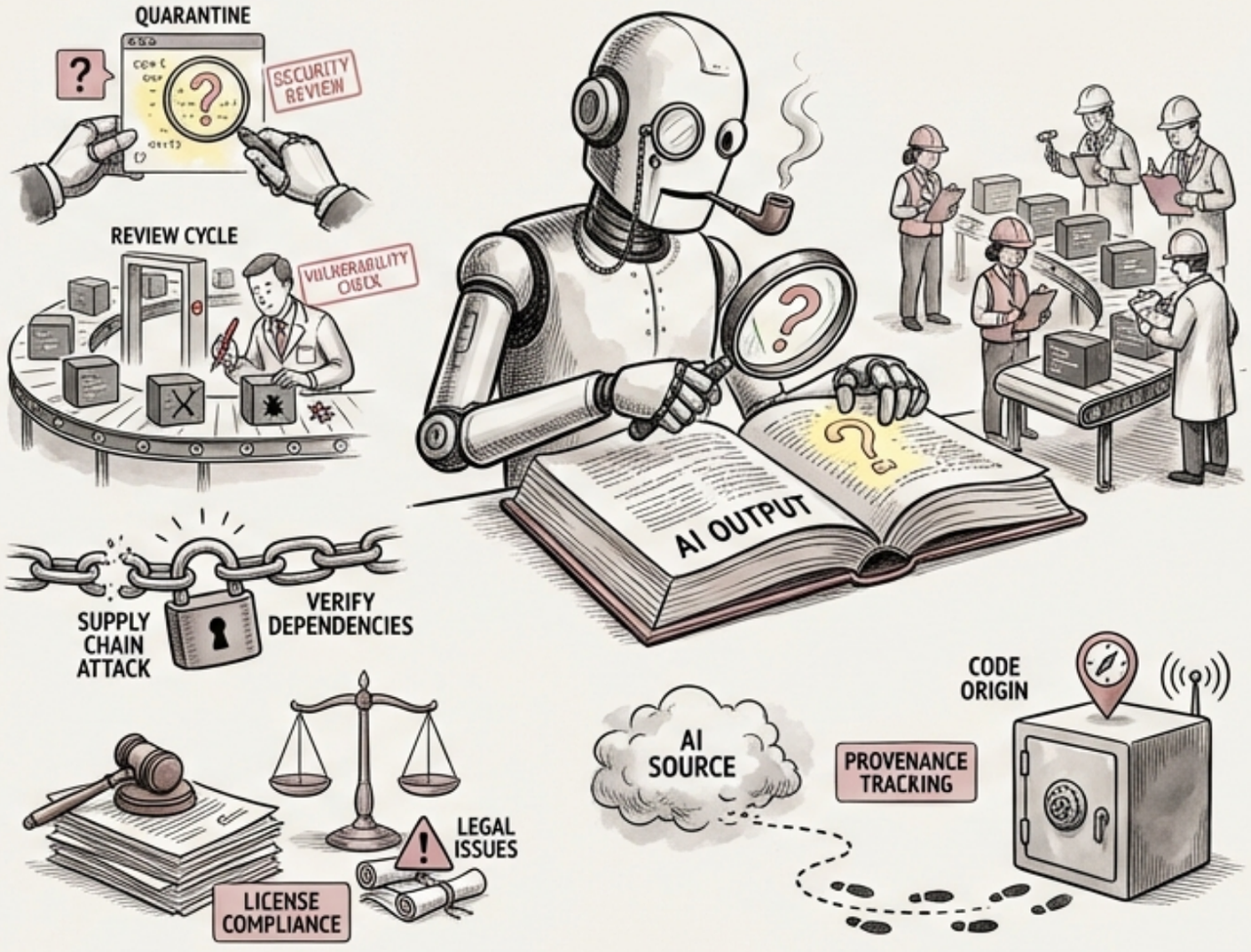

# INDUSTRY AI POLICIES: KEY TAKEAWAYS FROM TECH LEADERS

- Google emphasizes internal AI principles, red team testing, and responsible AI practices.
- Microsoft has a Responsible AI Standard and requires mandatory impact assessments for new AI products.
- Meta takes an open source approach but provides safety guidelines for AI development.
- Samsung banned ChatGPT after data leaks and is building internal alternatives for enhanced control.
- Apple has restricted external AI tool usage and is investing in on-device processing for improved privacy.

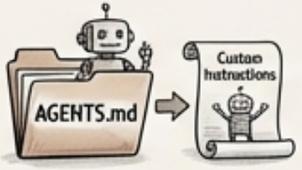


# OpenSSF Guidance: Securing AI-Assisted Development

- Treat AI output as untrusted code and subject it to rigorous security review.
- Implement mandatory review cycles for all AI-generated code to identify potential vulnerabilities.
- Verify dependencies used by AI-generated code to prevent supply chain attacks.
- Ensure license compliance for all AI-generated code to avoid legal issues.
- Implement provenance tracking to identify the source of AI-generated code.



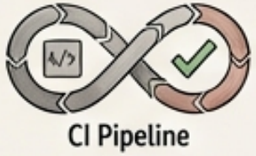
# Repository-Level AI Policies: AGENTS.md and Custom Instructions




- Use **AGENTS.md** and custom instruction files to define per-project AI behavior constraints.


- Specify security requirements, coding standards, and forbidden patterns within these files.


- Enforce these policies automatically through pre-commit hooks to prevent non-compliant code from being committed.




- Integrate these checks into the CI (Continuous Integration) pipeline to ensure ongoing compliance.

- Regularly update these files to reflect evolving security threats and coding standards.

# CSA Five-Step AI Governance Model: Adapting to Development Teams

- Identify AI assets used within the development environment, including tools, data, and models.
- Assess the risks associated with each AI asset, such as data leaks, security vulnerabilities, and compliance violations.
- Define controls to mitigate these risks, including technical controls, policies, and procedures.
- Implement monitoring mechanisms to track AI usage and detect potential security incidents.
- Continuously improve the AI governance framework based on feedback and lessons learned.



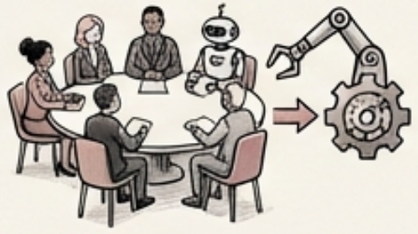


# Measuring Success: Key AI Governance KPIs for Your Team

- **Track tool adoption rates** to assess the uptake of approved AI tools within the development team.
- **Monitor policy compliance percentage** to measure adherence to AI usage guidelines.
- Calculate **shadow AI detection rate** to assess the effectiveness of shadow AI detection strategies.
- Compare **AI-generated code defect rate vs. human code** to evaluate the quality of AI-generated code.
- Measure **time to remediate AI-introduced vulnerabilities** to assess the speed of incident response.

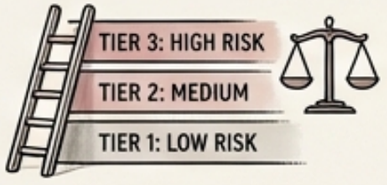
# Next Steps: Implementing AI Governance in Your Development Workflow

- Establish an AI Governance Board with cross-functional representation to oversee AI tool usage and policy changes.



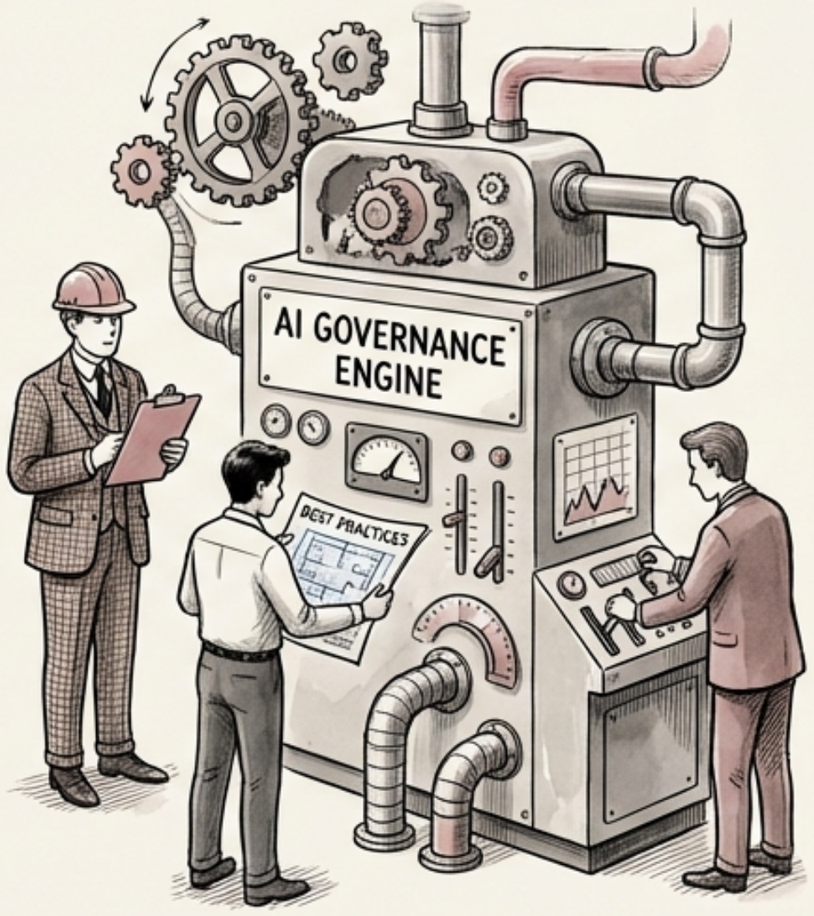
- Develop a clear and actionable AI Acceptable Use Policy that addresses key domains such as classification, code review, and IP handling.


- Implement a three-tier tool classification system to balance productivity and risk.




- Implement robust detection and mitigation strategies for Shadow AI.

- Educate developers on AI governance policies and best practices.



# Thank You

- Questions?