How to Write Secure AI Prompts

Prompt engineering for secure code generation

Step 1: Set Security Context in System Prompts

When configuring AI coding assistants, include security requirements in the system context or custom instructions.

- Specify: 'Always use parameterized queries, never string concatenation for SQL'
- Specify: 'Always validate and sanitize user input before processing'
- Specify: 'Use approved cryptographic libraries, never implement custom crypto'
- Specify: 'Follow OWASP secure coding guidelines for [your language]'

Step 2: Structure Prompts for Secure Output

Include security constraints in every code generation prompt:

- State the security requirement explicitly: 'Implement login with rate limiting and account lockout after 5 failed attempts'
- Specify what NOT to do: 'Do not log passwords, tokens, or PII'
- Request error handling: 'Include proper error handling that does not expose internal details'
- Ask for input validation: 'Validate all input parameters for type, length, and allowed characters'

Step 3: Apply the RCI Pattern

Every piece of AI-generated code goes through Review-Correct-Integrate:

- REVIEW: Read every line. Check for hallucinated APIs, missing validation, hardcoded values
- CORRECT: Fix security issues, add missing error handling, replace deprecated functions
- INTEGRATE: Run SAST scan, execute tests, submit for code review with AI-generated flag

AI code should receive MORE scrutiny than human-written code, not less.