AI Code Review Toolkit — PREVIEW

5 of 50 Prompts + 1 of 10 Scripts (Sample)

nopii.xyz

2026

AI Code Review Toolkit — FREE PREVIEW

This is a preview of the full toolkit. You're seeing **5 of 50 prompts** and **1 of 10 scripts**.

Get the full toolkit (50 prompts + 10 scripts): https://tabdullah8.gumroad.com/l/ai-code-review-toolkit

---

SECTION 1: Security Review (showing 2 of 15)

Prompt #1 — Full Security Audit

```
Perform a comprehensive security audit of this code. Check for:

1. Injection vulnerabilities (SQL, NoSQL, command, LDAP, XPath)
2. Broken authentication and session management
3. Sensitive data exposure (hardcoded secrets, API keys, passwords)
4. XML External Entity (XXE) vulnerabilities
5. Broken access control (IDOR, privilege escalation)
6. Security misconfiguration
7. Cross-Site Scripting (XSS) --- stored, reflected, DOM-based
8. Insecure deserialization
9. Using components with known vulnerabilities
10. Insufficient logging and monitoring

For each issue found:
- Severity: Critical / High / Medium / Low
- Line number(s) affected
- Attack scenario (how an attacker would exploit this)
- Concrete fix with corrected code

Code:
[PASTE CODE]
```

Prompt #5 — Smart Contract Security

```
Audit this Solidity smart contract for:

1. Reentrancy (external calls before state updates)
2. Integer overflow/underflow
3. Access control (missing onlyOwner, role checks)
4. Front-running vulnerabilities
5. Flash loan attack vectors
6. Delegatecall risks
7. Storage collision
8. Denial of service (gas limit, unbounded loops)
9. Oracle manipulation
10. Signature replay
```

```
For each: severity, attack PoC pseudocode, and fix.

Contract:
[PASTE CODE]
```

13 more security prompts in the full toolkit: Auth flaws, Injection detection, Crypto audit, API security, Dependency risk, Race conditions, File system, Cloud config, Memory safety, Docker, GraphQL, Mobile, CI/CD…

SECTION 2: Code Quality (showing 1 of 15)

Prompt #17 — Performance Bottlenecks

```
Analyze this code for performance issues:

1. N+1 query problems
2. Missing database indexes (suggest needed indexes)
3. Unnecessary memory allocations in loops
4. Missing caching opportunities
5. Synchronous operations that should be async
6. Inefficient algorithms (O(n^2) when O(n) is possible)
7. Unnecessary deep copies
8. Missing pagination for large datasets
9. Unoptimized regex patterns
10. Missing connection pooling

For each: estimated performance impact, benchmark suggestion,
and optimized implementation.

Code:
[PASTE CODE]
```

14 more quality prompts in the full toolkit: Architecture, Error handling, Type safety, Test coverage, API design, Database schema, Logging, Duplication, Naming, Async/Concurrency, Config, Migration safety, Accessibility, i18n…

SECTION 3: Language-Specific (showing 1 of 10)

Prompt #31 — Python: Pythonic Code Review

```
Review this Python code for Pythonic best practices:

1. Using loops instead of list comprehensions
2. Not using context managers (with statement)
3. Mutable default arguments
```

```
4. Using type() instead of isinstance()
5. Not using f-strings (Python 3.6+)
6. Missing type hints (PEP 484)
7. Not using pathlib for file operations
8. Missing dataclasses or NamedTuple
9. Using bare except instead of specific exceptions
10. Not using enumerate() for indexed iteration

For each: the non-Pythonic pattern, why it's a problem,
and the Pythonic version.

Code:
[PASTE CODE]
```

9 more language prompts in the full toolkit: TypeScript, Rust, Go, Solidity, React, SQL, Swift/Kotlin, C++, Shell...

SECTION 4: Automation Scripts (showing 1 of 10)

Script #1 — AI Pre-Commit Hook

```bash
#!/bin/bash
# AI Pre-Commit Hook
# Install: cp this to .git/hooks/pre-commit && chmod +x

set -euo pipefail

STAGED=$(git diff --cached --name-only --diff-filter=ACM | \
  grep -E '\.(py|js|ts|go|rs|sol)$' || true)
[ -z "$STAGED" ] && exit 0

echo "AI reviewing staged changes..."
DIFF=$(git diff --cached)

REVIEW=$(echo "$DIFF" | claude -p "Review this diff for security
issues and bugs. Only flag High or Critical. Format:
FILE:LINE -- ISSUE -- FIX. If clean, say LGTM." 2>/dev/null \
  || echo "LGTM")

if echo "$REVIEW" | grep -qi "critical\|high"; then
    echo "AI found issues:"
    echo "$REVIEW"
    read -p "Commit anyway? (y/N) " -n 1 -r
    echo
    [[ $REPLY =~ ^[Yy]$ ]] || exit 1
else
    echo "AI review: $REVIEW"
```

```
fi
```

9 more scripts in the full toolkit: PR reviewer, Secret scanner, Batch review, Dependency monitor, Diff review, Test generator, Security headers check, Codebase documenter, Multi-model review…

Get the Full Toolkit

50 prompts + 10 scripts for $9

https://tabdullah8.gumroad.com/l/ai-code-review-toolkit

Section	Preview	Full
Security Prompts	2	15
Quality Prompts	1	15
Language Prompts	1	10
Automation Scripts	1	10
Total	**5**	**50 + 10**

100% money-back guarantee.

Try our free online tool: https://nopii.xyz/review.html

GitHub Action: https://github.com/ETwithin/ai-code-review-action

Free samples on GitHub: https://github.com/ETwithin/ai-code-review-prompts