Suhas Bhairav

AI Pull Request Code Review Rules Setup

Automate basic code checks while preserving human oversight and architectural integrity.

Set up an AI-assisted PR reviewer to catch bugs, style issues, and architecture violations before merge.

Suhas Bhairav

suhasbhairav.com

What to Automate in PR Review

Define the automation boundaries for the AI reviewer. Distinguish automated checks from human decisions.

Key move **Assign an automation owner to maintain rules and update thresholds; avoid scope creep.**

PRACTICAL CHECKLIST

01	Automate basic syntax and style checks using language-specific linters.
02	Detect obvious bugs like null dereferences, unreachable code, and simple off-by-one errors.
03	Enforce architecture conformance (layer boundaries, prohibited dependencies, and module coupling risks).
04	Flag security signals (hard-coded secrets, unsafe eval, insecure network calls).
05	Identify performance red flags (unbounded loops, expensive queries) detected by static analysis.
06	Require human review for complex algorithms, edge cases, and performance trade-offs.
07	Automated data needs: repo metadata, test results, code owners, historical failures.

Suhas Bhairav AI Resource

Set up an AI-assisted PR reviewer to catch bugs, style issues, and architecture violations before merge.

Data, Tools, and Environment

Outline the data inputs and tooling required to run AI reviewer.

Key move **Never log raw PR diffs; redact secrets to protect privacy.**

PRACTICAL CHECKLIST

01	Data inputs: PR diff, file paths, language, tests, and build results.
02	Tooling: CI pipeline, AI reviewer model, and linter plugins.
03	Environment: containerized runtimes, ephemeral workers, rate limiting.
04	Output: annotated comments, risk scores, and suggested fixes.
05	Access controls: least privilege, secrets management, audit logs.
06	Feedback loop: persist reviewer decisions for iterative improvement.
07	Guard privacy: sanitize outputs and restrict data exposure in logs.

Suhas Bhairav AI Resource

Set up an AI-assisted PR reviewer to catch bugs, style issues, and architecture violations before merge.

1–6 Week Adoption Roadmap

This weekly plan translates policy into an actionable, testable program. Each increment reduces manual review time and builds confidence.

Key move **Run weekly retrospectives to prune false positives.**

PRACTICAL CHECKLIST

01	Week 1: define policy, select initial languages, and configure baseline CI integration.
02	Week 2: enable lint/style checks; ship automated comments for trivial issues.
03	Week 3: add basic bug patterns and architecture checks; tune thresholds.
04	Week 4: set gating rules; require human review for flagged items.
05	Week 5: expand checks to tests, security signals, and dependency rules.
06	Week 6: measure impact, collect feedback, adjust rules, appoint ownership.

Suhas Bhairav AI Resource

Set up an AI-assisted PR reviewer to catch bugs, style issues, and architecture violations before merge.

ABOUT THE CREATOR

Suhas Bhairav

I build AI systems that combine distributed architecture, retrieval, knowledge graphs, security, and practical workflow design. My focus is AI that can be used repeatedly by real teams: understandable, reliable, and grounded in the work people already do.

EXPERTISE

- Principal systems architecture and applied AI engineering
- Production-grade AI systems, agentic workflows, RAG, and knowledge graphs
- Full-stack workflow applications across AWS, Google Cloud, Azure, on-premise, and hybrid environments
- Practical AI implementation patterns for automotive, manufacturing, finance, logistics, cybersecurity, sales, and customer operations
- Security, governance, observability, approvals, and failure-mode thinking for operational AI systems

WEBSITE

suhasbhairav.com

EMAIL

suhasbhairav@gmail.com