ChatGPT

AI Coding Agents for CI/CD in GitHub Environments

Modern software teams are increasingly leveraging **AI-powered coding agents** to automate code generation, code review, testing, and bug fixing tasks as part of their CI/CD pipelines. These tools augment or complement services like OpenAI's Codex (the model behind GitHub Copilot) by integrating directly into GitHub via Actions, Apps, or bots. In this report, we survey notable solutions – prioritizing those that work in GitHub workflows – and evaluate their capabilities for security scanning, mobile development (Android/iOS) support, and automatic CI failure fixes. We include open-source/free options alongside high-value commercial tools, and focus on a tech stack spanning Node.js, MongoDB, PostgreSQL, Kotlin, Swift, and Python. The goal is to help identify AI assistants suitable for individual GitHub accounts within an organization, highlighting how they integrate, what languages they support, and their strengths/limitations.

AI Integration in GitHub Workflows

GitHub's own AI offerings set the baseline. **GitHub Copilot** (powered by Codex/GPT models) began as an autocomplete tool, but has evolved into a "full AI coding assistant" that can *"run multi-step workflows, fix failing tests, review pull requests, and ship code"* directly from VS Code or the GitHub interface [1]. For instance, Copilot can generate unit tests on demand and even suggest patches for failing CI tests or for security alerts. GitHub has introduced a **Copilot PR agent** which automatically reviews pull requests for issues (like bugs or inefficient code) and suggests improvements [2]. It supports a wide range of languages and frameworks – including mobile stacks (e.g. React Native, Flutter) – and integrates with GitHub's workflow seamlessly [3]. Another GitHub-native capability is **Copilot's Code Scanning Autofix**, which uses an OpenAI GPT-4.1 model to propose fixes for security vulnerabilities identified by CodeQL scans [4] [5]. This means if your repository is scanned for security issues, Copilot can auto-generate a patch to remediate certain alerts (available on public repos or for organizations with GitHub Advanced Security) [6]. Copilot's strengths are deep integration and multi-language support, though it is a proprietary paid service (with free trials and free access for certain users). Its limitations include potential **AI hallucinations** and the need for human validation of suggested changes (as with any LLM-based tool).

Beyond Copilot, OpenAI has also released a **Codex CLI and GitHub Action** to directly embed Codex into CI flows. One notable use-case is automatically fixing broken builds: OpenAI's reference workflow demonstrates that when a CI job fails, a follow-up job can *invoke the Codex CLI* to analyze the failure, apply a minimal code change, and open a pull request with the fix [7] [8]. **Figure 1** below illustrates this auto-fix process. This kind of "self-healing CI" agent can keep the build green with minimal human intervention. It requires providing an OpenAI API key and crafting prompts for the Codex agent, but it's powerful for quick fixes of test failures or small bugs.

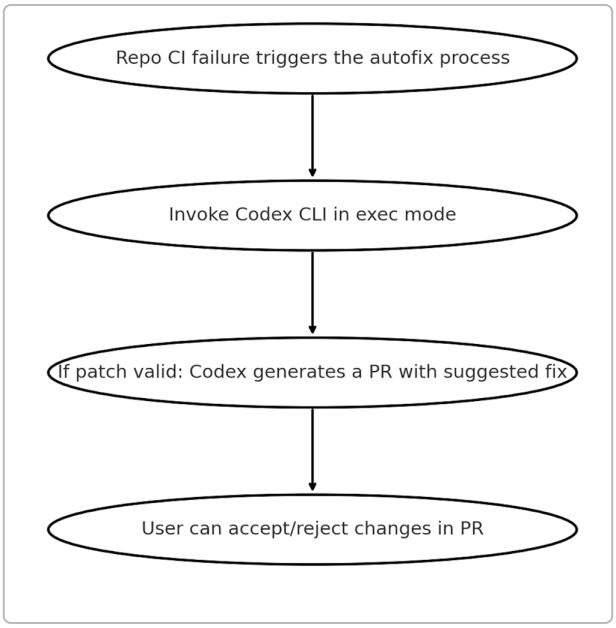

Figure 1: Example CI auto-fix workflow using OpenAI Codex. When a CI job fails, a GitHub Action triggers the Codex CLI in exec mode to propose a targeted code change and opens a pull request with the suggested fix [7]. The team can then review and merge the AI-generated patch.

While Copilot and Codex focus on in-editor assistance and basic CI auto-fixes, there are **many other AI coding agents** that integrate with GitHub to bolster code quality and automate development tasks. Below, we highlight several notable tools and their features:

- **AI Review (Nikita-Filonov/ai-review)** – *Open-source* GitHub Action and CLI that performs AI-driven code reviews. AI Review runs in CI pipelines and posts inline PR comments and summary reports automatically [9]. It supports multiple LLM backends (OpenAI GPT-4, Anthropic Claude, Google Gemini, local models via Ollama, etc.) and works with GitHub, GitLab, Bitbucket, Azure DevOps, etc. [10]. You can customize its review prompts to enforce your team's guidelines. Strengths: runs fully client-side (no code is sent to third-party servers except the LLM API), highly configurable, and free (Apache-2.0 licensed). Limitations: It doesn't automatically fix code – it only comments – and the usefulness depends on the chosen LLM's quality and context length (you must provide an API key or self-host a model).

- **Aider (aider-ai/aider)** – *Open-source* AI pair-programming agent that can be scripted into CI. Aider is a CLI tool that lets an AI assistant directly edit your codebase. It supports over 100+ languages and frameworks (including Python, JavaScript, Go, Rust, Kotlin, Swift, etc.) [11] [12]. Uniquely, Aider can automatically run tests or linters after making changes, and if failures are detected it will attempt to *fix those problems and re-run* – essentially an iterative loop until tests pass [13]. This makes it suitable for auto-fixing failing CI tests or lint issues. It integrates with Git via committing changes with messages, and a community GitHub Action (e.g. `mirrajabi/aider-github-action`) exists to trigger Aider on demand. Strengths: completely free and self-hostable, can utilize local models or cloud APIs, and can handle non-trivial multi-file edits. It supports all major languages in our stack (Node.js/JS, Python, Kotlin, Swift, etc.). Limitations: requires careful prompt scripting for autonomous use (to avoid going off-track), and for security review it relies on the LLM's general knowledge (no

specialized vulnerability DB). Also, using cloud LLMs in an automated loop can be costly if not monitored.

- **CodeRabbit** – *Commercial (free tier available)* AI code review bot available as a GitHub App and IDE extension. CodeRabbit uses GPT-3.5/4 models to provide *comprehensive PR analysis*, commenting on potential bugs, performance issues, and stylistic improvements [14]. It auto-generates summaries of code changes and can learn from your team's feedback to improve its suggestions over time. Notably, CodeRabbit combines traditional static analysis (linters, security scanners) with generative AI reasoning to reduce noise [15]. It supports real-time analysis of code as you write or open a PR, and has broad language support (frontend frameworks like React/Vue/Angular [16], as well as backend languages). For example, it will detect security issues or code smells and provide one-click fix suggestions in the PR interface [17]. Strengths: Easy GitHub integration (claims to be one of the most-installed AI apps on GH), team-specific learning, and a *"full context"* analysis across your repo. It also explicitly supports security scanning and even generates sequence diagrams or release notes from PRs [15] [18]. Limitations: It's a hosted solution (your code is analyzed on their servers), so you have to be comfortable with their privacy and data policies. Also, while a free tier exists [19], advanced features (or higher usage) require a subscription.

- **Tabnine Code Review Agent** – *Commercial (enterprise-focused)* AI assistant from Tabnine, known for code completions. Tabnine's Code Review Agent integrates via a GitHub Action (`codota/tabnine-pr-agent`) to automatically analyze pull requests using Tabnine's models [20]. It enforces best practices by checking code against a set of rules (including custom team rules) and flagging policy violations or bugs [21] [22]. It supports a wide array of languages (Tabnine's models are trained on many languages, including Java, Python, JavaScript, Kotlin, Swift, C#, etc.). A key feature is that teams can define *plain-language rules* for code standards, which the agent converts to checks during PR review [22]. It provides suggestions to fix any issues it finds. Strengths: Can run on-premises or self-hosted (addressing privacy), learns team coding patterns for tailored feedback [23] [24]. Also, Tabnine's solution emphasizes **compliance** (e.g., ensuring no secrets or license issues in code) and can autonomously apply certain fixes in a guided manner [21]. Limitations: It is mainly a paid enterprise tool (no broad free tier for individuals, aside from limited trials). Setup may be more involved (personal access token and possibly running a Tabnine engine). Its "AI" is also less of a generative LLM and more of a trained model on code patterns (good for structured suggestions, but perhaps less flexible than GPT-4 in understanding novel context).

- **Amazon CodeGuru Reviewer** – *Commercial (AWS service)* that uses ML to comment on code review diffs. It integrates with GitHub via a native **GitHub App** or through GitHub Actions, analyzing PRs for issues. CodeGuru specializes in **performance and security** findings: for example, spotting inefficient Java/Python code or AWS API misuse, and detecting common vulnerabilities. It currently supports Java, Python, and JavaScript (for security detectors) – which covers Node.js to an extent – but does *not* support Kotlin or Swift as of now (so it's less useful for mobile code) [25] [26]. Strengths: it provides very specific recommendations drawn from Amazon's internal best practices (e.g., identifying expensive database calls or concurrency risks) [27]. It also offers a complementary **CodeGuru Security** module (recently launched) focusing on static application security testing for Java/Python/JS [25]. CodeGuru can be set to run automatically on each PR and will add comments with findings and links to documentation on how to fix them. Limitations: Limited language support (no native support for Kotlin/Swift/PHP, etc.), and it is a paid service charged per lines of code

analyzed (pay-as-you-go) [28]. It tends to find fewer but high-confidence issues, so it might miss more subtle problems that a generalized LLM could discuss.

- **DeepCode (Snyk Code)** – *Commercial (free for open source)* AI-powered static analysis. DeepCode was an AI code review startup (using machine learning trained on millions of commits) that is now part of Snyk [29]. Snyk Code (DeepCode) integrates into GitHub via the Snyk GitHub Action or App to scan code for bugs and security issues on each push/PR. It supports **11+ programming languages** including JavaScript/TypeScript, Python, Java/Kotlin, C#, PHP, Ruby, Go, as well as mobile languages like Swift and Objective-C [30] [31]. A highlight is its *one-click autofix* suggestions for certain issues [32] – for example, it can suggest the code change to remediate a vulnerable code pattern. It also catches dependency vulnerabilities and code quality issues. Strengths: Strong **security scanning** focus (with a constantly updated vulnerability database and rules engine), and wide language coverage (including Android/iOS code scanning) [30]. It integrates easily with CI pipelines and can fail the build on high-severity issues, or open fix pull requests for dependency upgrades. Snyk offers a free tier (especially for open-source projects or small teams) [33]. Limitations: Being primarily a static analyzer, it may flag false positives or lack context that an LLM might consider; however, its ML-based engine is more precise than basic linters. It doesn't generate new code beyond small fixes. Also, using the full features (like Snyk's automated fix PRs for code issues) may require a paid plan in enterprise settings.

- **Refact.ai** – *Open-source core (with cloud service)*, an **autonomous AI coding agent** that can be self-hosted for privacy [34] [35]. Refact.ai acts like a "AI developer" that can plan and execute code changes across a project. It has an IDE plugin and CLI, and can integrate with GitHub and CI pipelines via API. Notably, Refact emphasizes on-premise deployment (the AI model and agent can run on your hardware or VPC) for data confidentiality [36] [35]. It supports many languages (Python, JS/TS, Go, Java, Kotlin, Swift, etc.) and includes specialized analysis for mobile app security and code quality [37]. For example, it can detect Android or iOS specific issues (like insecure Android component usage) as part of its rules. The agent can not only review code but also generate substantial code bases from scratch given instructions (it was a top performer on a recent software engineering benchmark) and apply fixes autonomously. Strengths: Highly customizable and **extensible agent** – you can fine-tune it to your codebase and even choose which underlying LLM to use (it supports plugging in models from Anthropic, OpenAI, Meta, etc. or local models) [38] [39]. It can connect to external tools (databases, documentation, web) for additional context when coding [40]. Limitations: While the core is open-source, running your own instance with large models requires infrastructure and expertise. The cloud SaaS version has a free plan, but advanced usage might be limited unless you pay. Also, an autonomous agent executing code changes needs careful constraints to avoid going out of scope – in critical projects you'd still do a manual review of its output.

- **Qodo – PR-Agent / Qodo AI Platform** – *Hybrid (open-source AGPL version + commercial platform)*. Qodo's PR-Agent was one of the first open-source GPT-4 code review bots (with thousands of stars on GitHub) that could analyze pull requests and provide feedback [41]. The open-source "PR Agent" can still be self-hosted as a GitHub Action to get basic GPT-based PR reviews (it's limited to public info and some generic rules). Qodo, the company, has since launched a more advanced platform that adds numerous features: **automatic unit test generation** for pull requests, code coverage analysis, issue ticket linking, "PR chat" (you can ask follow-up questions to an AI about the PR), and enforcement of organizational policies [42] [43]. Qodo's agent can also auto-fix simple issues – e.g., if

a coding standard is violated, it might suggest a direct code change – and it validates that new code has appropriate tests and documentation [44] [45]. It integrates as a GitHub App and also offers IDE plugins for local review. Strengths: **Enterprise compliance** focus – Qodo can ensure every PR adheres to security rules, licensing guidelines, formatting, etc. (it even checks for things like missing JIRA issue IDs in commits or evaluates the impact of a change) [46] [47]. It supports all major languages, including our target stack (they advertise support for 15+ frameworks and languages, plus front-end and mobile frameworks) [48]. The base tool is free for developers (and free for open-source projects, with their paid tier being free if your project is OSS) [49]. Limitations: The latest features (like "agentic workflows" and context engine) are part of their SaaS platform – using them means relying on Qodo's service and possibly sending code to their cloud. The open-source PR-Agent, while free, is now "legacy" and not as smart or maintained as the commercial version [50]. Also, as with others, the AI suggestions need human judgment to accept or refine.

Each of these tools brings a different mix of capabilities. For example, some are stronger in **security scanning** (Snyk/DeepCode, SonarQube – which can be used in tandem with AI agents) while others excel at **mobile development support** (Copilot's broad knowledge, Refact's specialized mobile rules, Snyk's Swift/Kotlin scanning). Tools like Codex CLI or Aider provide **automatic fixing** of CI failures, whereas code review bots (AI Review, CodeRabbit, Qodo) provide conversational feedback and ensure quality gates are met. Many organizations combine multiple solutions – for instance, using static analyzers (CodeQL, Sonar, Snyk) to catch critical security issues and an AI reviewer (Copilot or AI Review) to handle style and logic suggestions, with perhaps an agent like Codex or Aider to propose actual code patches for failing tests. The table below summarizes the key comparisons of the identified AI coding agents:

Comparison Table of Selected AI Coding Agents

Name	**Core Capabilities** (What it does)	**Integration** (GitHub workflow)	**Languages / Frameworks**	**Strengths**	**Limitations**	**Licensing**
GitHub Copilot	Code generation (autocomplete), PR review suggestions, test generation, and code scanning auto-fixes [1] [2].	IDE extension; GitHub PR UI and code-scanning integration.	~20+ languages (JavaScript/Node, Python, Java/Kotlin, Swift, etc.) [51] [52]; frameworks like React Native, Flutter [51].	Seamless GitHub/VSC integration; powerful GPT-4-based suggestions; multi-language & mobile framework support.	Proprietary (closed source); requires subscription (free for some users); suggestions may need vetting for accuracy.	*Proprietary* – Paid plans (Free trial; $10/mo indiv.) [53].

Name	**Core Capabilities** (What it does)	**Integration** (GitHub workflow)	**Languages / Frameworks**	**Strengths**	**Limitations**	**Licensing**
OpenAI Codex CLI	Auto-fixes failing CI builds by generating minimal code patches and opening PRs [7] [8]; can also generate code via CLI prompts.	GitHub Action (`openai/codex-action`) triggered on CI failure [7].	Any language (model-agnostic) – shown with Node/Jest example, but works with Python, etc. via prompt [8].	Directly automates CI remediation (self-healing builds); leverages OpenAI's latest Codex/GPT models for code understanding.	Needs careful prompt tuning and an OpenAI API key; may not handle large codebases due to context limits; costs API credits per use.	*Proprietary API* – Codex CLI is free to use, but OpenAI API usage is billed.
AI Review (open-source)	AI-powered code **review** – posts inline PR comments, summaries, and even replies in discussions [9]. Highly configurable prompts and multi-LLM support.	GitHub Action (also supports GitLab, Bitbucket, etc.) runs in CI [9].	Virtually all languages (uses LLMs like GPT-4/ Claude that handle any language code) – tested on common languages (JS, Python, Java, etc.).	Open-source (Apache-2.0); works across platforms; can use local or hosted LLMs [10]; keeps data private (runs client-side).	Review-only (does not modify code); quality depends on chosen LLM and context size; requires supplying an API key or model.	*Open-source* (Apache-2.0); Free to self-host (API costs for LLM).
Aider (open-source)	AI pair-programmer that **edits code** on request. Can iteratively fix code until tests and linters pass [13]; generates new code or refactors per user prompts.	CLI tool (developer runs locally or in CI via scripts); GitHub Action available for automation.	100+ languages (broadly supports Python, JS/ Node, Go, Rust, Kotlin, Swift, etc.) [11].	Fully scriptable and automatable; automatically tests and fixes code issues in a loop [13]; can integrate with git for commit history.	Not a turnkey GitHub App – setup and prompts require engineering effort; without guardrails the AI might make unintended changes; depends on external LLMs for intelligence.	*Open-source* (Apache-2.0); Free (choose your own LLM API or local model).

Name	Core Capabilities (What it does)	Integration (GitHub workflow)	Languages / Frameworks	Strengths	Limitations	Licensing
CodeRabbit	Automated **pull request review** with GPT-3.5/4: generates PR summaries, comments on bugs & improvements, suggests one-click fixes [14] [17]. Also runs integrated static analysis and security checks.	GitHub App (install to repos); also VS Code/ JetBrains plugins for in-IDE review.	Many languages & frameworks – confirmed support for JS/TS (React, Vue, Angular) [16], Python, Go, Java, etc. (adapts to team stack).	High adoption and mature UI; combines AI with linters for high signal [15]; learns from team feedback to reduce false positives; free tier available [19].	Code is processed on vendor servers (privacy considerations); commercial features beyond basic review; may occasionally misjudge context (needs human final say).	*Commercial SaaS* – Free basic plan [19]; paid plans for advanced use.
Tabnine Code Review	AI **code analysis** agent enforcing best practices. Analyzes PRs for code quality, coding standard violations, and potential bugs, providing fix suggestions. Learns team-specific rules [22].	GitHub Action (`tabnine-pr-agent`) for PR events [20]; also IDE plugin for pre-commit review.	Wide language support (Tabnine supports Java, Python, JS/TS, C#, Go, Ruby, PHP, plus **Kotlin, Swift** via JVM and LLVM support).	On-premises/ self-host deployment option (keeps code private) [23]; team-tailored model improves over time [24]; can codify custom review rules in natural language [22].	Mainly targeted at enterprises – setup and custom rule training can be involved; not focused on generating new code, only reviewing existing code; requires Tabnine license for full use.	*Commercial* – Proprietary (Enterprise licenses; Tabnine Pro for individuals with limited features).
Amazon CodeGuru	ML-driven **automated code review** – finds issues in code (duplicate code, resource leaks, AWS API misuse) and gives recommendations. Also performs security analysis for common vulns.	GitHub App or GitHub Actions workflow (analyzes PRs or codebase on push).	**Java & Python** primarily (full support) [54] [55]; limited JavaScript support for security rules. *Does not support Kotlin/Swift.*	Excellent at performance optimizations and AWS-specific guidance [27]; integrates with AWS CI/CD (CodePipeline) and GitHub; low false-positive rate due to targeted rules.	Limited language scope (no native Node, Kotlin, Swift analysis); suggestions can be basic for non-AWS code; usage costs can accumulate (pay per LOC analyzed) [28].	*Commercial AWS service* – Pay-as-you-go pricing [28] (free trial available).

Name	**Core Capabilities** (What it does)	**Integration** (GitHub workflow)	**Languages / Frameworks**	**Strengths**	**Limitations**	**Licensing**
Snyk Code (DeepCode)	AI-enhanced **static code analysis** for bugs and security flaws. Scans code in real-time and provides instant feedback, including vulnerable code patterns and potential fixes [29] [32].	GitHub App or Action (Snyk scans on PRs/ commits; results shown in PR checks and Snyk UI).	Supports **11+ languages**: JavaScript/ TypeScript, Python, Java/ Kotlin, C#, PHP, Ruby, Go, **Swift/ Obj-C** (mobile), etc. [30] [31].	Strong security focus (up-to-date vuln database); *autofix* suggestions for certain issues [32]; tight CI integration (fail builds on findings, suggest PRs to fix deps); free for open-source projects.	Not a generative code writer – only flags issues; some findings need developer judgment to fix properly; enterprise features (like extensive reporting) require paid plan.	*Commercial (SaaS)* – Free tier for OSS and limited use [33]; paid plans for teams.
Refact.ai	**Autonomous AI agent** that can write, refactor, and review code with minimal human input. Plans multi-step tasks, searches the codebase, and applies changes to meet high-level objectives [56] [38]. Also provides in-IDE AI chat and completions.	IDE integration (VS Code, etc.) for live agent; can be invoked in CI via API/CLI (for automated coding tasks or PR review).	Multi-language (designed for Python, JS, Go, Java, C#, etc. with project-wide context); supports **mobile (Swift, Kotlin)** and even DB/ schema interactions (via plug-ins).	Open-source core with self-host option (you can deploy on-prem for privacy) [57]; very **context-aware** (indexes entire repo, docs, DB for context) [38] [40]; can execute complex fixes (beyond linter scope).	Complexity – an autonomous agent can produce incorrect changes if misconfigured; requires significant resources for large models; the cloud service for Refact has usage limits on free plan.	*Open-source* (Agent is OSS); Cloud service has *Free* tier and paid upgrades [58] [59].

Name	Core Capabilities (What it does)	Integration (GitHub workflow)	Languages / Frameworks	Strengths	Limitations	Licensing
Qodo (PR-Agent)	**AI code review and quality enforcement**. Reviews PRs with GPT-based analysis, auto-generates **unit tests** for new code [45], ensures compliance with security/coding standards, and offers an interactive PR chatbot. Can provide code explanations and even directly suggest code changes.	GitHub App for PR reviews; also offers CLI and IDE plugins for local use [60] [61].	Many languages (officially supports 15+ including JavaScript/ TypeScript, Python, Java, C#, plus **Kotlin, Swift** and front-end frameworks) [48].	Advanced "shift-left" capabilities (catches missing tests or design issues early) [47]; highly customizable to team's standards (plain English policies enforced) [62]; free for individual devs and open-source projects (paid tiers add more features) [49].	The most powerful features (e.g. deep context engine, certain automations) are available only in the paid platform; open-source version of PR-Agent is basic and no longer actively expanded [50]; as with others, AI recommendations must be reviewed for safety.	*Hybrid*: Open-source AGPL-3.0 core [63]; **Commercial** SaaS (Qodo platform) with free developer tier [64] and paid plans for teams.

Sources: The information above is compiled from documentation and reports on each tool. Key references include GitHub's official blog and docs for Copilot [1] [2] and Copilot Autofix [6], OpenAI's developer guide for Codex CI integration [7], the AI Review GitHub README [9], Aider's documentation [13], DigitalOcean's 2025 roundup of AI review tools (covering Copilot, CodeRabbit, DeepCode, etc.) [29] [27], Tabnine's PR agent usage notes [20], Amazon's CodeGuru FAQs [54], Snyk's supported languages docs [31] [30], Refact.ai and Qodo product pages [37] [44], among others. These sources are cited inline for accuracy and can be consulted for further details on each solution.

---

[1] GitHub Copilot tutorial: How to build, test, review, and ship code faster (with real prompts) - The GitHub Blog
https://github.blog/ai-and-ml/github-copilot/a-developers-guide-to-writing-debugging-reviewing-and-shipping-code-faster-with-github-copilot/

[2] [3] [14] [16] [19] [27] [28] [29] [32] [33] [37] [44] [45] [48] [51] [52] [53] [58] [59] [64] 10 AI Code Review Tools That Find Bugs & Flaws in 2025 | DigitalOcean
https://www.digitalocean.com/resources/articles/ai-code-review-tools

[4] [5] [6] Responsible use of Copilot Autofix for code scanning - GitHub Docs
https://docs.github.com/en/code-security/code-scanning/managing-code-scanning-alerts/responsible-use-autofix-code-scanning

[7] [8] Autofix CI failures with Codex
https://developers.openai.com/codex/autofix-ci/

9 10 GitHub - Nikita-Filonov/ai-review: AI-powered code review tool for GitHub, GitLab, Bitbucket Cloud, Bitbucket Server, Azure DevOps and Gitea — built with LLMs like OpenAI, Claude, Gemini, Ollama, and OpenRouter
https://github.com/Nikita-Filonov/ai-review

11 12 13 GitHub - Aider-AI/aider: aider is AI pair programming in your terminal
https://github.com/Aider-AI/aider

15 17 18 AI Code Reviews | CodeRabbit | Try for Free
https://www.coderabbit.ai/

20 GitHub - codota/tabnine-pr-agent: This repository contains the tabnine PR agent
https://github.com/codota/tabnine-pr-agent

21 22 62 Unveiling Tabnine's Code Review Agent: Improving quality, security, and compliance uniquely for every development team - Tabnine
https://www.tabnine.com/blog/unveiling-tabnines-code-review-agent/

23 24 Announcing Tabnine GitHub integration - Tabnine
https://www.tabnine.com/blog/announcing-tabnine-github-integration/

25 26 55 Amazon CodeGuru Reviewer: already time for retirement?
https://pcg.io/insights/amazon-codeguru/

30 Snyk Vs Github Advanced Security Comparison - Aikido
https://www.aikido.dev/blog/snyk-vs-github-advanced-security

31 Swift and Objective-C | Snyk User Docs
https://docs.snyk.io/supported-languages/supported-languages-list/swift-and-objective-c

34 35 36 38 39 40 56 57 AI Coding Agent for Software Development - Refact.ai - Refact.ai
http://Refact.ai

41 42 43 46 49 50 63 GitHub - qodo-ai/pr-agent: PR-Agent: An AI-Powered Tool for Automated Pull Request Analysis, Feedback, Suggestions and More!
https://github.com/qodo-ai/pr-agent

47 60 61 Qodo | AI Code Review That Meets Your Standards
https://www.qodo.ai/

54 Amazon CodeGuru Reviewer FAQs
https://aws.amazon.com/codeguru/reviewer/faqs/