apriorit

CASE STUDY

Building a Multi-Agent AI Platform for Automatic Code Review

AI & ML

Background

As a software development vendor, Apriorit regularly conducts code reviews as part of engineering processes. But Apriorit developers struggled with slow Bitbucket pull request workflows due to an excessive reliance on manual reviews, which significantly extended code review cycles and increased the engineering team's workload.

To remedy this, Apriorit experts from the AI and ML software development team proposed creating an internal AI-driven pull request code review platform.

The platform automates first-pass code reviews, using an integrated AI code analysis engine to enforce code quality in line with Apriorit's established standards, security requirements, and language-specific best practices. This allows for reducing pull request processing time by 30% and helps developers speed up code review by 60%.

The client

Apriorit specializes in cybersecurity engineering and the development of secure IT products for regulated industries, including healthcare and FinTech. The company provides a wide range of software engineering services, backed by deep expertise in kernel and driver development, reverse engineering, and AI-powered solution development.

Although this was Apriorit's internal project, our engineering team approached it with the same rigor as any client project, applying our own development practices and secure SDLC principles throughout. The project had clearly defined stakeholders, a dedicated team, approved business and technical requirements, and a detailed scope of work.

Our client:
Apriorit

Operation geography
USA, Canada, Poland, Cyprus, Ukraine

Industry:
Cybersecurity engineering
Software development
QA services

Solution we delivered:
AI-powered code review platform (with multiple agents)

Services we provided:
AI agent and multi-agent system development

The challenge

Apriorit's Bitbucket pull request workflows relied heavily on manual code review, forcing developers to repeatedly verify coding standards and spend significant time on repetitive checks, context switching, and routine comments. This slowed down code reviews, increased developers' cognitive load, and pulled developers away from complex engineering tasks.

Apriorit wanted to build an AI-powered platform to automate pull request code reviews, enhance code quality, and speed up the code review process. The main requirements for the solution were:

- Semantic code search to provide the AI agent with awareness of the entire codebase
- Analysis of kernel and low-level code
- Support for multi-language codebases, including languages such as Python, C++, C#, C, Swift, Rust, and Go
- Support for secure coding standards for C and C++ reviews

Project details

Having analyzed the initial requirements, we formed a dedicated project team and decided on the technology stack. For this project, we engaged experts in AI and ML software development, cybersecurity engineering, and Python solution development, along with a QA specialist, a DevOps specialist, and a project manager.

Apriorit team

- Python Backend Engineer
- AI/LLM Engineer
- Software Architect
- QA Specialist
- DevOps Specialist
- Project Manager

Tools and technologies

LANGUAGE

- Python

WEB FRAMEWORKS

- FastAPI
- Uvicorn

AI ORCHESTRATION

- Pydantic AI
- OpenCode CLI
- GitHub Copilot

LLM PROVIDERS

- OpenAI
- Anthropic

MCP

- ast-grep
- Think MCP Server

JOB QUEUE

- Procrastinate (PostgreSQL-backed async task queue)

DATABASE

- PostgreSQL

The result

We developed a custom multi-agent AI-based code review platform. The platform automatically reviews every pull request, generates inline comments with review findings, and delivers them to the Bitbucket interface for additional review by the developer.

This solution automates and scales code reviews by applying collaborative review patterns typically used by senior engineers.

The multi-agent AI system runs entirely within Apriorit's infrastructure, giving complete control over data.

How we did it

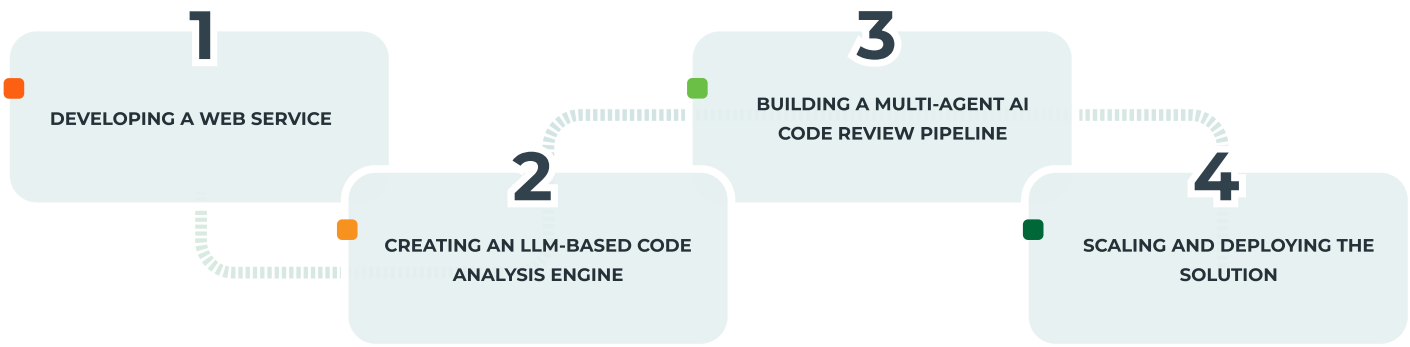
We started by selecting an appropriate architectural approach for the future Apriorit code review platform and chose an event-driven microservice architecture.

Then we focused on identifying the key components needed to ensure the solution meets all approved requirements:

- Bitbucket client
- FastAPI-based web service
- Procrastinate task queue (PostgreSQL-based)
- AI-powered code review pipeline
- LLM-based code analysis engine

After discussing and approving the product's architecture and component list with stakeholders, we proceeded to development.

Our team broke down the process of building a multi-agent AI system into the following four stages:



1. Developing a web service

We started by building a web service. This FastAPI-based webhook receiver:

- Receives Bitbucket pull request webhook events
- Validates Bitbucket webhook signatures
- Checks pull request eligibility (such as open/updated, within file count limits)
- Enqueues the review job to the Procrastinate task queue

To safeguard Bitbucket data, we implemented webhook signature validation and enforced TLS encryption for secure data transmission.

2. Creating an LLM-based code analysis engine

Once a web service was ready, we implemented a versioned system of AI agent personas that supported multiple programming language review prompts.

In this system, each AI agent is responsible for a specific stage of code review:

- Planning and structuring the code review process
- Reviewing code
- Retrieving additional code context (for example, dependencies) to more accurately evaluate the architectural impact of a file change
- Validating code review findings to reduce hallucinations and false positives
- Generating inline comments with code review findings

We built an orchestration layer that allows for running AI agents in a defined order and manages dependencies between code review stages. This ensures traceability and predictable behavior across the review process.

3. Building a multi-agent AI code review pipeline

The pipeline consumes queued review jobs from the Procrastinate task queue and processes them through the structured workflow.

The pipeline's workflow consists of the following steps:

- Cloning the target repository into an isolated workspace
- Extracting the changed files with diffs
- Retrieving project-specific rules
- Filtering the ignored files
- Running an AI review on every changed file through the code analysis engine
- Creating issues discovered during the code review with the exact code line number
- Posting inline comments with review findings to Bitbucket

4. Scaling and deploying the solution

Finally, to prevent Apriorit's platform from overwhelming repositories or exhausting the LLM API quota under high load, we introduced Redis-backed distributed semaphores to cap the number of concurrent reviews. When the limit of reviews is reached, review jobs automatically apply a backoff-and-retry mechanism. This helps to ensure platform stability without dropping requests.

To deploy the solution, we packaged it into a Docker Compose stack. This deployment supports shared volume mounts, health checks, and environment-based configuration for all services.

After launching the code review platform, we received feedback from Apriorit engineers who said that it enables faster issue detection during reviews and, where needed, proactively highlights development best practices to consider.

Impact

Our team created a solution that automatically reviews every qualifying pull request. The key benefits of implementing the multi-agent AI platform are:

- **60% reduction in the time developers spend on code reviews.** Catching bugs, security issues, and best practice violations before an expert engages in code review allows engineers to save time and focus on more complex tasks. Moreover, the solution reduces pull request processing time by 30%.

- **Enhanced code review accuracy.** With a multi-agent AI approach, dedicated agents retrieve additional code context and independently validate review findings. This allows the Apriorit team to improve the accuracy of code analysis and reduce hallucinations and false positives, resulting in more reliable code review outcomes.

- **One platform for diverse codebases.** By supporting multiple programming languages including Python, C++, C#, C, Go, Swift, and Rust, and by applying language-specific rules with tailored review logic, the platform enables code reviews across heterogeneous codebases.

Having proven that the AI-based platform delivers stable results in reducing manual code review effort, Apriorit is currently preparing to roll it out to clients.