GitLab

The enterprise guide to agentic AI in software development

Table of contents

Executive summary: The era of agentic AI

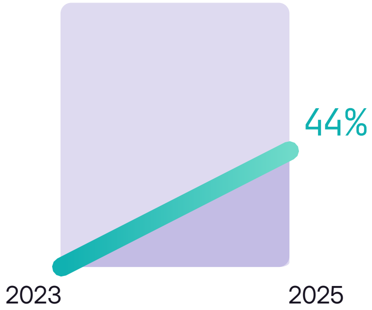
After two years of AI hype, reality is setting in. Executives estimate a **44% increase in revenue** due to the use of AI, yet developer satisfaction with AI tools is declining. Positive sentiment for AI tools decreased from over **70% in 2023 and 2024 to just 60% in 2025**.
The problem isn't AI itself — it's how we're implementing it.

AI is being deployed as a point solution, often focused on code suggestions, and doesn't have the context needed to account for all aspects of your environment. The tools are layered onto existing dysfunctions, such as tool sprawl, siloed teams, pre-existing technical debt, or understaffing.

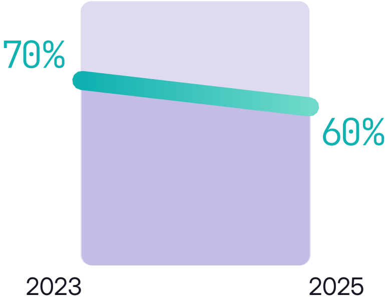
When AI tools are added to already fragmented workflows, they amplify rather than solve underlying organizational problems. The initial enthusiasm around AI is giving way to frustration as teams struggle with integration challenges and unrealistic expectations.

These integration issues point to a deeper need: AI that can work with existing systems rather than adding to their complexity. Agentic AI represents a fundamental shift that addresses these core integration limitations. Unlike generative AI tools that function as standalone point solutions, agentic AI-native platforms bridge system gaps by orchestrating workflows across your technology stack. When agentic AI has access to your entire environment, it has the context to make autonomous decisions, optimize your existing systems, and deliver higher-quality results than AI tool predecessors.

Executive estimated increase in revenue



Developer positive sentiment for AI tools



While agentic AI won't solve every organizational challenge, it uniquely addresses the integration and workflow problems currently plaguing AI point solutions.

Agentic AI-native platforms offer measurable opportunities to realize productivity while enhancing your workflows. In this paper, you'll learn more about agentic AI-native platforms, how they can accelerate your development lifecycles, and drive your organization's overarching goals.

The state of enterprise development: Why many AI solutions fall short

Key insight: Experienced developers take 19% longer when using AI coding tools.

Let's examine the disconnect between the perception of AI as a universal productivity solution and the reality of implementing AI point solutions in everyday work.

The productivity crisis in modern engineering

There's increasing pressure for developers to do more, faster, and it's not just coding. Developers also handle security testing, compliance tracking, quality assurance, documentation, planning, and more. In fact, the majority of their time isn't spent coding at all. It's dedicated to mundane, repetitive tasks that pull them out of their workflows.

On average, developers are interrupted **13 times per hour**, negatively impacting both productivity and job satisfaction. After only 20 minutes of interrupted tasks, **a study by the University of Irvine** found that people experienced significantly higher stress, frustration, and pressure.

Over time, this continuous pressure leads to burnout and turnover. For the organization, this results in significant setbacks. On average, it takes **35 days** to find and hire a software engineer and that doesn't include the time it takes for the new hire to **onboard and contribute meaningful work**. All in all, you're looking at months of productivity loss.

Why coding assistants don't solve the core problem

Teams adopt AI coding assistants to enhance productivity, but these tools don't address the root of the problem. **While AI coding assistants enhance individual coding tasks, they fail to address overarching workflow issues.** The isolated tools lack integration with your software development environments or workflows, so the tools don't have the full context of your team's projects, business logic, or organizational standards, leading to significant manual adjustments.

Trouble with tool sprawl

Enterprise organizations have approximately **254 tools with the IT department managing 61 tools**. Many of these tools have overlapping functionality creating redundancies, driving up unnecessary costs, and adding to complex workflows. **Sixty percent of employees** find it difficult to obtain the information they need to do their job, spending on average 5.3 hours per week waiting for information.

With so many plugins or APIs required to make the toolchain operable, developers end up spending more time searching for information, updating APIs, and maintaining integrations than building new features.

The AI point solution challenge

AI vendors may market their solutions as "plug-and-play," but in reality, it often involves complex data preparation, model training, and ongoing finetuning. For an AI point solution to be effective, it needs its own data pipeline, but that creates additional silos. Plus, teams must dedicate time to cleaning and formatting data. What was supposed to be a time-saver, becomes another maintenance burden.

In addition to maintenance, understanding business outcomes becomes a challenge. With a smattering of AI tools, pinpointing the ROI of an individual tool becomes almost impossible. When each tool has its own usage metrics, data formats, and definitions of success, you end up with fragmented data that isn't easily interpreted into business insights.

The security-speed dilemma

With cloud-native architectures, microservices, APIs, and third-party integrations, security teams already have a hard time keeping pace with the speed and complexity of modern development.

Now that organizations are layering on multiple AI solutions, there's an even greater volume of code that needs to be secured. This significantly larger attack surface, and the added challenge of governing more tools, puts security teams in a predicament. With **one security team member** for every 80 developers, security teams don't have the manpower or resources to thoroughly vet tools and code without creating significant bottlenecks.

The plethora of AI tools leads to a host of security concerns:

- ✓ **Governance fragmentation:** Security teams must track and secure dozens of disparate AI tools instead of one governed platform
- ✓ **Declining code scrutiny:** **As developers become comfortable with AI-generated code**, manual security reviews decrease, creating gaps in code quality assurance
- ✓ **Expanded attack surface:** Each AI tool integration creates new vulnerabilities between systems
- ✓ **Data handling complexity:** Multiple tools with different data retention, privacy, and access policies increase compliance risk

Avoiding AI isn't the answer to these challenges — **76% of developers already use AI or plan to**.

Instead of pushing against AI usage and causing developers to become shadow AI users, security teams need a proactive approach that gives developers the AI capabilities they want while maintaining the security controls the business requires.

The rise of agentic AI

Key insight: 89% of executives agree, "Agentic AI will be the industry standard for software development in less than 3 years' time."

What is agentic AI?

Agentic AI represents the next step in human and AI collaboration, allowing teams to leverage autonomous AI systems that can plan, execute, and adapt without human guidance. These goal-oriented, context-aware, and self-correcting agents, solve many of the challenges with AI point solutions. To better understand agentic AI, it's helpful to think of it in comparison to other AI formats, such as traditional AI assistants and context-aware assistants.

Traditional AI assistants

This reactive form of AI only responds when prompted. The tool processes the request and returns a response. There are often separate AI tools for different functions, such as code generation, testing, security scanning, or documentation. Development teams spend significant time coordinating between tools and resolving conflicts.

Example: A developer asks a code completion tool to generate a function. The tool provides code suggestions but has no awareness of whether the code integrates properly with existing systems, follows security policies, or aligns with architectural standards. It simply responds to the immediate request.

Context-aware assistants

This proactive AI solution monitors ongoing activities, offers relevant suggestions, but still requires human approval before taking action.

Example: While a developer works on a feature, the AI notices they're implementing authentication logic and proactively suggests using the enterprise's standard authentication library instead of custom code. It might also flag potential security vulnerabilities in real-time as code is written.

Agentic systems

Autonomous agentic systems can independently initiate actions, make decisions within defined boundaries, and coordinate complex workflows without requiring human approval for each step.

Example: When a critical security vulnerability is discovered in a dependency, AI agents could take action by:

- Automatically scanning codebases to identify affected applications
- Assessing business impact and prioritizing remediation
- Creating patches for affected services
- Coordinating with deployment systems
- Updating documentation and notifying relevant stakeholders

Agentic AI enables "lights-out" development operations where routine tasks, optimizations, and even complex problem-solving happen automatically. This allows human developers to focus on strategic architecture, business logic, and innovation rather than operational overhead, with AI agents involving their human counterparts only when necessary.

AI point solutions vs. AI-native DevSecOps platform

AI is only as powerful as the data you give it. An AI-native platform is able to make much more intelligent decisions compared to AI point solutions or even third-party agentic AI tools because of its access to your organization's unified knowledge graph. Explore in the chart below the differences between an AI point solution and the agentic AI platform approach.

	AI point solutions	AI-native DevSecOps platform
Architectural differences	Each AI tool maintains its own data models, user interfaces, and integration points. Coordination happens through manual developer intervention or integrated systems.	Agents share semantic understanding of context, enabling them to communicate through standardized protocols, execute across lifecycle workflows, and allocate responsibilities based on current priorities and capabilities.
Decision-making capabilities	Point solutions provide recommendations, such as code suggestions, but do not have the same level of context as applications native to a platform.	Agentic systems drive AI and human collaboration, with the ability to make autonomous decisions within defined guardrails, escalating only when facing novel situations or policy conflicts.
Governance & compliance	Individual compliance validation is needed for each tool.	Pre-defined governance rules for all agents create consistent policy adherence.
Data & context management	Uses isolated data silos and requires manual context transfer to have sufficient information.	Agents have access to an enterprise's unified knowledge graph. When any agent updates information in the knowledge graph, relevant context automatically flows to other agents without manual intervention.
Scalability	Point solutions only offer linear scaling as each new use case requires new tools.	The scaling is exponential. Agents can combine capabilities for new use cases and organizations can design custom flows.
Maintenance overhead	Each tool requires its own updates, security patches, and integration maintenance.	Platform maintenance is centralized with coordinated maintenance that updates all agents.

How agents work together: Agent orchestration

Similar to the orchestration layer which coordinates and manages the execution of tasks or services within a larger system, agent orchestration is a system that manages and coordinates the interactions between multiple AI agents, enabling them to work together to achieve complex goals. By acting as a central control unit, the orchestration layer delegates tasks to agents, breaking down complex problems into subtasks, assigning those subtasks, managing dependencies, and integrating with external systems.

Just as individuals within DevSecOps teams have their own specialties, so do AI agents. Enterprise-grade agent orchestration employs specialized agents with distinct capabilities that operate using shared protocols and interfaces. The specialized agents could include:

Architecture agents

understand design patterns, technical debt implications, and system scalability constraints.

Security agents

maintain real-time awareness of threat landscapes, vulnerability databases, and compliance requirements.

Performance agents

monitor system behavior, identify optimization opportunities, and predict capacity needs.

These agents collaborate through standardized communication protocols (that your teams define) rather than rigid integration points. The AI system becomes greater than the sum of its parts, producing solutions that integrate seamlessly into existing systems rather than creating technical debt.

How it works in practice

Perhaps what's most remarkable about agent orchestration is that agent actions can occur simultaneously rather than sequentially. For example, if the agents are implementing a customer-facing API:

- ✓ **The architecture agent** ensures design consistency with existing services
- ✓ **The security agent** validates authentication mechanisms and data protection requirements
- ✓ **The testing agent** generates comprehensive test suites including edge cases
- ✓ **The deployment agent** coordinates blue-green deployment strategies

The orchestration layer manages agent negotiations, resource allocation, and conflict resolution. When agents disagree on implementation approaches, the system can invoke higher-level agents or escalate to human decision-makers.

Flow-based task coordination

Complex enterprise development requires sophisticated coordination beyond simple API calls. Flow-based orchestration defines business process templates that agents execute while adapting to specific contexts and constraints.

A feature development flow might coordinate requirements analysis, architecture design, implementation, testing, security validation, performance optimization, documentation updates, and deployment coordination. Rather than rigid sequential processing, flows enable parallel execution where possible, dynamic exception handling, and adaptive responses to changing requirements without manual reconfiguration.

Flows also play an important part in maintaining enterprise governance by encoding critical business controls, like:

Approval gates

Automated stakeholder notifications with escalation timers for human review

Compliance checkpoints

Continuous validation against regulatory requirements and industry standards

Audit trail requirements

Comprehensive logging of decisions, changes, and approvals for compliance reporting

Quality gates

Automated quality checks with defined thresholds for advancement to next stages

This governance framework allows agents autonomy within defined boundaries while ensuring that enterprise policies, security requirements, and compliance obligations are enforced across development activities.

Adaptive agents create optimized systems

One of the benefits of agentic AI is its ability to take all of the information at its disposal and adapt accordingly. Agents can reallocate resources in real-time, adjust processing priorities, and modify workflows based on current conditions without requiring human intervention for routine issues.

This adaptive capability includes proactively optimizing your systems: agents continuously analyze system performance, identify improvement opportunities, and implement optimizations during low-impact periods, ensuring enterprise systems evolve based on usage patterns and business requirements.

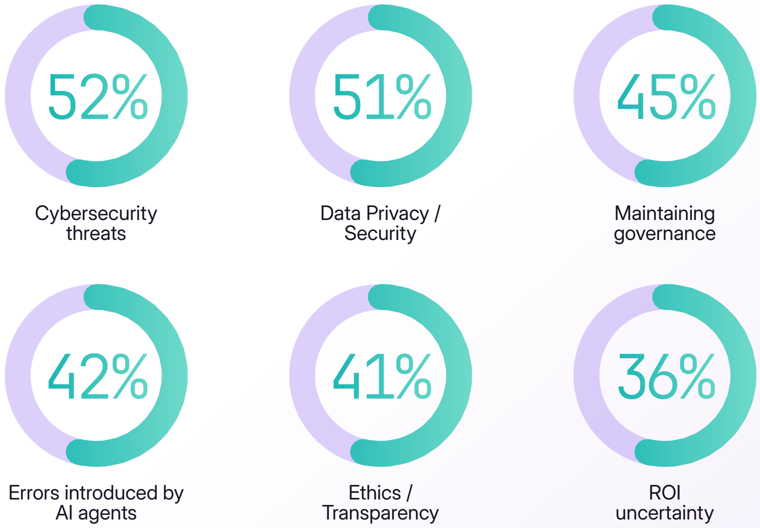
Secure by design: Agentic AI in application security

Key insight: While 85% of executives anticipate that agentic AI will create unprecedented security challenges, 45% recognize that improved security is one of AI's biggest benefits to software development.

Application security leaders face uncharted challenges when evaluating agentic AI adoption. Recent research reveals **executives' top concerns** around agentic AI adoption, cybersecurity threats (52%), data privacy and security (51%), and maintaining governance frameworks (45%).

These concerns are valid. Agentic AI systems operate with elevated privileges, access sensitive codebases, and make autonomous decisions that, if not set up with the right guardrails in place, could impact security posture. However, this challenge presents an opportunity: properly implemented agentic AI can potentially strengthen enterprise security through data analysis for complex anomaly detection, platform-specific context for vulnerability scoring, and alert summaries for faster remediation.

Top concerns around adoption of agentic AI



Note: 4% answered "N/A – no concerns around the adoption of agentic AI."

To securely design AI usage, application security teams are implementing the following guardrails:

Access control and privilege management

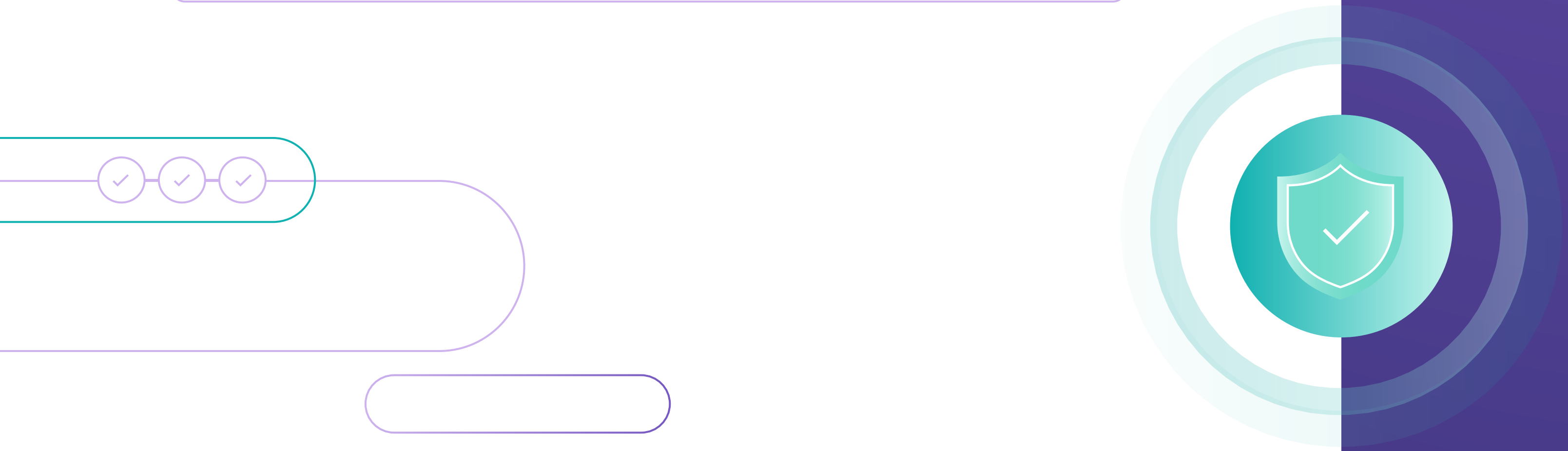
Implement role-based access controls (RBAC) where AI agents operate with minimum necessary privileges. For example, security agents could be limited to read-only access to vulnerability databases and require human approval for critical system changes. Whereas, code generation agents could have access to only relevant repositories with clear boundaries around sensitive data.

Data classification and handling

Establish automated data classification that tags code repositories, documentation, and system configurations by sensitivity level so that agents automatically apply appropriate handling procedures. For example, general functions can be processed in shared AI infrastructure while highly sensitive financial algorithms never leave secure environments.

Audit trails and monitoring

Leverage activity tracking to capture all agent decisions, data access patterns, and system modifications. Security teams can trace any change back to the specific agent, decision logic, and human oversight involved. Real-time monitoring with automated alerts for unusual agent behavior or policy violations flags teams if anything goes awry.



Measuring agentic AI success and ROI

Key insight: 91% of executives agree that measuring business impact provides a more accurate reflection of software development team performance than traditional metrics like code volume.

Measuring AI requires moving beyond traditional metrics like code volume to business outcomes that impact an organization's bottom line. Leading organizations are evaluating AI's impact on core business objectives such as: increased business growth, improved security posture, enhanced developer productivity, greater cost efficiency, and enhanced customer experience.

Business growth

Tying AI tools to revenue is the number one way to prove value and show ROI. Fortunately, AI investments pay for themselves in **less than two years**, with 44% of executives saying it takes less than one year.

To help you identify AI's influence, use the following revenue attribution metrics. Make sure to establish baseline delivery timelines pre-AI usage, then track revenue generated from accelerated releases and new capabilities.

- **Feature velocity to revenue:** Track revenue generated from features delivered faster due to AI acceleration
- **Time-to-market reduction:** Measure days/weeks saved in product launches and calculate revenue impact
- **Market response speed:** Monitor competitive advantage gained through faster feature responses to market demands
- **Innovation pipeline:** Count new products/services enabled by AI-freed developer capacity

Improved security posture

Reducing the amount of time spent addressing security issues means more time for developers and security team members to focus on projects that move the needle. By tracking incident management metrics, vulnerability scan results, and compliance audits, you can see how AI is reducing organizational risk.

- **Vulnerabilities remediated:** Types of security issues identified and fixed by AI agents
- **Incident response time:** Compare the duration of time between the occurrence of an incident and the first actions taken to remedy the incident before/after AI implementation
- **Mean time to resolution (MTTR):** Compare time to resolve the incident before/after AI implementation
- **Compliance automation:** Percentage of regulatory requirements automatically validated and maintained
- **Proactive threat prevention:** Number of potential problems identified and resolved before reaching production

Enhanced developer productivity

While developer productivity may not be the sole component you focus on, it certainly should be included. Business leaders estimate that developers **saved 943 hours (117 business days)** over the past year as a result of investing in AI.

To help you identify time-saved and productivity gained, monitor sprint metrics, commit frequency, and developer time allocation.

- **Story points per sprint:** Sustained velocity improvements across development teams
- **Code review cycle time:** Reduction in review duration through AI-assisted quality checks
- **Deployment frequency:** Increased release cadence enabled by automated testing and validation
- **Developer focus time:** Hours reclaimed from routine tasks and allocated to strategic work

Greater cost efficiency

One of the easiest ways to evaluate ROI is to look at dollars saved. By reviewing operational costs, including tools, infrastructure, and human effort, you can see the impact of agentic AI on your bottom line.

- **Tool consolidation savings:** License costs eliminated through unified AI platform adoption
- **Infrastructure optimization:** Resource utilization improvements and right-sizing
- **Manual process reduction:** Hours saved on repetitive tasks multiplied by developer hourly costs
- **Error prevention costs:** Avoided costs from bugs, security incidents, and compliance violations

Enhanced customer experience

A metric that is less talked about, but vital to business success is customer quality metrics. By monitoring satisfaction scores, application performance metrics, and support systems data, you can learn how AI improvements have impacted the customer experience.

- **Bug escape rate:** Reduction in customer-reported defects
- **Application performance:** Response time improvements and uptime increases
- **Feature satisfaction:** Customer feedback on AI-accelerated feature quality
- **Support ticket volume:** Reduction in customer issues due to proactive problem resolution

Put AI ROI measurement into practice. **Transform raw usage data into actionable business insights and ROI calculations with this in-depth tutorial. >**

Transforming your business with agentic AI

Key insight: With agentic AI, development teams can achieve 10x productivity gains through intelligent automation and context-aware assistance.

Early adopters of AI solutions are already experiencing results like accelerated time-to-market and positive ROI within two years. Organizations that avoid the next wave of AI innovation risk losing ground to competitors that use autonomous agents to build secure software faster (and at a lower cost).

Success requires moving beyond point solutions toward unified platforms that provide comprehensive context of the software development lifecycle (SDLC) and seamless workflow orchestration. **GitLab's Duo Agent Platform** delivers this advantage, providing AI agents with the full context of your environment to understand project history, architectural decisions, and business requirements simultaneously.

The GitLab advantage: Native AI at every step

Solving the data foundation challenge

Most enterprise AI implementations fail because they operate on fragmented, incomplete datasets scattered across disconnected tools. GitLab's unique position as the comprehensive system of record for the entire SDLC provides the unified data foundation that makes truly intelligent AI possible.

Enterprises need AI that works across wider contexts throughout the SDLC, not just coding assistance. While point solutions address isolated development tasks, GitLab’s Duo Agent Platform captures the complete development narrative, from initial requirements and architecture decisions through deployment and monitoring.

Leveraging a unified knowledge graph

GitLab's Knowledge Graph connects code repositories, merge requests, issues, CI/CD pipelines, security scans, and deployment metrics. With real-time code indexing, your team gains access to more accurate, faster, contextual results. This unified context enables GitLab's specialized agents to unlock insights and understand not just what code does, but why it was written, how it integrates with business objectives, and what compliance requirements it must satisfy.

Native AI vs. bolt-on solutions

Unlike bolt-on AI solutions that require complex integrations and manual context transfer, GitLab's agentic AI capabilities are natively integrated into every SDLC stage. This eliminates the friction, data silos, and coordination overhead that plague multi-vendor AI deployments.

GitLab's **Model Context Protocol (MCP)** and extensive integration ecosystem ensure that AI agents can coordinate across all relevant enterprise systems, including those external to GitLab, while maintaining GitLab as the central orchestration layer. Through our MCP, teams can connect external AI assistants like Claude or Cursor to work within your GitLab environment, greatly reducing context switching.

GitLab offers integrations for Claude Code, Codex, Amazon Q, Google Gemini, and opencode. By @ mentioning your agents directly in issues or merge requests, you can loop in third-party agents while keeping all interactions natively within GitLab's interface.

Built-in security and compliance

Security and compliance aren't afterthoughts — they're integrated into GitLab's AI architecture from day one. Our built-in agent tracking provides visibility into agent decision-making processes. This visibility serves as a system of record for auditing purposes and provides insights to improve future agent flows.

If you have a self-hosted GitLab instance, Duo Code Review keeps all intelligence within your infrastructure. Meanwhile, hybrid model configurations let you mix self-hosted models for sensitive work with GitLab's cloud models for other tasks, delivering flexibility and governance.

The business impact of agentic AI adoption

AI agents across the SDLC:
10x developer productivity

As mentioned in this **video** with GitLab CEO Bill Staples, development teams can achieve 10x productivity gains through intelligent automation and context-aware assistance, cost reduction via optimized resource utilization and reduced tool sprawl, and integrated security that strengthens rather than compromises enterprise governance.

GitLab's AI-native DevSecOps platform creates the data foundation, contextual awareness, and user experience required for AI to deliver sizable enterprise value rather than incremental tool improvements.

> “GitLab Duo Agent Platform enhances our development workflow with AI that truly understands our codebase and our organization. Having GitLab Duo AI agents embedded in our system of record for code, tests, CI/CD, and the entire software development lifecycle boosts productivity, velocity, and efficiency. The agents have become true collaborators to our teams, and their ability to understand intent, break down problems, and take action frees our developers to tackle the exciting, innovative work they love.”
>
> **Bal Kang**
> Engineering Platform Lead at NatWest

Explore the latest agentic AI features we’re building at GitLab >

Step into the era of agentic development

GitLab Duo Agent Platform delivers the unified agentic AI solution your enterprise needs. Unlike point solutions that add complexity, our platform provides AI agents with complete SDLC context through GitLab's comprehensive unified data platform and knowledge graph — from code repositories and merge requests to CI/CD pipelines and security scans.

What makes GitLab different:

- **System of Record:** Your unified data platform securely holds source code, project plans, CI/CD configurations, and compliance data — creating contextual intelligence unavailable to generic AI tools.
- **Native integration:** AI agents work seamlessly across your entire development lifecycle through our integrated control plane, not as bolt-on tools.
- **AI extensibility:** The GitLab MCP server enables secure integration with Cursor, Claude Code, Amazon Q, Google Gemini, and other AI tools while maintaining governance.
- **Intelligent orchestration:** GitLab Flows coordinate multiple AI agents with pre-built workflows, from the Software Development Flow to the Issue to MR Flow, handling routine tasks autonomously.
- **Enterprise security:** Securely design your AI usage with built-in governance, audit trails, and compliance controls with least-privilege access and centralized policy management.

Stop managing disparate AI tools.
Start orchestrating intelligent workflows.

Learn more about GitLab Duo Agent Platform >

GitLab

Software.
Faster.