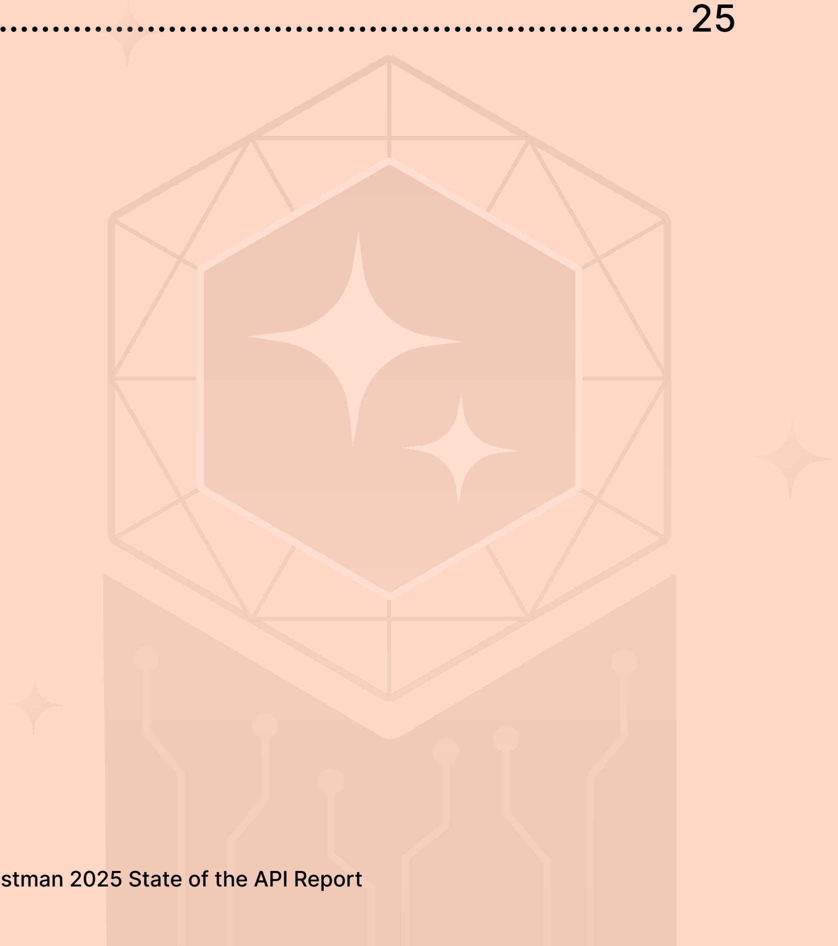
POSTMAN

7th Annual

2025 State of the API Report

APIs are no longer just powering applications. They're powering agents. This year, our survey of over 5,700 developers, architects, and executives across the globe reveals that API strategy is fast becoming AI strategy.

Table of Contents



Introduction

When we first started working with APIs, they were internal tools that looked like glue code between services, wrappers around business logic, or endpoints buried deep inside engineering docs. They were brittle, undocumented, and hard to share. Fast-forward to today, and we're witnessing an inflection where teams are either modernizing APIs to support AI-native use cases or struggling to retrofit in a world that's already moved beyond human-only interactions for APIs.

This year's State of the API report captures a turning point: APIs are no longer just powering applications; they're powering agents.

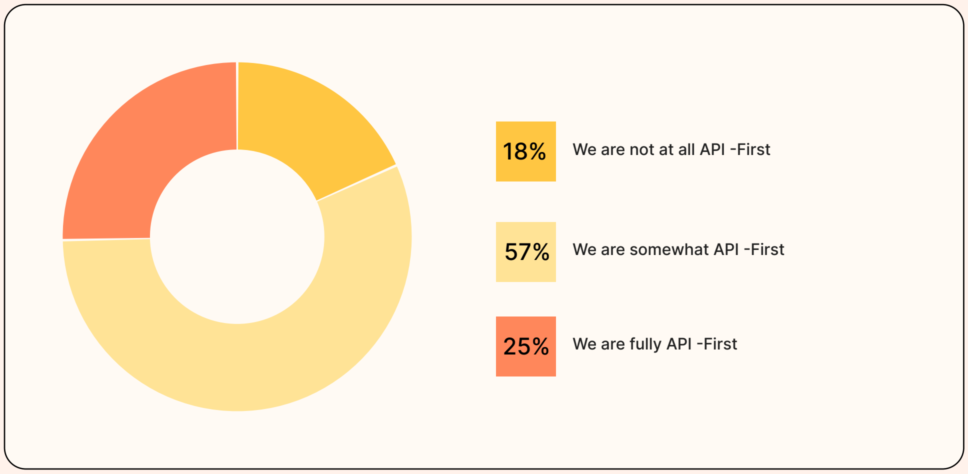
To understand how this shift is reshaping developer experience, product strategy, and operational models, we surveyed over 5,700 developers, architects, and executives across the globe.

What emerged is a clear signal: API strategy is fast becoming AI strategy.

API-first development is accelerating, up 12% from last year

For years, API-first was a promising idea: by treating APIs as products rather than projects, organizations would drastically benefit from how they built, scaled, and monetized their digital offerings.

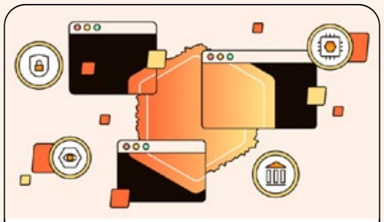
Today, the data makes it clear: the shift from code-first to API-first is not just happening, it's accelerating.



Eighty-two percent of organizations have adopted some level of an API-first approach, with 25% operating as fully API-first organizations, a 12% increase from 2024. This represents a strong signal that APIs are no longer seen as byproducts of engineering, but as durable products that are the foundation for adopting AI agents.

This shift mirrors what we've seen firsthand: when APIs are treated as long-lived products with roadmaps, feedback loops, and SLAs, they unlock scale in ways code-level abstractions never could. Instead of brittle handoffs and repeated rewrites, teams start designing for reuse.

And it's not just a technical transformation, it's organizational. Fully API-first teams are often the ones aligning product and engineering early, embedding governance into workflows, and thinking about how APIs will be consumed both by humans and machines, internally and externally.



API Maturity Assessment

You may be paying the price for poor APIs.

Take the assessment →

Inside the report

We'll examine five critical trends reshaping how an API-first approach supports AI adoption:

The AI-API gap

89% of developers use AI, but only 24% design APIs for AI agents.

AI agents are the new API consumers

AI agents bring efficiency and scale, but also introduce fresh concerns—51% of developers now cite unauthorized agent access as a top security risk.

APIs drive revenue

APIs have become profit drivers, with 65% of organizations generating revenue from their API programs.

MCP awareness surges, but adoption lags

The Model Context Protocol (MCP) is emerging as the connective layer between AI agents and APIs for machines to discover, understand, and invoke APIs. While 70% of developers are aware of MCP, only 10% are using it regularly, pointing to a growing interest but limited readiness.

Broken collaboration means broken APIs

93% of teams struggle with API collaboration, leading to duplicated work, delays, and degraded quality.

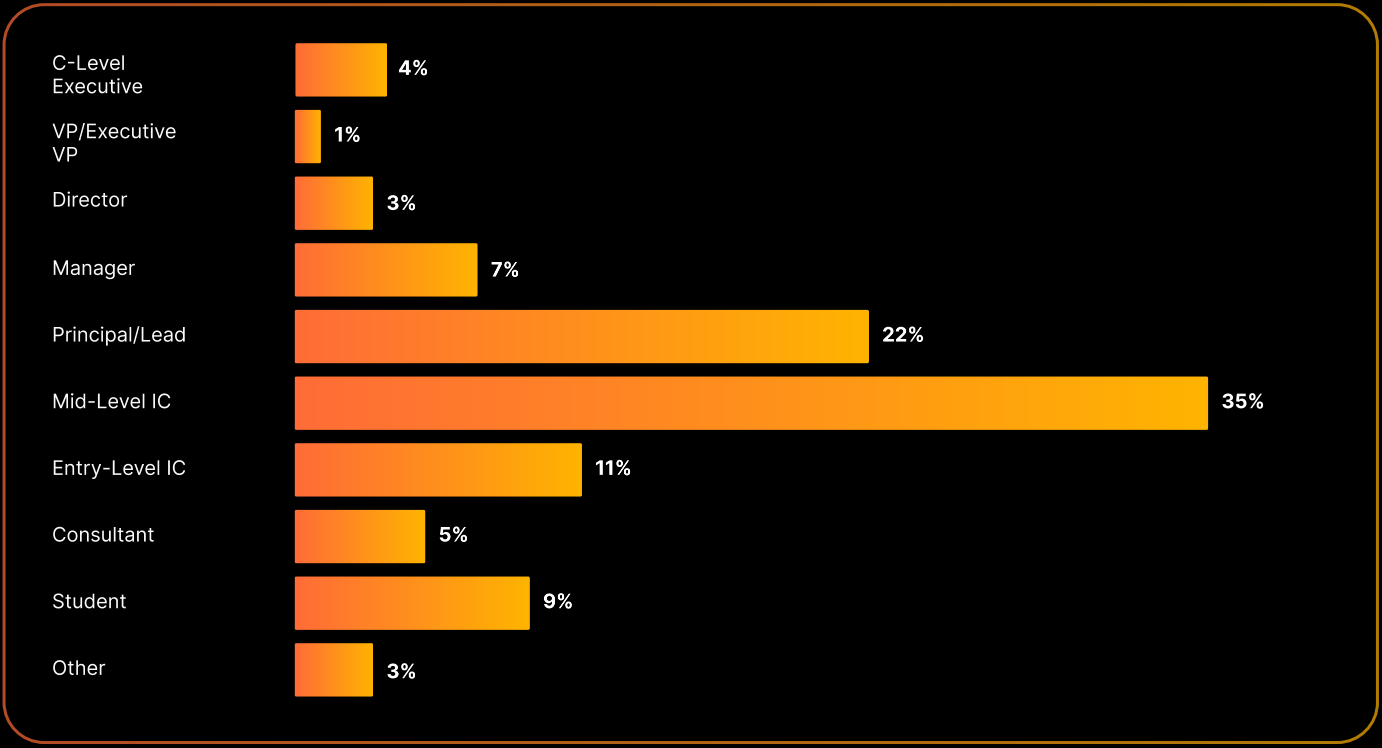
Who's behind the data

To understand where APIs are headed, it's critical to learn from the people building them. This year's survey reflects insights from those on the front lines. Seventy-three percent of respondents work in engineering or software development.

The voices span seniority levels:

- 8% are executives, VPs, or directors (including 4% C-level) shaping API strategy.
- 22% are principals and tech leads translating vision into architecture.
- 35% are mid-level developers focused on implementation.

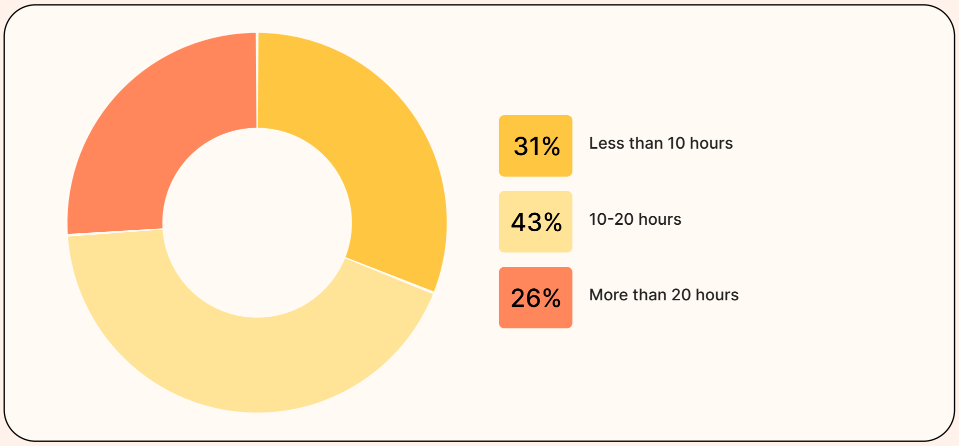
This distribution gives us a balanced view grounded in real-world challenges, strategic decisions, and day-to-day API work.



Who's Behind the Data

API development is global and asynchronous

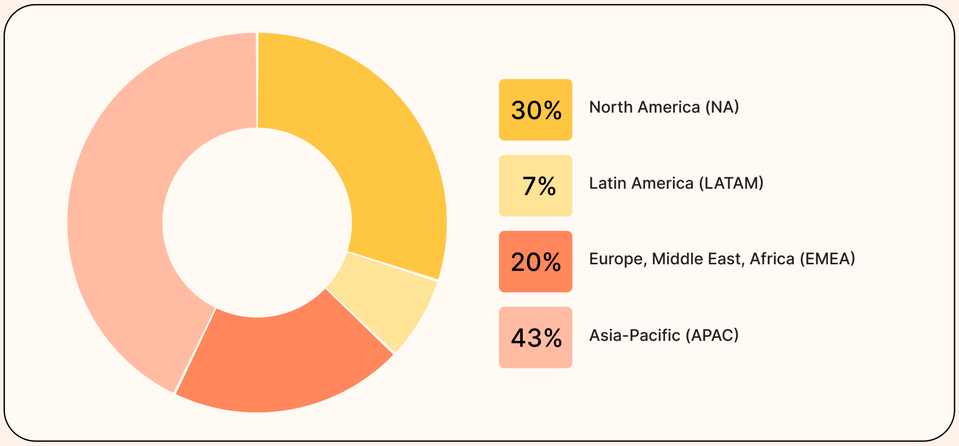
APIs are foundational, critical work for developers with 69% spending 10+ hours per week on API-related tasks, making it a significant portion of their professional focus.



26%

of developers spend more than 20 hours a week on API-related tasks.

The work is truly global: 43% are located in Asia-Pacific and 30% in North America, creating a distributed workforce that spans time zones. When mid-level engineers across different continents do most of the API work, code-only practices can break down. You need shared, repeatable artifacts so anyone can ship safely without tribal knowledge.

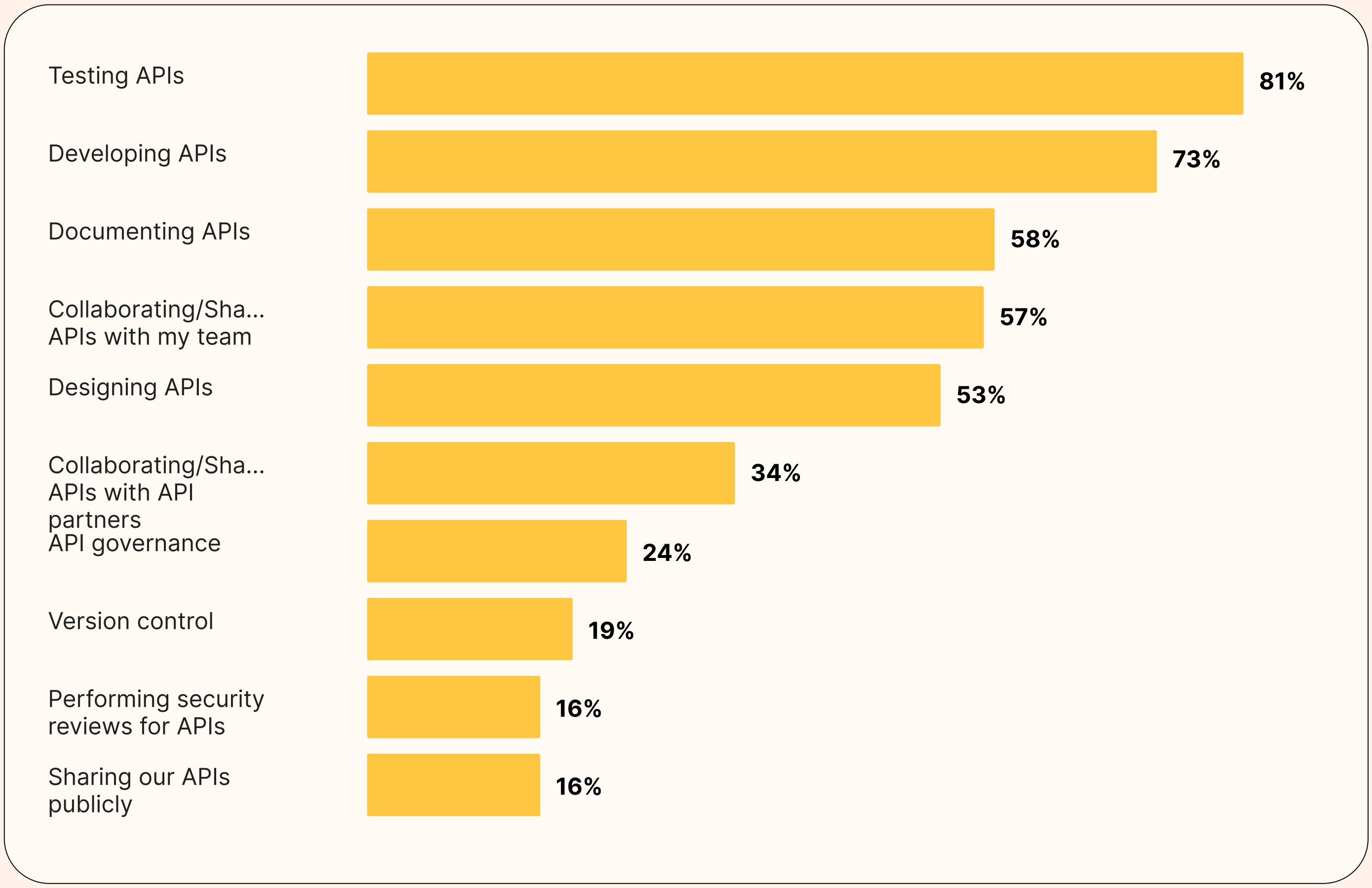


API-related work demands a solid foundation

The most common API activities reveal why coordination matters so much:

- **Testing (81%):** verifying API behavior and contracts
- **Developing (73%):** writing and maintaining API implementations
- **Documenting (58%):** creating and updating API specifications

Who's Behind the Data



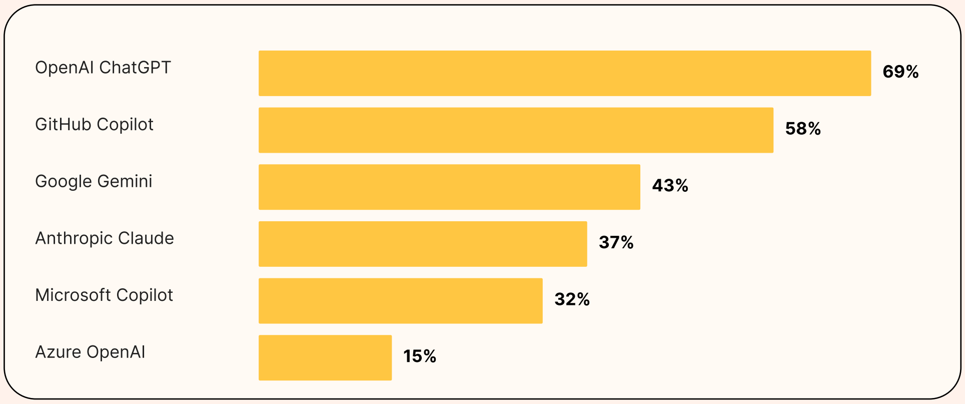
With distributed teams spending significant time on testing, development, and documentation, most changes happen asynchronously. This reality makes the following trends even more critical because the practices that work for co-located teams that build simple applications break down when applied to global teams that build complex API ecosystems.

Understanding who works with APIs helps explain why the trends we're seeing matter so much. When engineers distributed across time zones handle the majority of API work, the challenges around AI adoption, security, collaboration, and standardization become magnified. Let's examine how these global teams are navigating five critical shifts in the API landscape.

Developers are AI-native. Most APIs are not.

Developers have universally embraced AI tools. Eighty-nine percent use generative AI in their daily work. However, not as many design APIs to handle AI workloads.

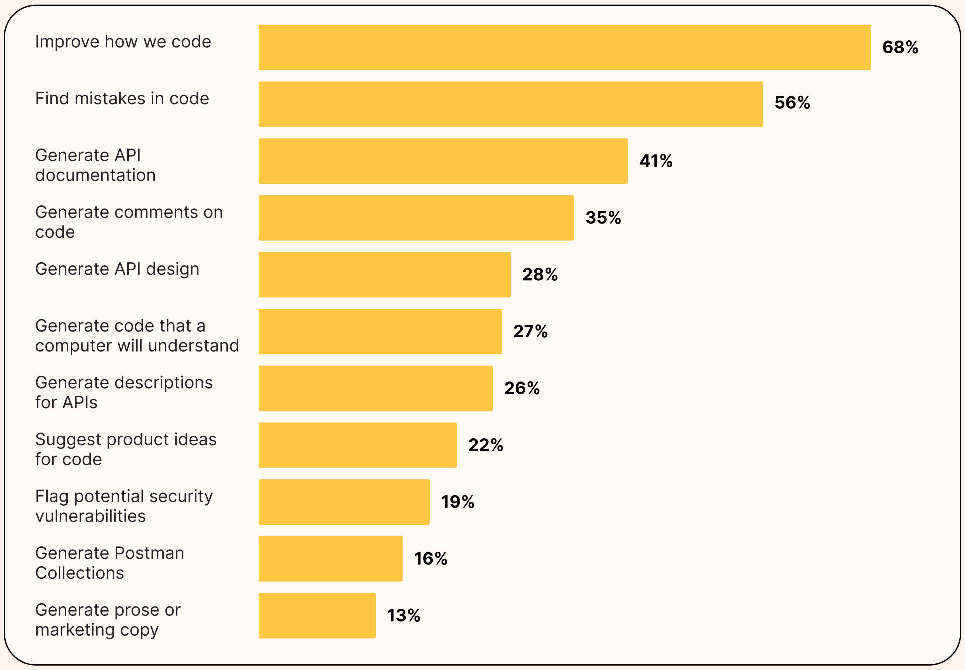
Many developers still assume human consumption, which creates a fundamental mismatch between how software is built and how it's designed and for whom it's designed.



7.5m

In the past 12 months, Postman has seen 7.53M calls made to AI APIs (+40% YoY).

Respondents rely on AI to improve code quality (68%), generate API documentation (41%), and accelerate development cycles.

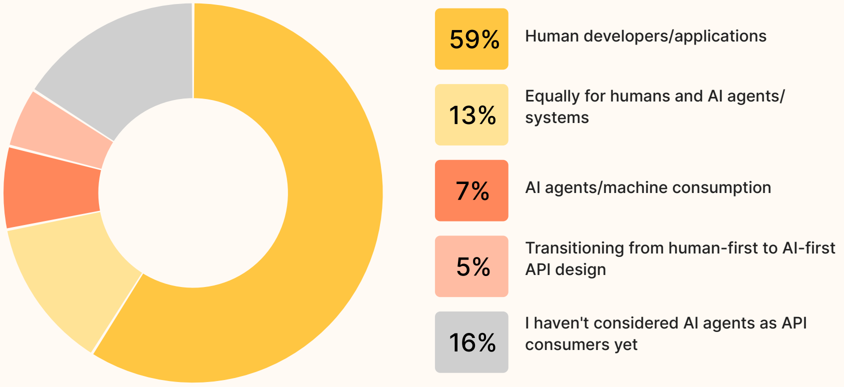


Yet this widespread AI adoption reveals a critical gap: while developers build with AI, most APIs aren't designed for AI consumption.

Only 24% of developers actively design APIs with AI agents in mind. The breakdown reveals the industry's cautious approach:

- 13% design equally for humans and AI agents
- 7% primarily design for AI agents/machine consumption
- 5% actively transitioning from human-first to AI-first design

Meanwhile, 60% still design primarily for humans only, and 16% haven't even considered AI agents as API consumers yet.



This mismatch has real consequences. More of your integration code is now written or assisted by AI. AI Agents rely on precise, machine-readable signals, not tribal knowledge. When your API lacks predictable schemas, typed errors, and clear behavioral rules, AI agents can't function as they're intended to.

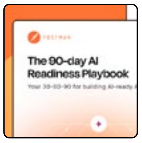
Pro Tip

Use Postman's API documentation tool to generate dynamic, machine-readable documentation for your APIs and automatically keep it up to date.

Learn More →

Outdated or incomplete documentation frustrates consumers and triggers hours of back-and-forth to find accurate information. Integrated documentation keeps API docs live and in sync with collections, so consumers always work from the latest version.

The challenge isn't whether to adapt, it's how quickly you can bridge the gap between AI-powered development and AI-ready infrastructure.



The 90-day AI Readiness Playbook

AI-ready APIs in 90 days

Get the resources you need for AI-ready deployment in 90 days, with APIs that are structured, tested, and trusted by both humans and agents.

Download the playbook →

While developers rush to embrace AI tools, a new challenge emerges: These same AI systems are becoming the primary consumers of APIs, creating unprecedented security risks that traditional models weren't designed to handle.

OpenAI dominates AI traffic (56% of total Postman AI traffic), racking up 4.2M calls over the past 12 months.

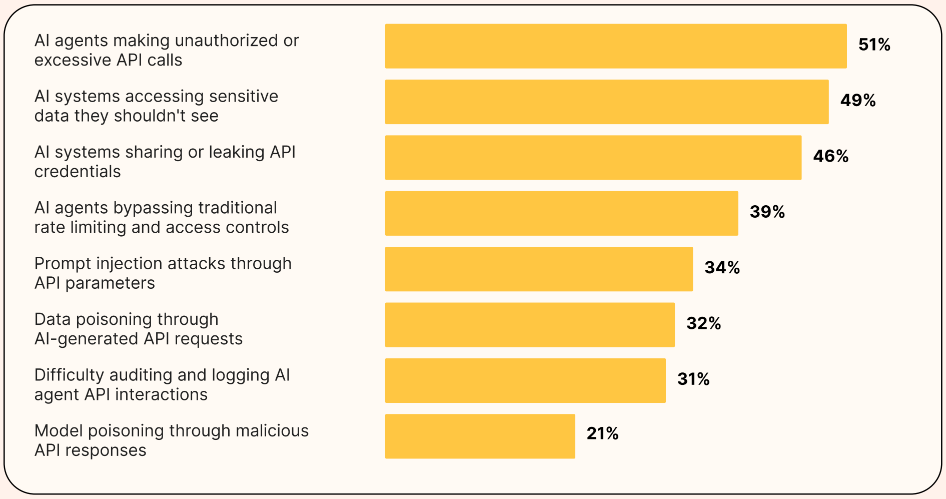
Gemini and Llama experienced 3.1x and 6.9x year-over-year growth, respectively.

AI Agents are your new API consumers.

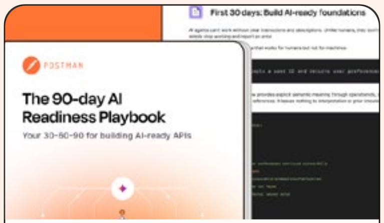
An AI agent can hit your APIs at machine speed with perfect persistence. One leaked token or an over-scoped key can create a system-wide vulnerability. What's more? If you can't tell a human from an agent, you can't enforce least privilege, detect abuse, or meet compliance requirements.

Nonetheless, AI agents are API consumers, calling endpoints thousands of times per second, processing data at scales no human could match, and integrating systems in ways that traditional API design never anticipated.

But this transformation brings an unexpected twist: The same AI agents that developers build with are now their top security concern.



Fifty-one percent of developers worry about unauthorized or excessive API calls from AI agents—making it their number one security concern. Close behind, 49% are concerned about AI systems accessing sensitive data they shouldn't see, and 46% worry about AI systems sharing or leaking API credentials.



POSTMAN

The 90-day AI Readiness Playbook

Your 30-60-90 for building AI-ready APIs

First 30 days: Build AI-ready foundations

90-Day AI Readiness Checklist

Begin your 90-day transformation to turn your APIs into AI-ready tools.

Learn more →

New threat models require adjustments

Traditional API security models were designed for predictable human behavior: developers making dozens of calls per day, following documented patterns, and operating within reasonable rate limits.

AI agents disrupt these assumptions:

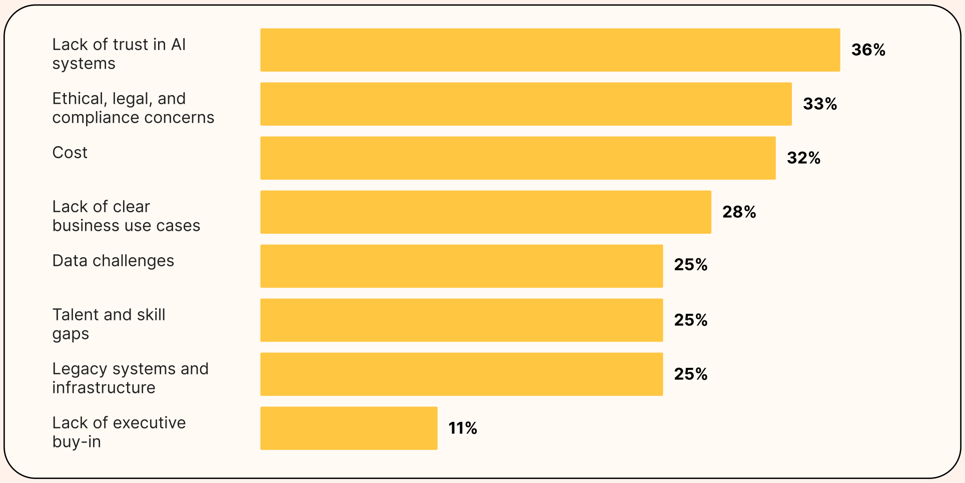
- **Machine-speed exploitation:** Agents probe vulnerabilities and exploit weaknesses faster than security teams can respond.
- **Persistent automated attacks:** Unlike humans, agents maintain attacks indefinitely, systematically testing every endpoint.
- **Credential amplification:** A single compromised API key becomes a gateway to multiple systems and vast data extraction.
- **Behavioral unpredictability:** Agents access APIs in unexpected ways, making legitimate automation hard to distinguish from attacks.

51%

of developers worry about unauthorized or excessive API calls from AI agents.

Defending against non-human consumers

These security concerns aren't unfounded. When asked about the biggest obstacles to using AI tools at their organizations, developers cite real trust and compliance issues: 36% lack trust in AI systems, and 33% have ethical, legal, and compliance concerns. The solution isn't to avoid AI, though. It's to build the security infrastructure that makes AI adoption safe and compliant.



Strategic security adaptations:

- **Agent identification:** Distinguish between human and agent traffic with headers, tokens, or other mechanisms.
- **Dynamic rate limiting:** Move beyond simple requests-per-minute to behavioral pattern analysis.
- **Least privilege for agents:** Scope API keys granularly so agents access only what they need.
- **Enhanced monitoring:** Build real-time detection systems for suspicious agent behavior.
- **Credential rotation:** Implement shorter-lived tokens and automatic rotation to limit breach impact.

What you can do right now, with Postman:

- **Store secrets in Postman Local Vault** or leverage one of Postman's Vault Integrations and reference them in environments.
- **Add governance rules** to flag missing auth, missing 429 contracts, or overly broad scopes.
- **Use Postman MCP servers** to add a secure layer between your APIs and AI agents by strictly defining what data and actions are exposed. MCP servers in Postman only use the tools and endpoints you intentionally expose, giving you full control over what agents can access.
- **Reproduce production issues** with Postman Insights.
- **Run contract and security tests** in CI on every PR.
- **Track agent traffic and 401/403 spikes** with monitoring tools.
- **Connect Postman to your Git tool** so policies, tests, and specs live with code.
- **Set up real-time security alerts** by configuring notifications for suspicious events, policy violations, and anomalous API usage patterns through Slack/Teams integrations.

The organizations that adapt their security models now will be prepared for an agent-driven future.

As security challenges mount, there's another reality. APIs are no longer just infrastructure costs, they're revenue engines that demand a product mindset.

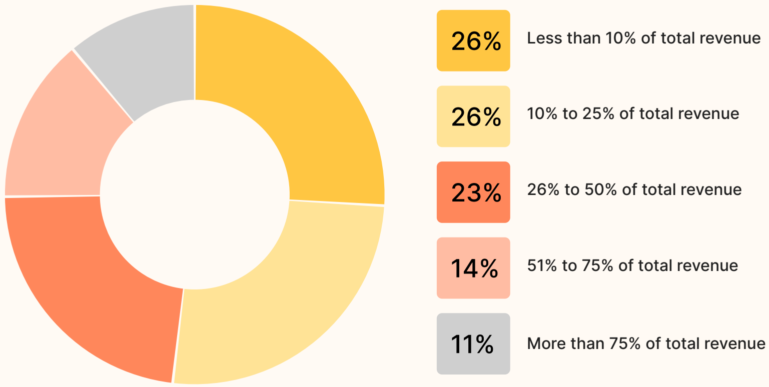
APIs drive revenue. Build them like it.

The business case for treating APIs as products, not projects, has never been clearer. Sixty-five percent of organizations now generate revenue from their APIs, proving that well-designed API programs transcend cost centers to become profit drivers.

65%

of organizations now generate revenue from their APIs.

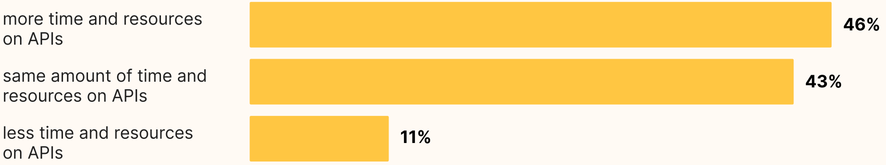
Among organizations that do generate API revenue, the distribution reveals substantial business impact across all levels. The majority (74%) generate at least 10% of their total revenue from APIs, with nearly a quarter (25%) deriving more than half their total revenue from API programs.



Momentum in API investment

Organizations are backing this revenue generation with increased investment. Forty-six percent plan to spend more time and resources on APIs in the next 12 months, compared to just 11% planning to reduce investment. This isn't just about direct monetization. It's about recognizing APIs as strategic assets that enable business growth.

In the next 12 months, my org will spend:

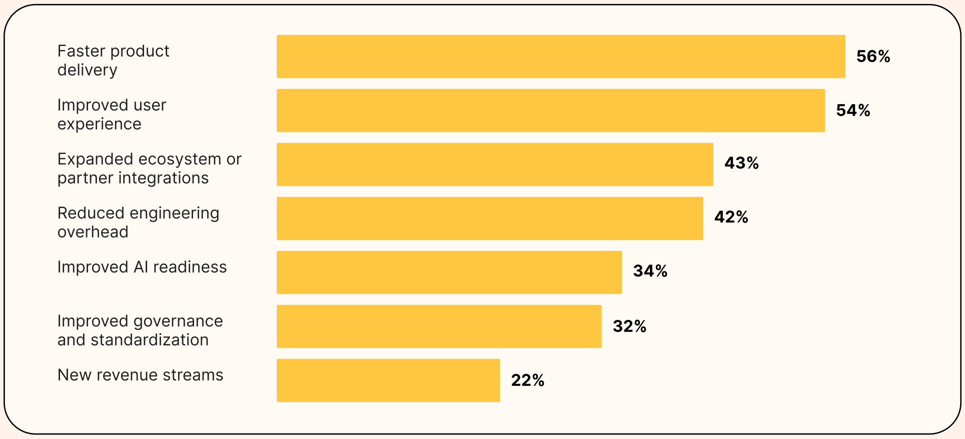


Pro Tip

Launch, distribute, and grow your APIs with the Postman API Network, where consumers can deliver real-time feedback for improvement.

Not only do organizations directly monetize their APIs, but revenue comes through multiple channels that compound over time:

- **Improved user experience (54%):** Better connected services and faster feature delivery create customer value that translates to business value.
- **Reduced engineering overhead (42%):** Less duplicate work and clearer integration patterns free up resources for innovation rather than maintenance.
- **Improved AI readiness (34%):** APIs designed for machine consumption position organizations to capitalize on AI-driven opportunities.
- **New revenue streams (22%):** Developer programs, partner ecosystems, and marketplace offerings provide direct monetization opportunities.



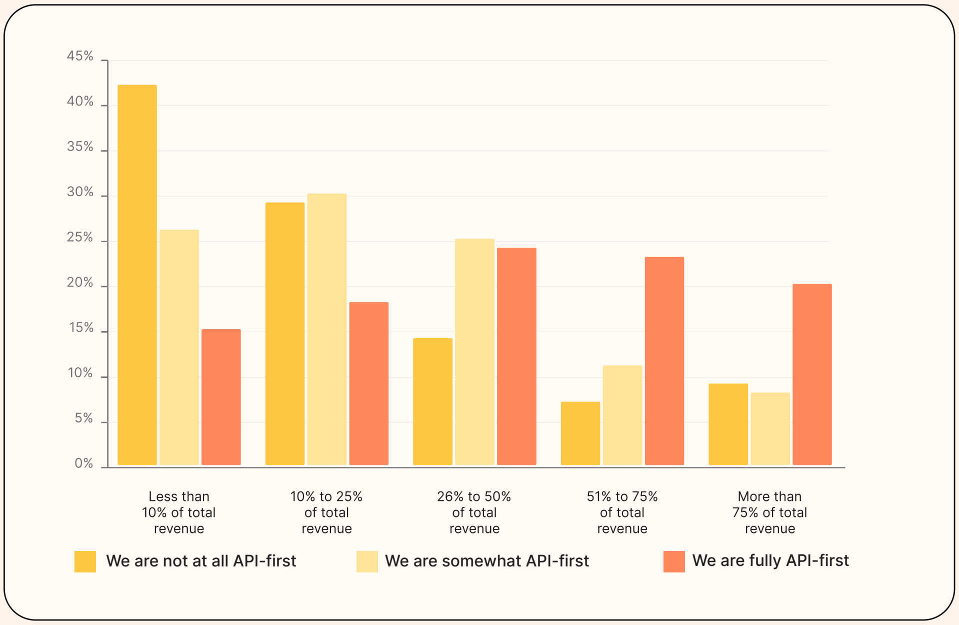
The API-first revenue reality

The connection between API-first practices and revenue generation is clear in the data. Organizations that are fully API-first are significantly more likely to generate substantial revenue from their APIs:

- **43% of fully API-first organizations** generate more than 25% of total revenue from APIs, compared to just 23% of somewhat API-first and 16% of non-API-first organizations.
- **20% of fully API-first organizations** generate more than 75% of total revenue from APIs, more than double the rate of other organizations.
- **Conversely, 42% of non-API-first organizations** generate less than 10% of revenue from APIs.

43%

of fully API-first organizations generate more than 25% of total revenue from APIs.



APIs as Revenue Drivers

This isn't coincidental. The organizations generating the most revenue from APIs share common characteristics in how they approach API development:

- **Contract-first design** enables parallel development and reduces integration friction.
- **Centralized governance** ensures consistency across teams and products.
- **Developer-focused documentation** makes adoption faster and more successful.
- **Usage monitoring and analytics** provide visibility into what drives value.
- **Automated testing and deployment** maintain reliability at scale.

> An API-first approach enables us to offer developers increased flexibility, accelerated time to market, and scalability. At PayPal, we're committed to creating a simplified experience for developers.
>
> **Mudita Tiwari,** Senior Director, Developer Experiences, PayPal

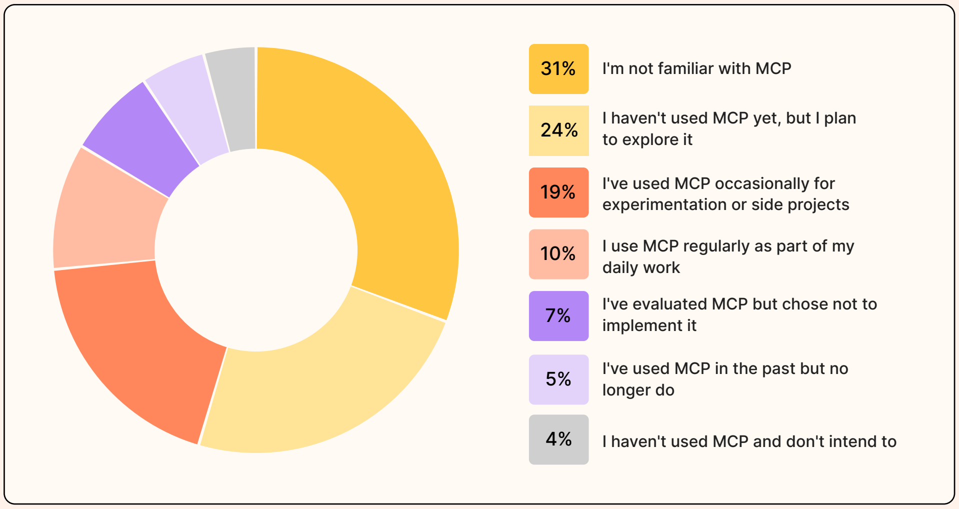
When adoption and reliability move the P&L, not just delivery speed, every API decision becomes a business decision.

With APIs driving real revenue, organizations face a critical infrastructure choice. How do they balance an API-first approach with preparation for an AI-driven future while emerging protocols are still gaining adoption?

MCP is early, but it's gaining momentum.

MCP is emerging as the connective layer between AI agents and APIs for machines to discover, understand, and invoke APIs.

Given that MCP launched just nine months ago, an impressive two-thirds of developers are already aware of it. This signals positive momentum for AI's emerging universal language. However, awareness doesn't equal implementation: only 10% use MCP regularly in daily work, though 24% plan to explore it, indicating significant future adoption.



This shift creates both a risk and an opportunity. While AI agents are becoming routine API consumers, most organizations haven't yet invested in the governance, observability, or security models needed to manage non-human access at scale.

24%

of developers plan to explore MCP.

Here's how you can build an MCP server yourself.

Learn More →

Agents don't wait for standards

MCP promises to be the structured interface between AI models and real-world systems. In theory, it solves critical problems like unified agent access, standardized security models, and structured tool definitions that agents can reliably understand.

But here's the critical insight: Agents are already calling your APIs, with or without MCP. With only 10% regular adoption, MCP hasn't reached mainstream developer workflows. Even among those who've evaluated it, many chose not to implement it, suggesting practical barriers to adoption.

If your interface isn't AI agent-ready, every team builds one-off wrappers that break, leak secrets, and waste time. Making your APIs agent-consumable now provides immediate benefits and future flexibility—whether MCP becomes the standard or something else emerges.

15k

MCP servers have been generated using Public Postman Collections

Try our no-hassle way of using existing API documentation on Postman to create MCPs.

Learn More →

Build agent-ready APIs today

Regardless of MCP adoption timelines, certain practices make APIs more consumable by both humans and machines:

- **Machine-readable schemas** with comprehensive OpenAPI specifications, including detailed examples, error codes, and response formats that AI can parse and understand.
- **Predictable patterns** across endpoints, consistent naming conventions, standard HTTP status codes, uniform authentication, and error handling that reduces cognitive load.
- **Comprehensive documentation** that includes context about when and why to use endpoints, not just how, helping AI agents make intelligent decisions about API usage.
- **Robust error handling** with typed error responses that provide actionable information for both human developers and automated systems.
- **Rate limiting and authentication** designed for high-frequency automated access patterns, not just human usage.

The organizations that make their APIs agent-consumable now position themselves to benefit from any agent framework. This is true whether MCP becomes the standard or something else emerges.

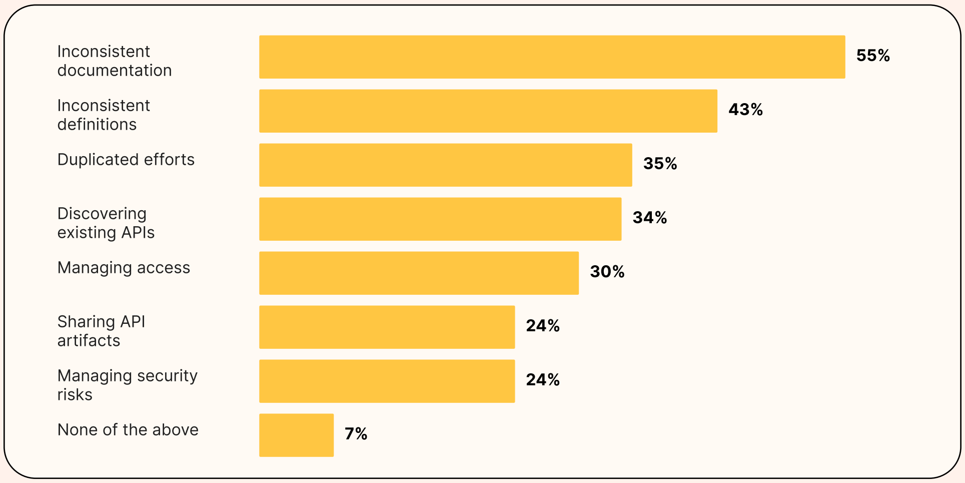
Agent-ready APIs solve the technical challenge, but there's an equally critical operational challenge. How do teams coordinate effectively when working with these increasingly complex systems?

Broken collaboration means broken APIs.

Despite all the progress in API tooling and methodologies, 93% of API teams still face collaboration blockers. Only 7% report having no collaboration challenges from the options provided, a surprisingly low number that reveals persistent operational friction even as technical capabilities advance.

The most common collaboration failures cluster around information and discovery:

- **Documentation struggles:** Inconsistent, outdated, or missing documentation creates confusion about API behavior, requirements, and usage patterns.
- **Duplicate efforts:** Teams rebuild functionality that already exists because they can't discover or access existing APIs within the organization.
- **Discovery problems:** Developers can't find APIs that solve their problems, leading to wasted time and duplicated work.



93%

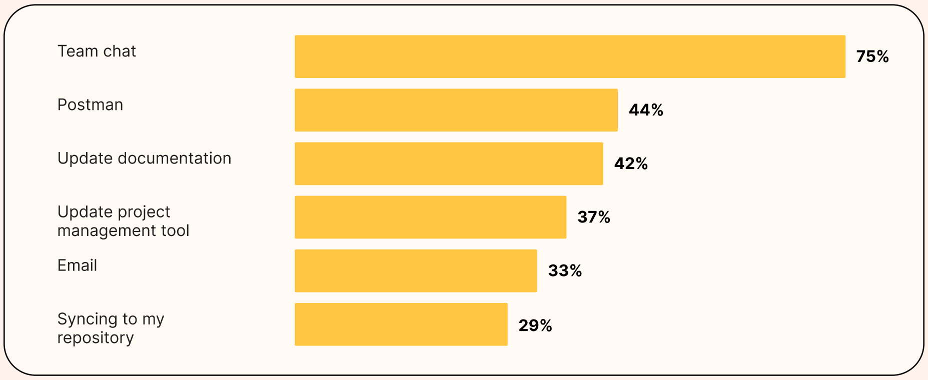
of API teams face collaboration blockers like inconsistent documentation and definitions.

These issues aren't just inconveniences. They directly impact delivery speed, code quality, and developer productivity.

What makes this especially striking is that 84% of teams work in small groups of 1-9 people, yet collaboration still breaks down. If small teams can't collaborate effectively on APIs, the challenges only compound as organizations scale.

The single source of truth dilemma

The root cause isn't a lack of documentation. It's that the documentation is scattered across too many places. Teams may use chat tools for quick questions, internal docs for formal specifications, emails for approvals, and wikis for examples. When API information lives everywhere, it becomes outdated or unreliable.



This means:

- **Context gets lost** when conversations happen in Slack, but specifications live in Confluence, and examples exist in someone's personal repository.
- **Updates go missing** when changes are communicated in one channel, but not updated everywhere the information exists.
- **Tribal knowledge builds up** when the real usage patterns and gotchas exist only in people's heads, not in discoverable formats.
- **Onboarding slows down** when new team members have to hunt across multiple systems to understand how APIs actually work.

Breaking down barriers to collaboration

Organizations solving collaboration challenges share common patterns:

- **Centralized API catalogs** where teams can discover existing APIs, understand their capabilities, and access current documentation without hunting across systems
- **Living documentation** that stays synchronized with code changes, ensuring information accuracy and reducing the documentation debt that causes confusion
- **Shared workspaces** where specifications, tests, examples, and conversations exist together, maintaining context and reducing information scatter
- **Usage analytics and feedback loops** that show which APIs are actually being used, how they're performing, and where integration problems occur
- **Governance workflows** that are built into development processes, rather than separate approval chains that slow teams down

The organizations that make their APIs agent-consumable now position themselves to benefit from any agent framework. This is true whether MCP becomes the standard or something else emerges.

Agent-ready APIs solve the technical challenge, but there's an equally critical operational challenge. How do teams coordinate effectively when working with these increasingly complex systems?

35m

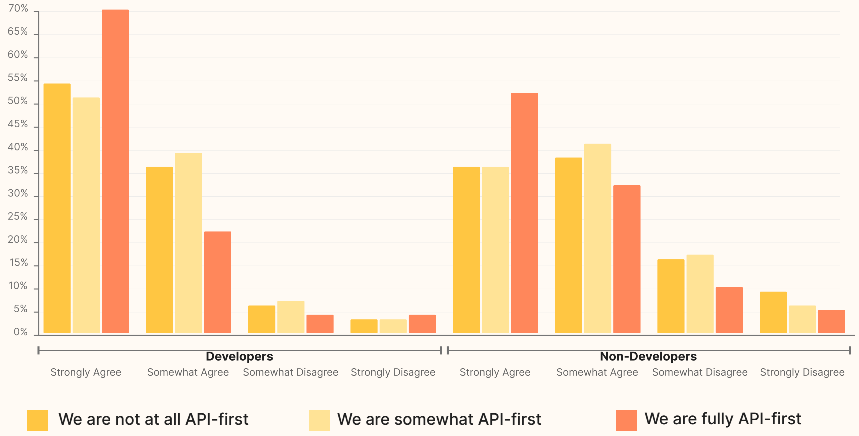
Postman Collections have been created in the last year across our top ten countries by collections usage.

Pro Tip

Centralize and store critical API-related information with Postman Workspaces where multiple collaborators can fork collections, iterate on designs, and maintain living documentation together.

Learn More →

Postman helps me collaborate with:



> “Postman's impact on how we build software at Toast has been huge. Not only has it saved time; it has enabled a massively distributed group of people to act as one team.
>
> **Liz Jackson,** Developer Relations Team Lead, Toast

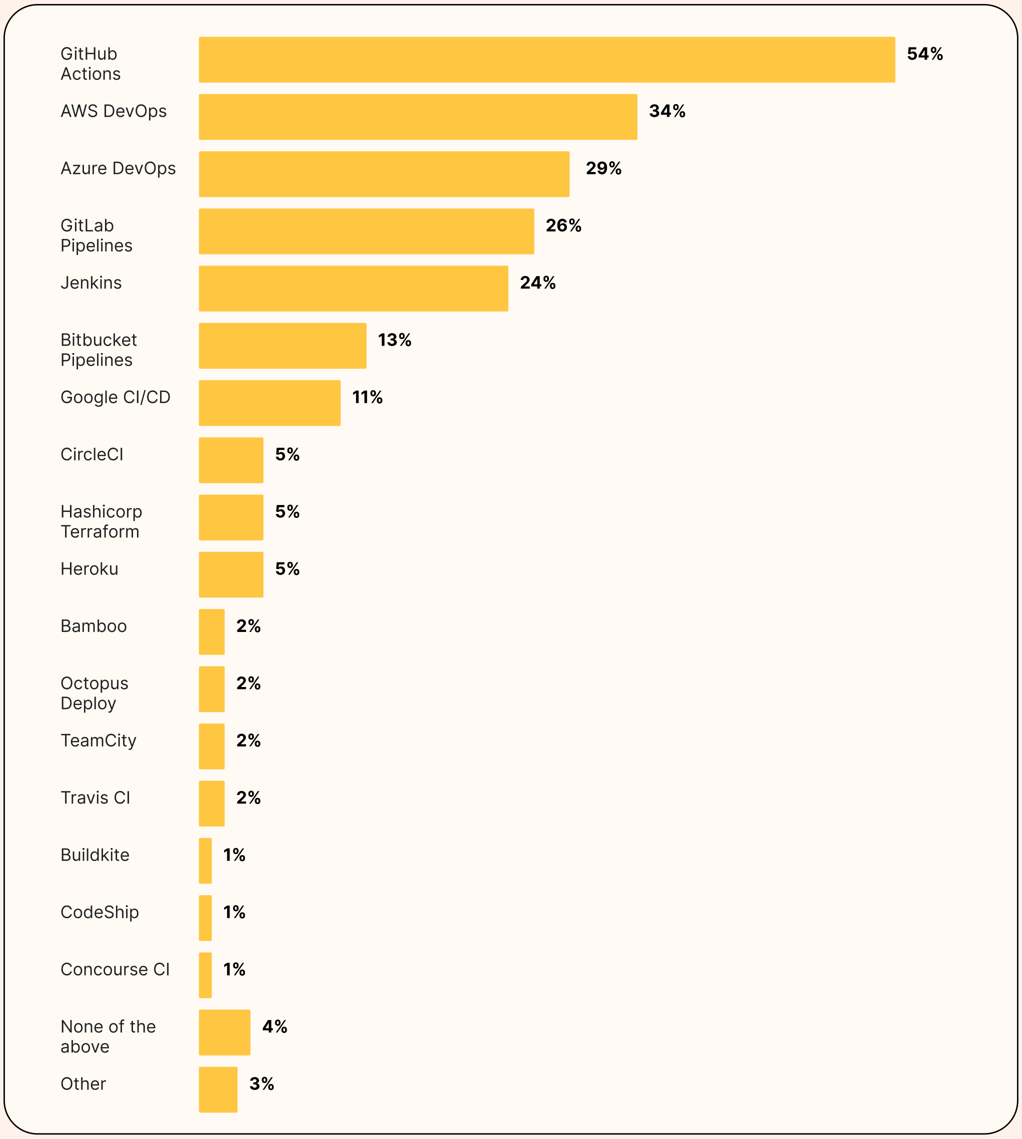
The organizations that solve collaboration don't just eliminate frustration. They unlock the productivity gains that API-first promises, but scattered tooling prevents.

The API tooling and testing landscape: Consolidate or remain fragmented?

The tooling landscape reveals both consolidation and fragmentation across the API development lifecycle.

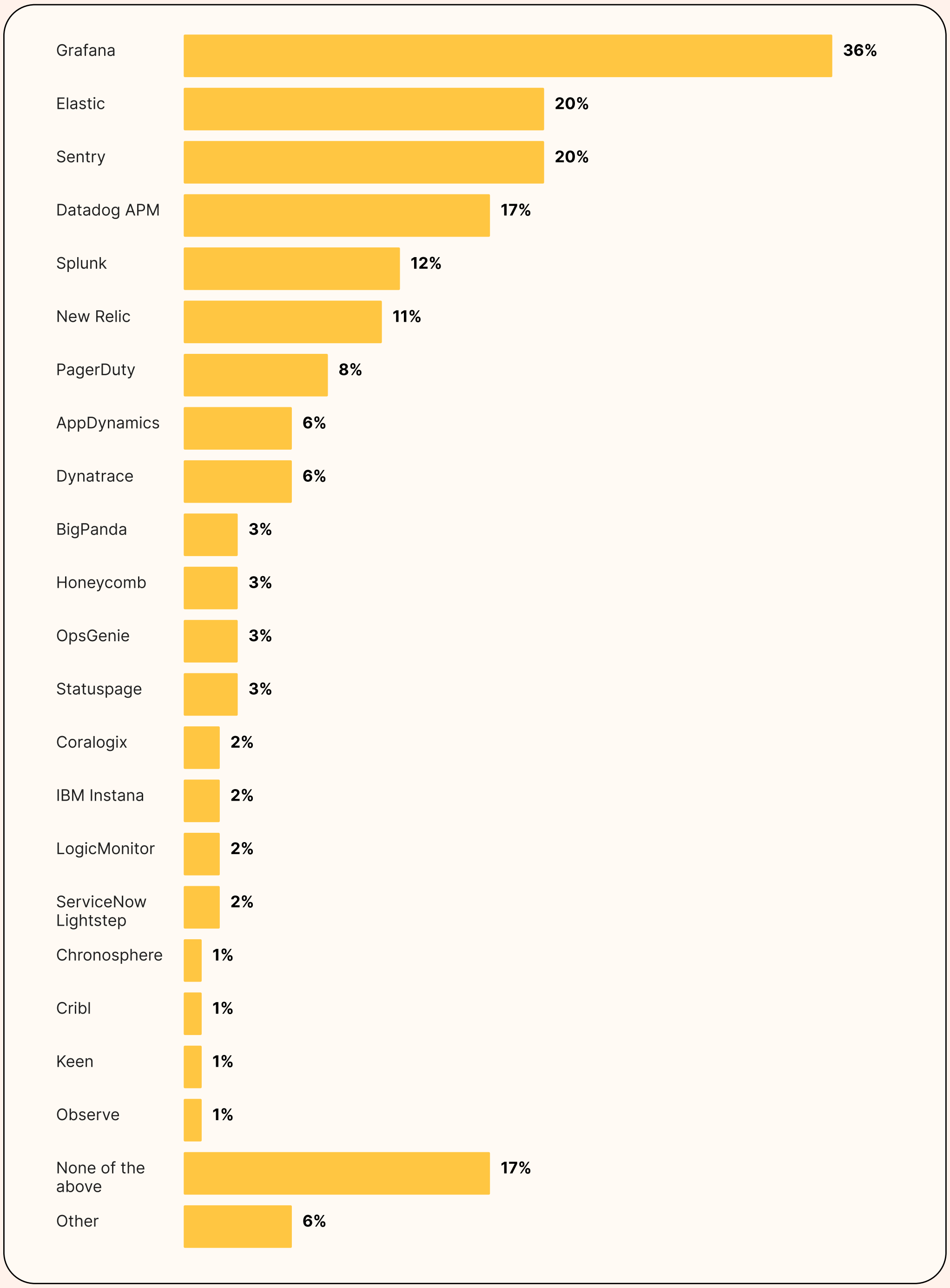
CI/CD tooling

GitHub Actions leads CI/CD adoption at 54%, beating AWS DevOps (34%) and Azure DevOps (29%). This shows that Git-native workflows are winning, with developers preferring tools that integrate directly with their code repositories.

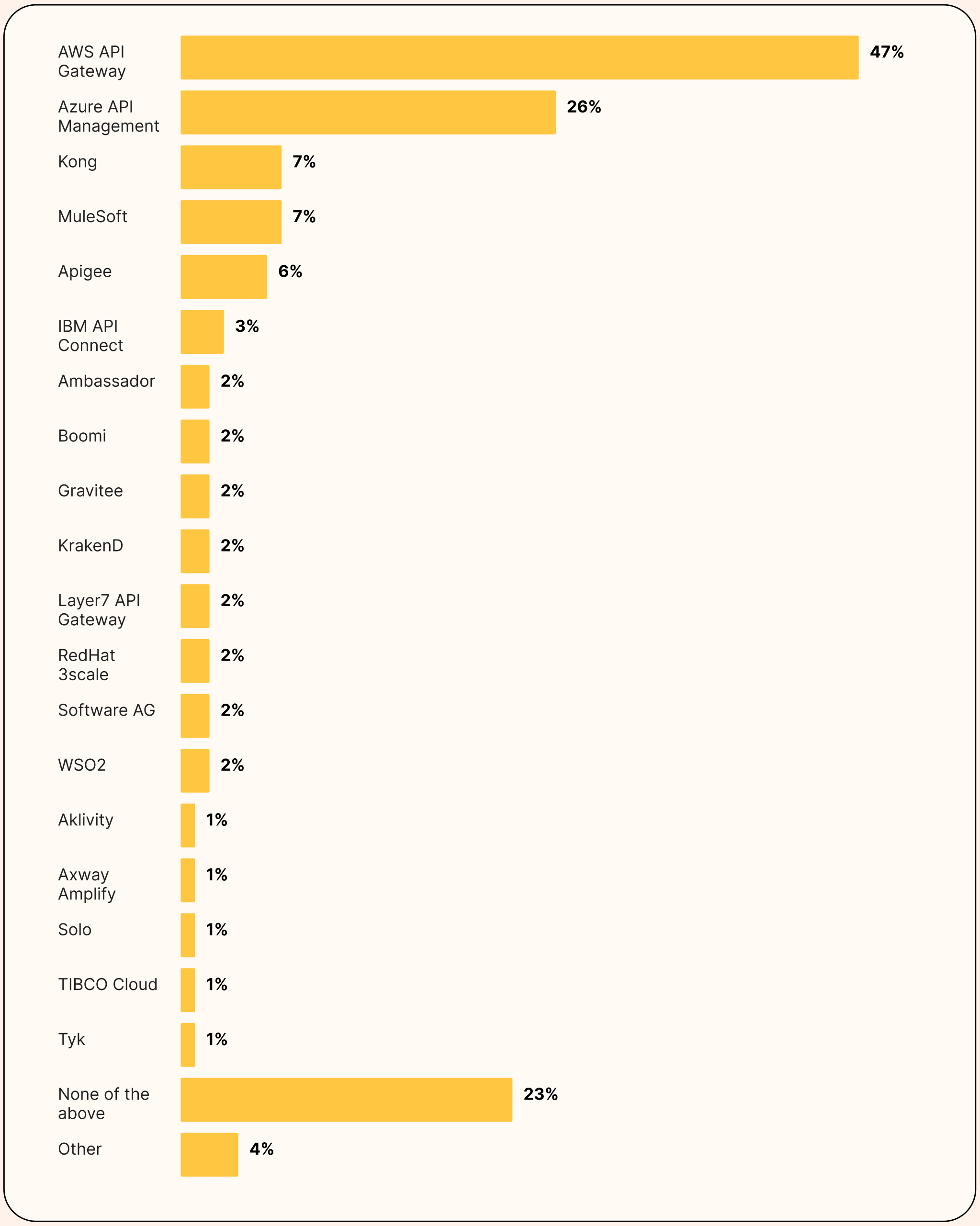


Monitoring and infrastructure gaps

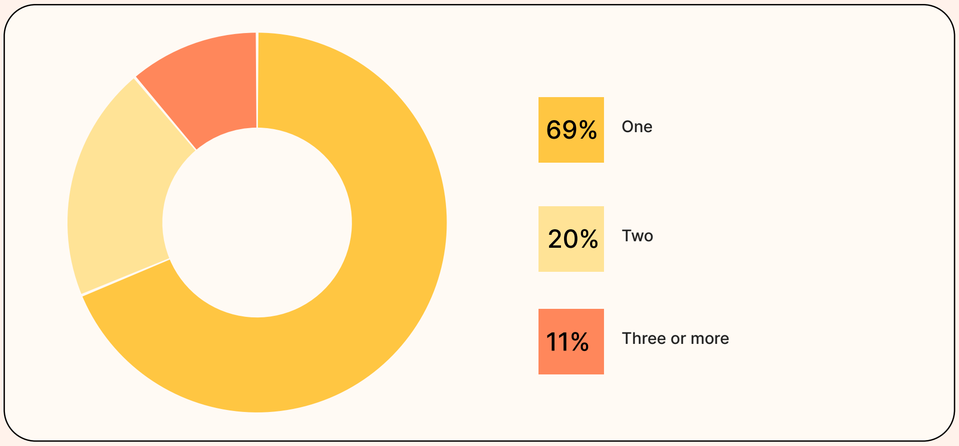
Monitoring reveals significant fragmentation: Grafana leads at 36%, followed by Sentry and Elastic at 20% each. Concerningly, 17% use no monitoring tools at all—a gap that becomes critical as APIs drive more business value and face new security threats from AI agents.



Gateway adoption follows cloud platform preferences but shows diverse tooling choices. AWS API Gateway leads at 47% and Azure at 26%, while 23% use other gateway solutions.



However, the most telling insight is that 31% of organizations use multiple API gateways—20% use two different gateways and 11% use three or more. This multi-gateway reality reflects the complexity of modern API architectures, where different teams, cloud providers, and use cases drive diverse tooling choices.



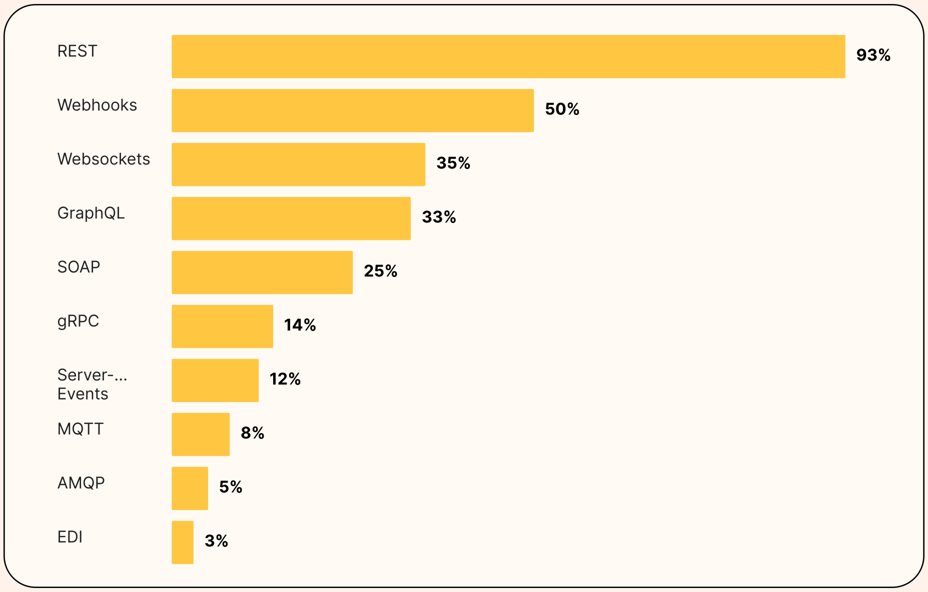
The traditional single-gateway model is becoming obsolete as organizations manage APIs across multiple clouds, different business units, and varied deployment patterns. Organizations need API management solutions that work across gateway diversity, not platforms that require gateway lock-in.

Pro Tip

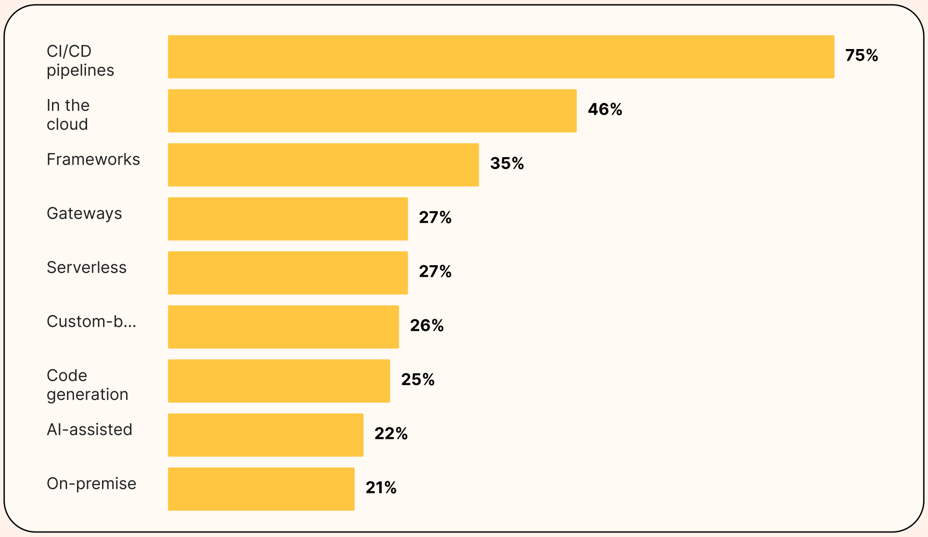
Integrated API management platforms offer a consolidated view of APIs, regardless of their gateway. These tools streamline API discovery, observability, and security, reducing fragmentation.

Learn More →

REST still dominates at 93%, but modern patterns are growing. Webhooks (50%), WebSockets (35%), and GraphQL (33%) show that while REST remains the foundation, teams are adopting additional patterns for specific use cases like real-time communication and efficient data fetching.

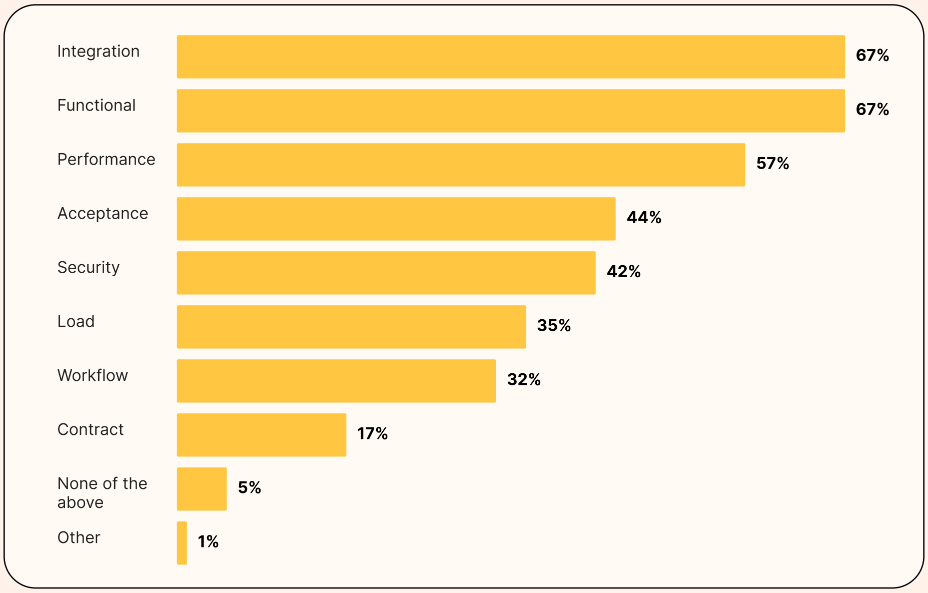


Deployment practices show strong automation adoption: 75% use CI/CD pipelines, making automation the standard. Cloud deployment (46%) and frameworks (35%) indicate varied approaches, but the CI/CD foundation provides consistency across different deployment strategies.



Testing maturity and gaps

Testing practices reveal a maturity gap. Functional and integration testing both reach 67%, showing strong adoption of basic testing practices. Performance testing at 57% indicates growing awareness of scalability concerns. However, contract testing lags at only 17%, a critical gap given the importance of API contracts for both human and AI consumers.

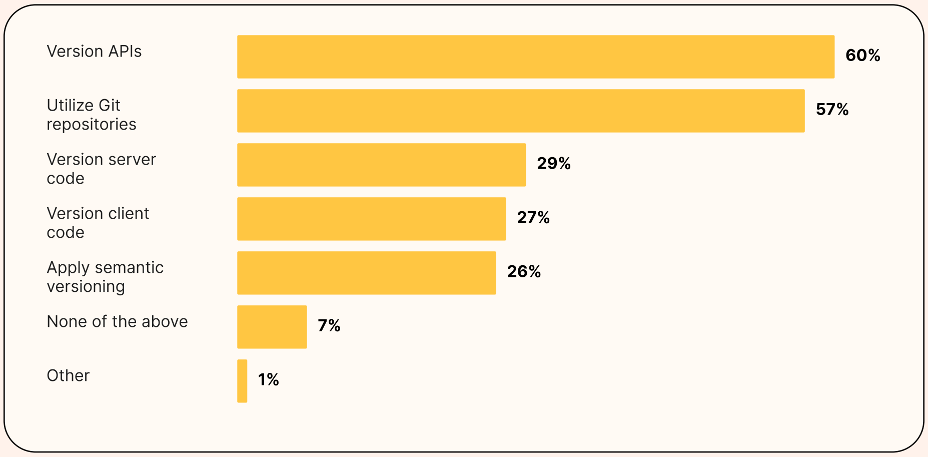


Pro Tip

Don't let your testing practices fall behind with Postman's easy-to-use API testing templates, including contract testing.

Change management inconsistencies

Change management shows inconsistent practices: 60% version their APIs and 57% use Git repositories, indicating version control is standard. However, only 26% use semantic versioning, meaning most teams track changes without communicating the impact of those changes effectively.



Pro Tip

Postman makes it easy to scale your testing maturity. Use Collection Runner for functional and regression tests, and get started with contract testing using built-in templates. Run tests in CI/CD with Postman CLI to catch issues before they reach production.

While some tooling pulls ahead and there's strong CI/CD adoption, the supporting ecosystem is fragmented and reveals an industry built on preference rather than standards. The biggest opportunities for improvement lie in bridging the gaps between basic practices (which are widely adopted) and advanced practices (which remain niche) that enable reliable, scalable API programs.

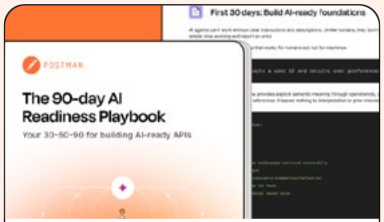
This tooling fragmentation compounds the challenges we've explored throughout this report. When distributed teams struggle with coordination, face new security threats from AI agents, and need to build revenue-generating APIs that work for both humans and machines, inconsistent tooling choices create additional friction. The organizations that will thrive are those that recognize these interconnected challenges and address them systematically.

Future Outlook

The API landscape stands at an inflection point where AI readiness will separate leaders from laggards.

The trends we've examined throughout this report converge on a single reality: organizations must choose to embrace API-first development and AI-readiness or risk falling behind as competitors build more adaptive, secure, and profitable API ecosystems.

Four urgent priorities that will determine competitive positioning in an AI-driven future:



POSTMAN

The 90-day AI Readiness Playbook

Your 30-60-90 for building AI-ready APIs

First 30 days: Build AI-ready foundations

90-Day AI Readiness Checklist

Begin your 90-day transformation to turn your APIs into AI-ready tools.

Learn more →

✓ APIs must be designed with AI agents in mind.

As AI agents become primary API consumers, the APIs designed with machine-readable schemas, predictable patterns, and comprehensive documentation will integrate faster and more reliably than those built only for human consumption.

✓ Security models must evolve for AI consumers.

Traditional security approaches designed for predictable human behavior cannot handle machine-speed exploitation, persistent automated attacks, and credential amplification. Organizations need new frameworks for identifying, monitoring, and protecting against non-human consumers.

✓ Documentation and discovery tools need urgent adoption.

With 55% struggling with inconsistent documentation and 34% unable to find existing APIs, the coordination challenges that plague small teams will only worsen as AI agents require precise, machine-readable specifications to function effectively.

✓ Revenue-driven API strategies require product thinking.

The 65% of organizations already generating revenue from APIs understand that success requires treating APIs as products with developer experience, usage analytics, and lifecycle management, not just technical interfaces.

The choice is no longer whether to adapt—it's how quickly and efficiently you can transform your API strategy to thrive in an AI-driven world.

POSTMAN