POSTMAN
API PLATFORM

The API-First Enterprise

Improving Security, Governance, and Collaboration with Strategic APIOps

Contents

The Rise of API-First and Its Effects

APIs are truly the lifeblood of modern enterprises. According to the 2022 State of the API Report, 51% of organizations spend over half of their engineering bandwidth on APIs, up 40% from 2020. Digital transformation initiatives now hinge on APIs to enable composability and act as connective tissue between services. APIs are also becoming pervasive as companies transform their monolithic codebases into a more iterative, decoupled microservices architecture.

As part of this shift, many organizations are adopting an API-first approach, in which APIs and their consumption model are designed ahead of user interfaces and applications. API-first enables companies to swiftly cater to multiple platforms and devices, and combine and reuse APIs to create innovative user experiences. API-first also reaps advantages such as reduced development costs, faster time to market, and increased software quality.

Establishing a successful API-first approach requires a robust API management foundation. Yet, platform architects often struggle with a disparate assortment of tooling to manage their APIs. These tools span areas like specification generation, documentation and developer dashboards, identity and access management, monitoring, security, and other categories. To get the job done, teams often adopt a blend of proprietary and open source components, which may require significant effort to manage. Without a unified API development strategy, it can be challenging to track and respond to bugs or cater to different types of APIs, whether they're internal, partner, or public-facing.

As the number of production APIs balloons, a fragmented API development approach will continue to hold back API-first initiatives—especially for large enterprises that require increased governance. As a result, many companies are turning to a centralized API platform to localize this tangled mess of tools. A unified workflow makes API development far more productive, as integrated capabilities reduce context switching. By uniting lifecycle operations into a single view, developers have a better understanding of their surface area and the state of their services. Fewer integration requirements also eliminate the time spent worrying about support and versioning for various API-related utilities.

An API platform mindset unifies all development and maintenance priorities into a single holistic platform. These areas include:

- **API lifecycle management:** Tracking and maintaining APIs from development to testing, production, and deprecation from a unified source.
- **API governance and security:** Implementing role-based administrative controls to the engineering environment and production APIs.
- **Unique workspaces for collaboration:** Creating shared collections customizable to the business domain or team makeup.
- **DevOps capabilities:** Iterating API changes and fluidly pushing updates into production across various cloud environments.

We're in an age of distributed, remote work. But just because teams are distributed doesn't mean API development has to be equally fragmented. By taking an API platform mindset, software leaders can break the unhealthy reliance on disparate styles of API management, reduce toil, and maintain guardrails around engineering efforts. Localizing APIs makes discovering an API catalog easier, increasing auditing efforts and avoiding shadow IT. The ability to create unique workspaces on a shared platform also makes onboarding new engineers easier, befitting remote working conditions. In short, through a central platform, enterprise architects can finally see the advantages of API-first materialize.

Why APIs Require Holistic Lifecycle Management

We all know that APIs are becoming ubiquitous across most industries and sectors. But with this newfound prevalence comes a major side effect for IT: the ongoing support of hundreds (if not thousands) of unique software lifecycles at different stages. You don't just ship a microservice and forget about it. These days, agile development is an endless feedback loop between the provider and the consumer. The constant emergence of bugs, unknown vulnerabilities, and the pressure to develop new features means today's software faces continuous evolution.

The API lifecycle is multi-faceted. It includes specification-driven development, quality testing, updating documentation, deployment, production monitoring, iterative versioning, planning for retirement, and more. API lifecycle management is the process of overseeing an API through every step of this evolution. And although organizations have historically adopted bespoke components to oversee individual aspects of this lifecycle, we're now seeing convergence toward unified centers for API lifecycle management.

Centralized API lifecycle management is a boon for developers, as it makes API discovery easier and increases collaboration potential. Let's take a tour of the major steps of an API lifecycle to understand these areas and see how a unified API platform can help.

Design and Development

Implementing an API platform mindset first means adopting an API-first approach that relies on specifications as a single source of truth. By building upon a common specification like OpenAPI, AsyncAPI, or JSON Schema, developers have a trustworthy foundation to implement platform-wide consistency. Development must also support polyglot API design styles that cater to unique use cases. This may include legacy SOAP-based services or standard REST web APIs. Newer types like GraphQL, gRPC, and asynchronous formats also have their place, and teams will require shared specifications and API style guides to follow. Lastly, agile development requires the ability to synchronize changes with Git.

Testing and Monitoring

Various testing and monitoring practices are required to sustain reliable APIs. 68% of API providers utilize functional testing, and 67% perform integration testing. Other testing types, like performance, security, acceptance, and contract testing, are just as important for deploying quality APIs that meet SLAs. With the aid of a unified development platform, engineers can utilize shared specifications for low-code automation mocking and testing. Collaboration in quality assurance is improved as testing processes are more localized; engineers have a better window into production performance and can respond to issues in real time.

Versioning and Sunsetting

APIs must evolve to meet new digital requirements. Yet, at the same time, platform owners want to avoid changes that disrupt client integrations. A shared development environment enables teams to implement consistent versioning policies that avoid unhappy clients. Having end-of-life policies from the upstart and clear communication channels to developer consumers is just as crucial for sustaining internal integrations as it is for updating partner or public integration points.

shutterstock

Shutterstock Leverages API Lifecycle Management for Continuous Alignment

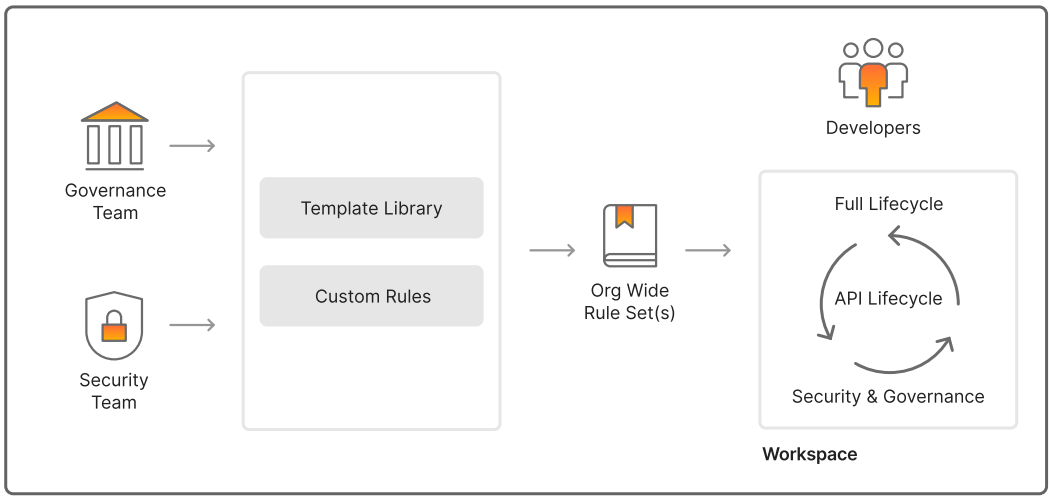
Shutterstock launched its public API over a decade ago, and since then, it has transformed its business through APIs. As of mid-2021, Shutterstock had 8,000 live partners using its API to integrate stock photography and videos into external applications. According to Alex Reynolds, former VP & GM, Platform Solutions at Shutterstock, API management has been critical to achieving this success. “I think of API management as the visibility layer—without that, we're completely blind,” he said. For Shutterstock, this centralized visibility was vital to de-silo APIs from technical groups to help align with business objectives. “Knowing you can have such control gives comfort that you can introduce things gradually.”

API Security and Governance Need Central Control

In recent years, cybersecurity has become a top concern for all of IT. We hear of new breaches and zero-day vulnerabilities on a near-daily basis, some of which carry dire consequences for the software supply chain. And when it comes to web APIs, security incidents are all-too-common. The 2022 Postman State of the API Report found that 20% of organizations experience an API security incident at least once a month. In Q3 of 2022, Salt Labs reported a 117% increase in malicious API traffic over the past year.

So, what factors are driving these security woes? Well, OWASP ranks Broken Object Level Authorization as the topmost security vulnerability for APIs, meaning that many over-privileged API endpoints aren't enforcing the proper roles for their consumers. Internal APIs are often overexposed on the public web, left misconfigured without appropriate access control. GraphQL brings new introspection concerns and additional authorization hurdles when orchestrating multiple services. And since API development has become so distributed, many organizations don't have a solid handle on the number of APIs they support, meaning they can't discover and audit their surface area effectively.

Postman Platform



Centralize and Shift API Security Left

Centralizing API development efforts with a unified platform is one way to help avoid many of the aforementioned security threats. By running all APIs through the same shift-left security testing and schema validation, development teams can scan for vulnerabilities and enforce common rulesets before they reach production. To meet regulatory compliances, custom policies can be created once and then shared across multiple development teams.

Adopting common API security capabilities across an organization also has the added benefit of standardizing how identity and access management (IAM) is applied over a suite of services. Customized rules and SCIM-based provisioning can further delegate identity and reduce the threat of broken access control. What's more, having a unified view of your API ecosystem is crucial to enact runtime security and continuous monitoring.

Adopt Platform-Wide Governance

A centralized development platform also decreases threats by enforcing common guardrails around all development workflows. This includes ensuring the API development platform itself is not prone to abuse. Too often, cloud-native tools are unintentionally misconfigured or left with insecure defaults. Developers can just as easily leak API keys and secrets on accident. To complicate things, API platform teams may loop in junior developers, QA teams, or business folks with incomplete security training.

It's therefore essential to enact role-based access control for the API development environment. Super administrative roles should be used to differentiate fine-grained responsibilities within an organization. By unifying API development, operators can also enforce tighter governance in the form of API style guides. Finally, shared API observability makes reporting more transparent and provides a fuller picture of active endpoints, informing auditing efforts.

How Boy Scouts of America Handles API-First Security and Governance

The Boy Scouts of America is the U.S.'s largest scouting organization, comprising over one million youths and over fifty thousand units. The Boy Scouts of America has also embraced API-first as a core tenant. As a non-profit short on resources, the API-first approach helps amplify the reuse, simplification, and standardization of its IT assets. Simultaneously, engineers must protect a lot of sensitive data. “Security is pretty top of mind as an organization,” said Vijay Challa, former Chief Information Officer of The Boy Scouts of America. According to Challa, engineers adopt role-based protection and short-lived JWT tokens. Their API management techniques help enable a zero-trust mindset from the onset, automatically denying every request unless it's whitelisted. An investment in observability has also helped the team maintain stability.

Shared Workspaces Streamline Development Collaboration

Effective collaboration is necessary for any technical team to function. And in software development, this is no exception. The problem is that organizations must simultaneously cater to new working conditions if they hope to retain talent. The 2022 State of the API Report found that remote work is “very important,” according to 72% of developers and API professionals. Additionally, 68% of developers expect to work from home at least four days a week in the coming year.

Today's software team makeups can vary dramatically. The current hybrid remote workplace now includes developers from multiple geographic locations. Distributed team members often work asynchronously in various time zones or employ a combination of off-site and on-site hybrid work. Also, modern software teams are often cross-functional and may include programmers, architects, cloud operators, and data analysts working on a particular problem domain.

Catered API Workspaces

The problem with today's development teams is that their workspaces rarely match the team makeup. This is an issue with APIs, as these integration points tend to have many types of stakeholders. Furthermore, the number of APIs a single organization supports (and consumes) is rapidly increasing. Simply maintaining a static list of all internal services and their corresponding OpenAPI Specifications doesn't supply context for multi-disciplinary teams working on a particular domain.

One solution is to adopt a central API development platform that provides the flexibility to segment views for different consumer groups. For example, suppose a team building out a payments microservice only wants to interact with endpoints associated with that domain. A unique workspace could be generated to collect common HTTP endpoints, methods, parameters, and sample requests and responses for that particular area. Similarly, private domain-specific workspaces could be catered to other teams within the organization. Workspaces could be exposed to select partners to aid their integration efforts or be shared publicly to assist API-as-a-product initiatives.

By adopting a central API development platform, organizations can increase the visibility of their entire API catalog. And, through workspaces, they can generate private API networks that correspond to specific groups. Collecting common requests in this way can significantly aid testing efforts and increase knowledge retention when onboarding team members.

WERNER ENTERPRISES

How Collections Aid Quality Assurance at Werner

Werner Enterprises is a logistics and transportation company that employs over 13,500 employees and contractors. Their software is a critical component of their business as it ensures the delivery of millions of goods and services across North America. “APIs are incredibly critical because they're the bridge from the application to the database,” said Tim Velasquez, Software QA Manager at Werner Enterprises. “You can have a phenomenal frontend, but if your APIs are not sound, it becomes volatile.”

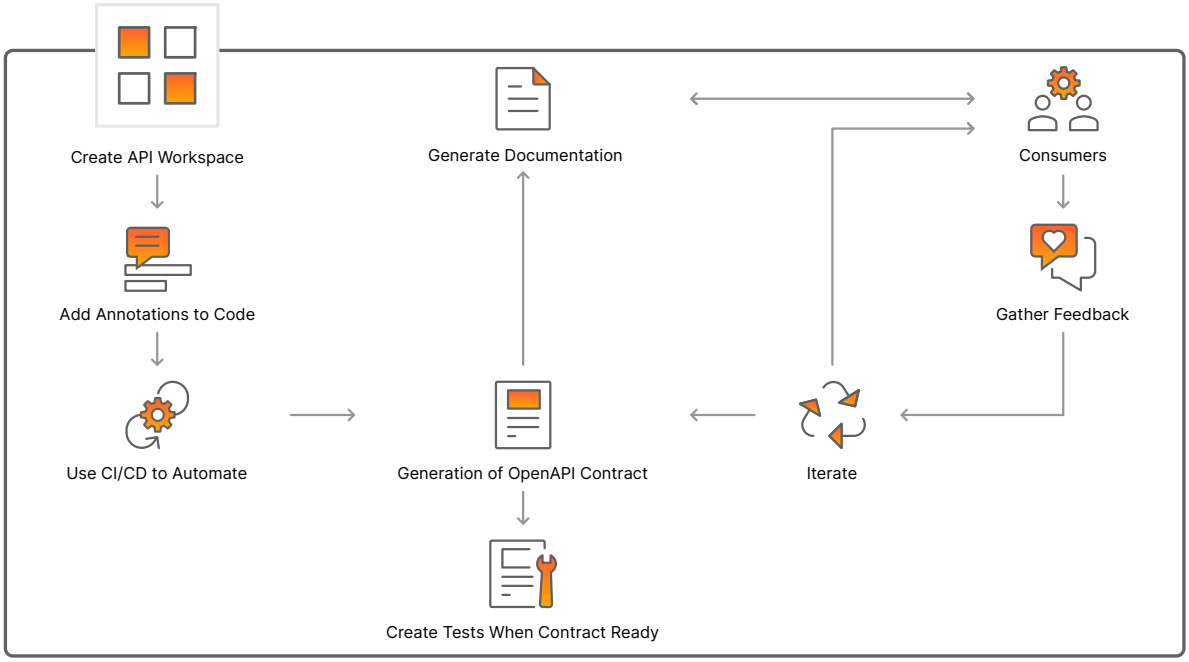
To ensure API stability and avoid defects, Werner employs quality engineers to maintain these integration points. Part of their process hinges on Postman Collections. Developers create collections to inform unit testing and baseline functional testing. After developers codify the surface area of the API, they hand these collections to QA personnel to derive specific business scenarios and perform end-to-end testing. “Having shared workspaces using Postman helps retain that knowledge for us,” said Velasquez. “When the next person comes in, they have that knowledge.”

Unified DevOps for the Win

It's not enough these days to release great software and forget it. Neither is it pragmatic to keep development and operations separate. A unified DevOps approach is the hallmark of modern software release strategies, as it retains platform agility and enables continuous deployment, which are both necessary to scale a microservices architecture. Yet, there are many roadblocks in the way of DevOps, especially concerning API-first development and iteration.

For example, varying API deployment models can make DevOps challenging to enact consistently throughout an organization. The 2022 State of the API Report found that 61% of API practitioners adopt CI/CD pipelines as their most common deployment tool. This is followed by deploying APIs in the cloud (38%), frameworks (35%), and custom-built deployment methods (27%). Most companies have disparate CI/CD tooling that is chosen on a team-by-team basis. Furthermore, many organizations are pressed to support a combination of multiple clouds, on-premise, and edge environments, each with unique deployment patterns.

Code-Led API Development



A Focused Approach to APIOps

Supporting today's tangled mess of distributed microservices can easily get out of hand. And the ubiquity of APIs means that DevOps teams are frequently managing change for countless endpoints. Over one-third of organizations are now deploying APIs to production between once per week and once per month. 19% of API-first leaders finish this process in a day or less, highlighting the importance of rapid deployment. Streamlining developer experience is critical to realizing a rapid change cadence, but it's impossible without the right automation in place.

A centralized development environment could alleviate the burdens associated with API DevOps. By uniting efforts around a single platform, you can integrate with source control from a unified center. A development platform can also abstract specific deployment processes, making it easier to simultaneously support cloud, on-premise, or hybrid deployments. A centralized API development platform also supports platform engineering ambitions, such as enhancing cloud observability and application performance monitoring to meet SLAs. Greater API observability also enables engineers to respond quickly to incidents.

stripe

Stripe's Iterative API Change Strategy

Stripe is a technology company that builds economic infrastructure for the internet. Stripe handles 5,000 requests per second to its API, which is the backbone of its service. Using Stripe, developers can integrate online payments and run financial operations within their applications. Stripe is well-known for its investment in developer experience. But Stripe has also embraced highly rapid change, releasing 13 to 14 new versions of the API every day. “You're leaving money on the table, and you're building up tech debt if you're not incrementally improving your stack,” said Chris Traganos, former Developer Advocacy Engineering Manager at Stripe. Prominent API companies like Stripe continuously iterate and must have processes to quickly deliver APIs and support older versions to avoid breaking changes.

Uniting the API-First Enterprise

Digital enterprises are continually adopting APIs within new projects. The 2022 State of the API Report found that the number of APIs will increase or stay the same over the next 12 months among 89% of organizations. The API-first approach is becoming more widespread as teams seek to build digital technology with reusability in mind from the onset. And arguably, API-first companies are well-positioned to weather economic uncertainties.

Yet, for all its strengths, a company-wide API-first approach is still challenging to implement. It requires forethought throughout each stage of the API lifecycle. It also requires security planning to ensure safe implementations—for runtime APIs as well as development and testing environments. Teams must also streamline their internal processes around collaboration and DevOps to retain an agile footprint.

A unified API platform is necessary to deliver all the above points. Converging disparate API-related tooling into a single platform can guide API-first alignment through the use of shared specifications and style guides. Engineers can make ongoing maintenance more predictable by centralizing API lifecycle duties, such as monitoring and versioning. Additionally, creating workspaces around domain-specific requests helps cater to cross-team development and testing. Lastly, having a single platform to oversee DevOps can benefit internal development by increasing automation and delivering more reliable services.

When it comes down to it, the benefits of API-first are really developer-first benefits. This is certainly the case at PayPal, which relies heavily on APIs to power its business. With the goal of creating a self-service API platform, PayPal invests heavily in improving their developer experience to reduce the time it takes to go from checking out the documentation to reaching “Hello World.” Also, to maintain quality and publish rapidly, engineers at PayPal utilize an API test suite that runs in Postman as part of their CI/CD process, explained Helen Tatum, Product Lead, Developer Experience at PayPal, on an episode of Breaking Changes.

Investing in developer experience can result in increased developer satisfaction and less turnover. This, in turn, improves overall morale and knowledge retention within a software team. Investing in a centralized API platform certainly encourages these outcomes and instills an abstraction layer that future-proofs development against inevitable technology shifts. The end result is higher quality, usable APIs, and more secure and reliable end-consumer digital experiences for all stakeholders involved.