APIFORTRESS

Evolving from API Testing to API Monitoring

Patrick Poulin

APIFORTRESS

One Reality

1) Engineers and QA teams have **automated** API testing to keep up with increased releases

2) QA teams hold the most domain knowledge of the APIs' goals and functionality

Two Truths

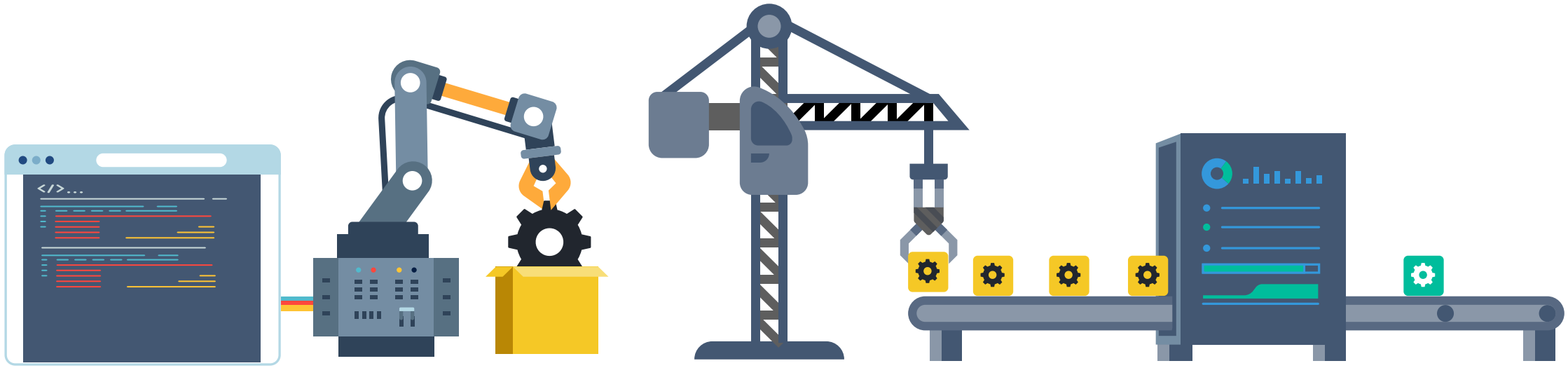
1) Proper API testing should include monitoring of the APIs against multiple environments including production

2) OPs teams are typically in charge of "monitoring," but lack testing experience and domain knowledge of the API

APIFORTRESS

API Testing Automation Drives CI/CD Success

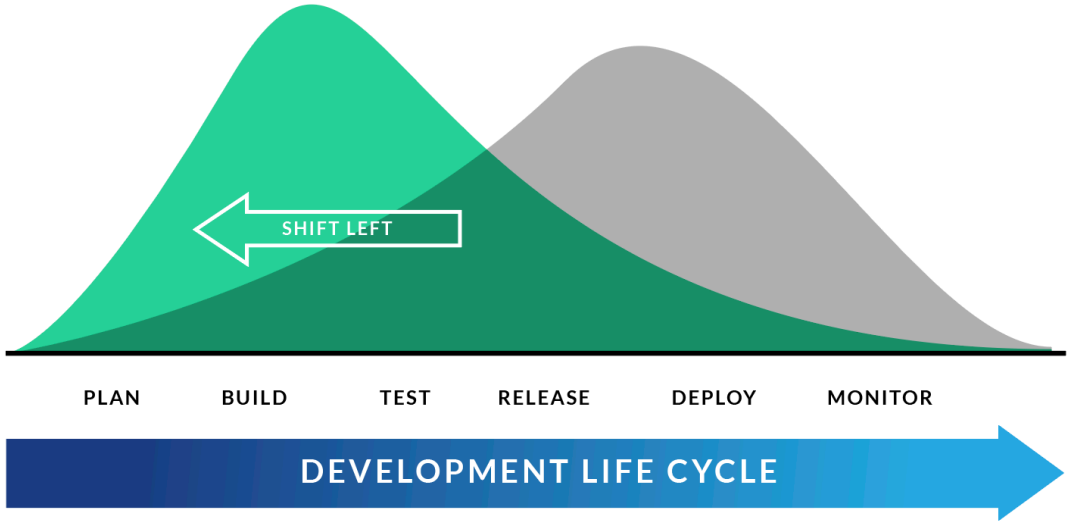
Companies must accelerate go-to-market — even as test cases become more complex for business success. A CI/CD pipeline may be the solution. But a successful pipeline needs API testing to unlock its true potential for quality at speed.



APIFORTRESS

API Testing Automation Must Evolve For Today's Speed And Quality Needs

That is why the market for API testing automation has grown so rapidly: There are now many platforms and libraries to facilitate API testing automation. However, API testing automation tools have not kept up with demands for even greater speed and integration complexity. As a result, many companies are trying to run automated tests continuously against various environments. This is **API Monitoring**.

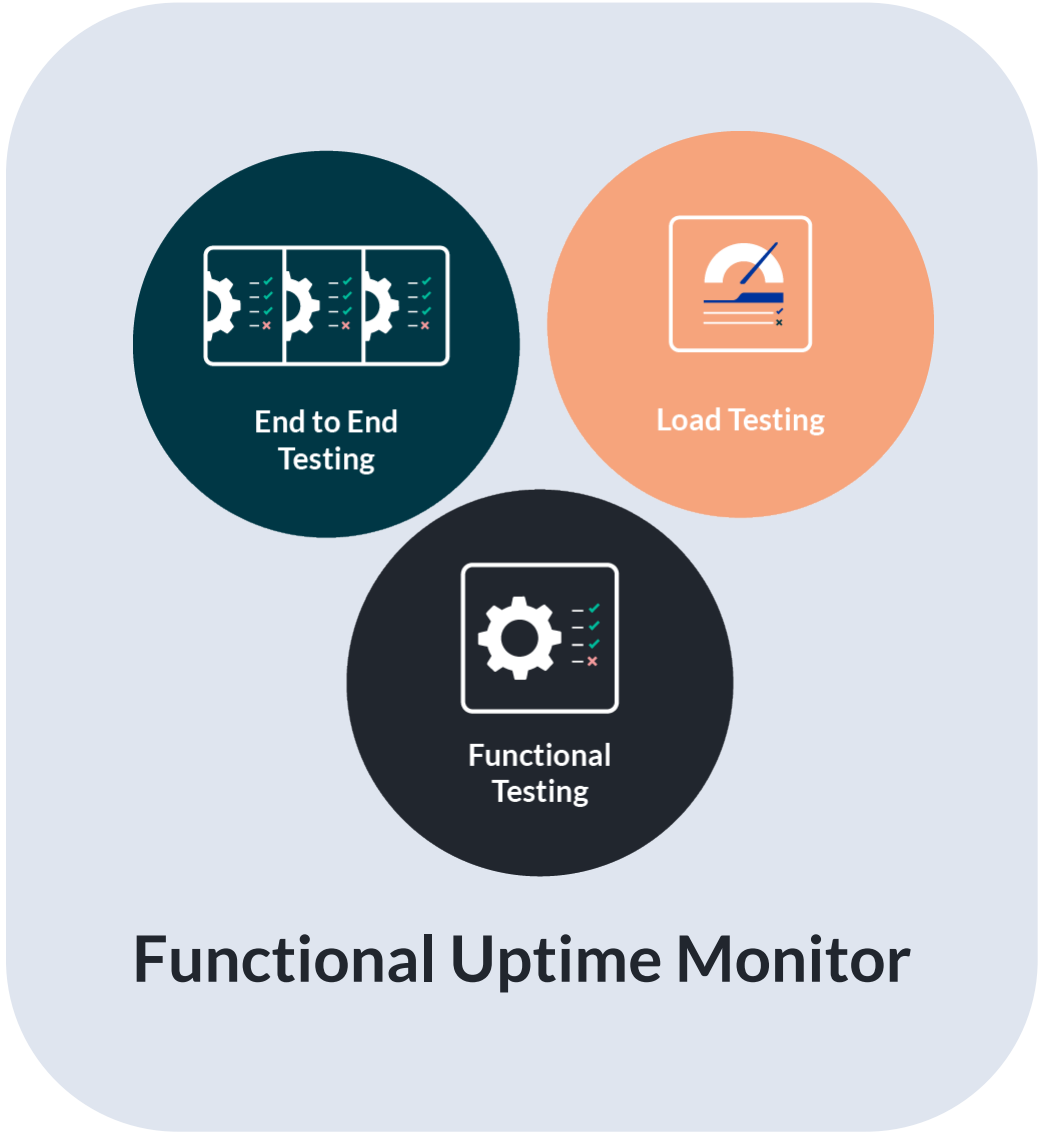


API Monitoring

While many platforms claim to offer monitoring, they cannot monitor for API functionality — and therefore, cannot support today's quality-at-speed. That means these platforms cannot check for *functional* uptime, resulting in too many API defects and vulnerabilities going to production.

Functional Uptime Monitoring

That is why the market for API testing automation has grown so rapidly: There are now many platforms and libraries to facilitate API testing automation. However, API testing automation tools have not kept up with demands for even greater speed and integration complexity. As a result, many companies are trying to run automated tests continuously against various environments. This is **API Monitoring**.

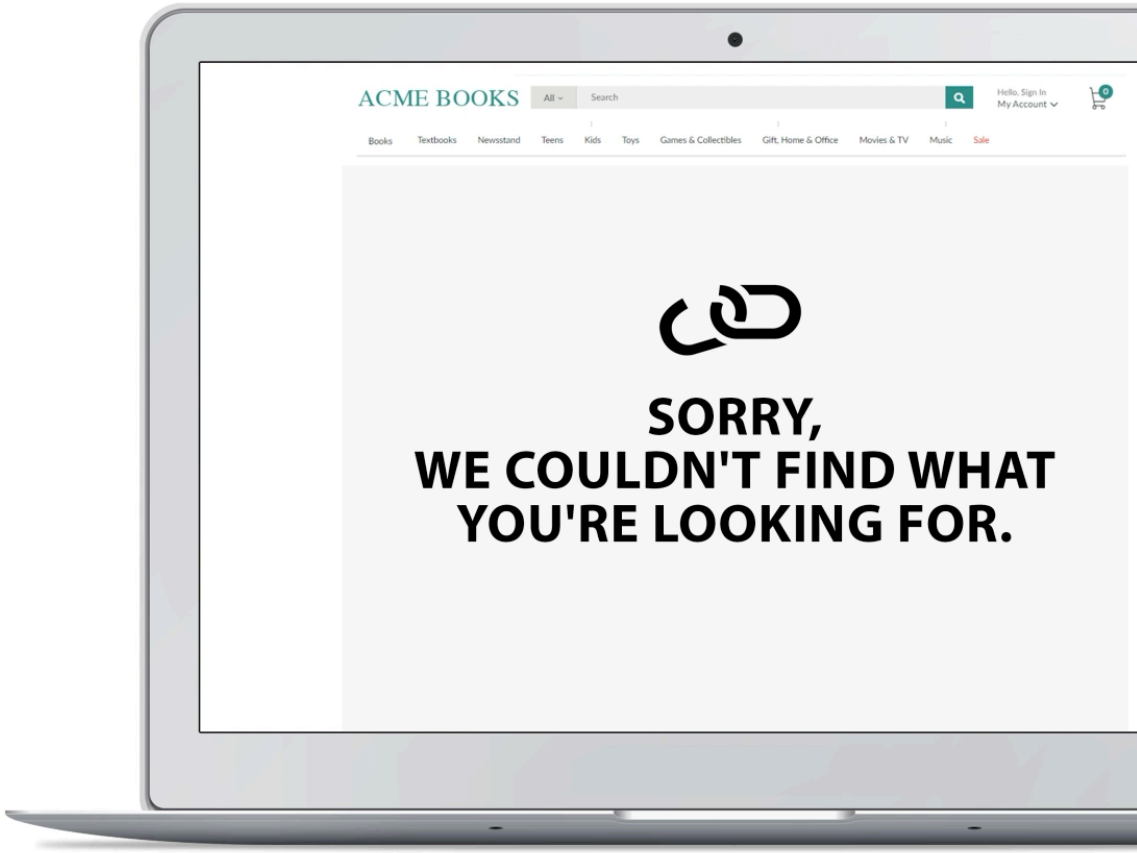


APIFORTRESS

CASE STUDY:

Cache Me If You Can

- A large book publisher created two partner APIs: one which lists all “active” ISBNs and another that gets product details for individual ISBNs
- The publisher’s legacy monitors reported 100% uptime
- Within 2 weeks of deploying API Fortress, the publisher was alerted to hundreds of errors and 404 soft errors
- With enhanced coverage and detailed reporting from API Fortress API tests, the publisher realized that their API gateway was caching endpoints to improve performance, but was not refreshing the cache often enough, resulting in hundreds of bad APIs
- Today, the publisher checks for functional uptime via API Fortress to catch API problems early

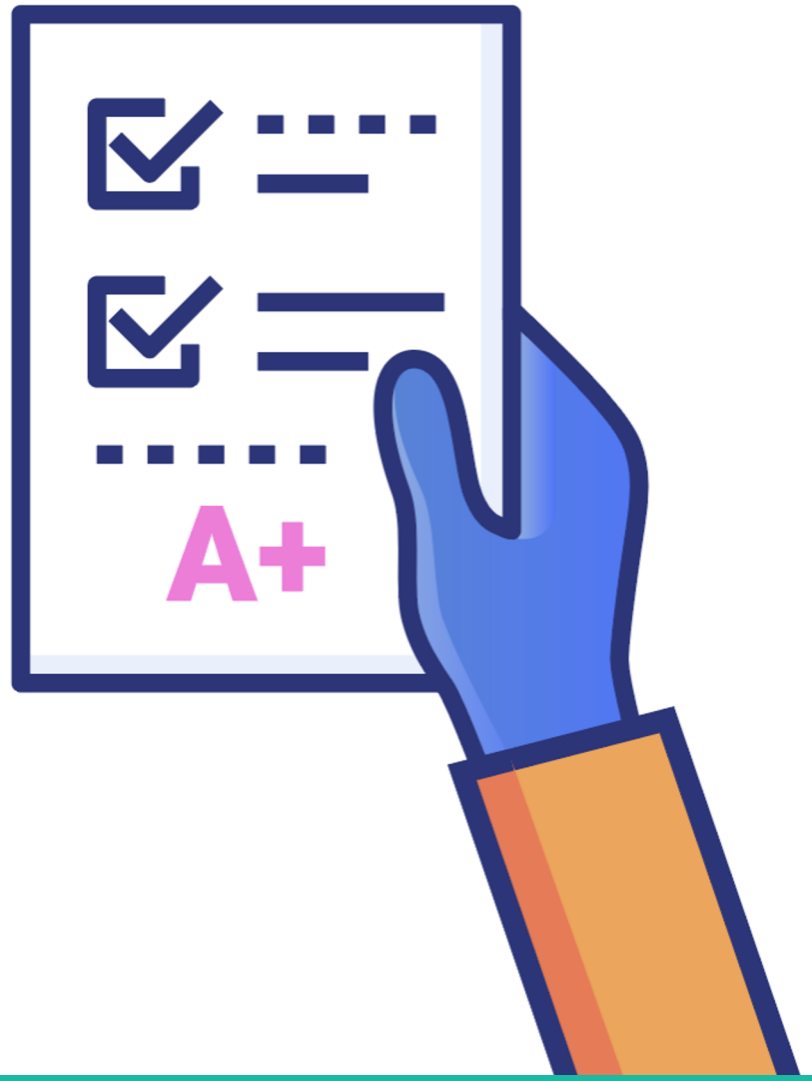


Uptime is NOT Functional Uptime

Why did the publisher's legacy monitoring fail? It was not doing enough. They did not monitor the use case of their average API consumer. They should have deployed a unified monitor that leveraged their functional, integration, end-to-end, and performance tests.

One way to think about proper functional API monitoring is to imagine that you are a parent concerned about your child's education. How do you measure their progress? Would you use home room attendance as your measure or KPI?

Or would you use their test results? A big part of proper API monitoring is to standardize API testing across your organization. Coverage should extend to every object, every bit of data, positive and negative testing, data-driven testing and more. That is a proper monitor. Yet companies still struggle to standardize proper testing and monitoring — a symptom of gaps in processes and technology.



Technology Gaps

Popular monitoring platforms today share one or more common shortcomings, including:

- Limited to Synthetic Testing
- Poor Test Reusability
- Insufficient API Privacy (third-party clouds)
- Incomplete Test Intelligence

APIFORTRESS

AlertSite

BlazeMeter

DATADOG

dynatrace

New Relic®

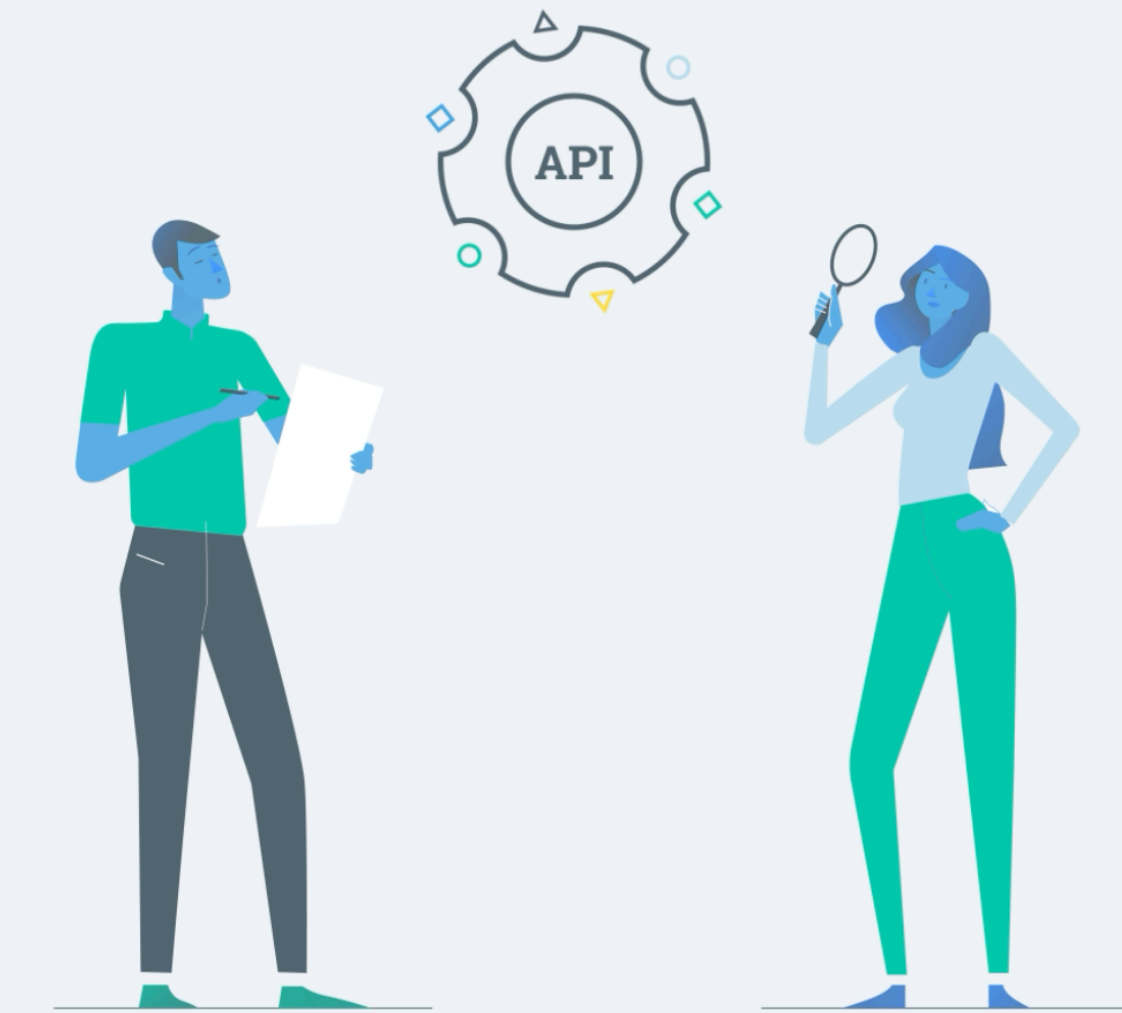
Process Gaps

A proper API test should validate an entire user flow. For example, an e-commerce website could significantly improve quality by running API tests that capture the entire flow across real-world behaviors such as searching for an item, adding to cart, and checking out. Three siloed tests could never capture the flow, because they cannot capture how the behaviors (when strung together) will impact function.

Conclusion

API test automation is here, and we should embrace it. The problem is that some companies view automation as a path to only reducing QA costs. Instead, automation should be helping to expand QA and test engineer roles to help reduce risk and accelerate time-to-market. Monitoring is the clearest first evolution on this path to digital transformation.

Testing professionals are critical for system reliability, and that shouldn’t only come into play on builds. QA and testing teams can leverage their skill sets for functional API monitoring throughout the lifecycle - from staging to preprod and production. API issues are often data/analytics issues: testing professionals know how to solve these issues.



Thank You!

Start a free trial at www.APIfortress.com