A webservice here, an additional frontend there, but how do we ensure quality?

BC/NAV TECH DAYS 2024

mibuso.com

Tobias Fenster

SPEAKER INTRO

TOBIAS FENSTER

Business

- Managing Director at 4PS Germany, part of 4PS by Hilti
  - BC ISV for the construction industry

Community

- Microsoft Regional Director and MVP for BC and Azure
- Docker Captain

Socials etc

- tobiasfenster at Twitter and LinkedIn
- tobiasfenster@hachyderm.io at Mastodon
- Blog at tobiasfenster.io, incl. “Window on Technology” podcast

Intro

Microsoft Azure
OpenAI

QUALITY?!

Quality

Is this your approach?

Might turn really uncomfortable some day...

THIS IS FINE.
I'M OKAY WITH THE EVENTS THAT ARE UNFOLDING CURRENTLY.
THAT'S OKAY, THINGS ARE GOING TO BE OKAY.

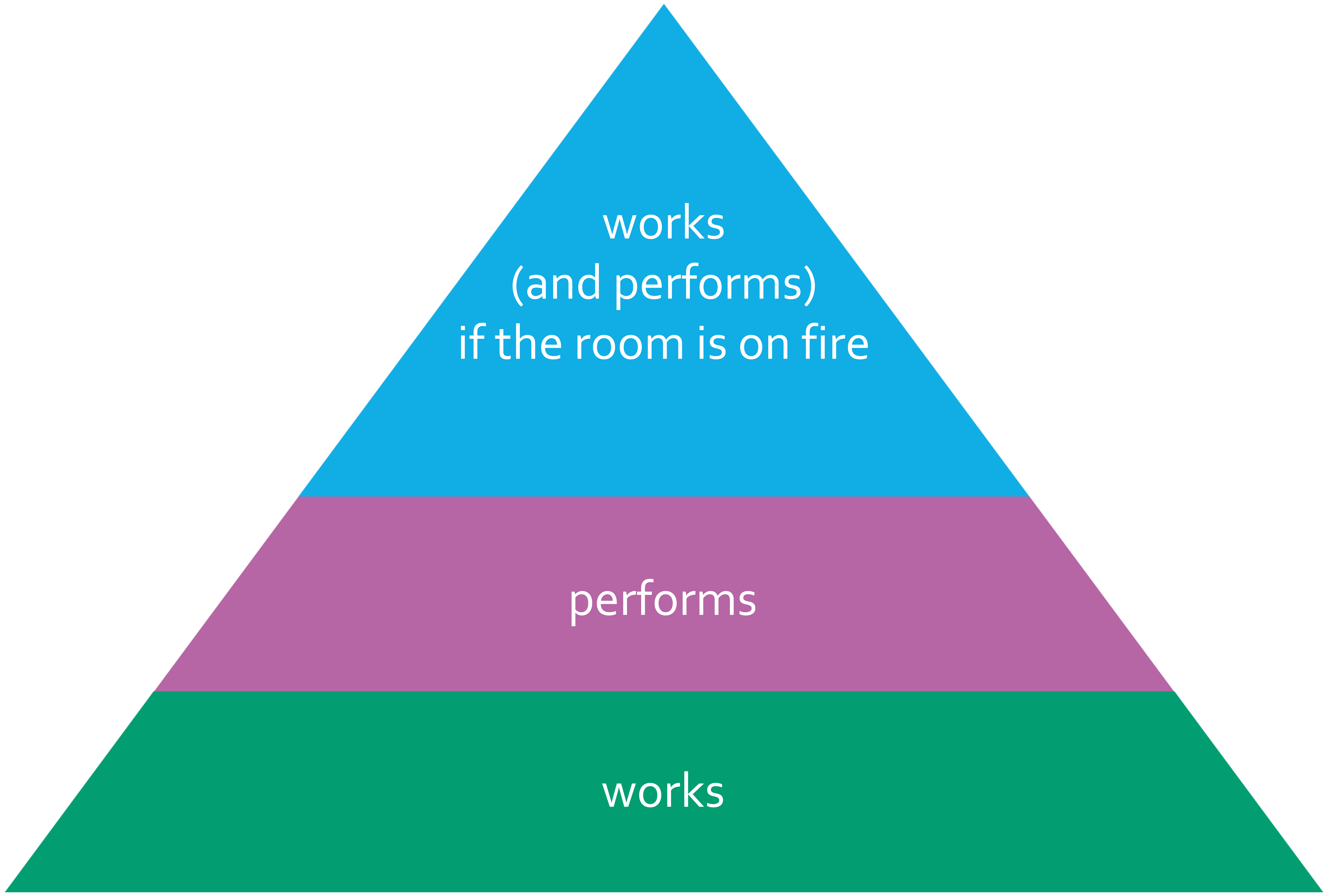
BC/NAV TECH DAYS 2024
mibuso.com

https://knowyourmeme.com/memes/this-is-fine

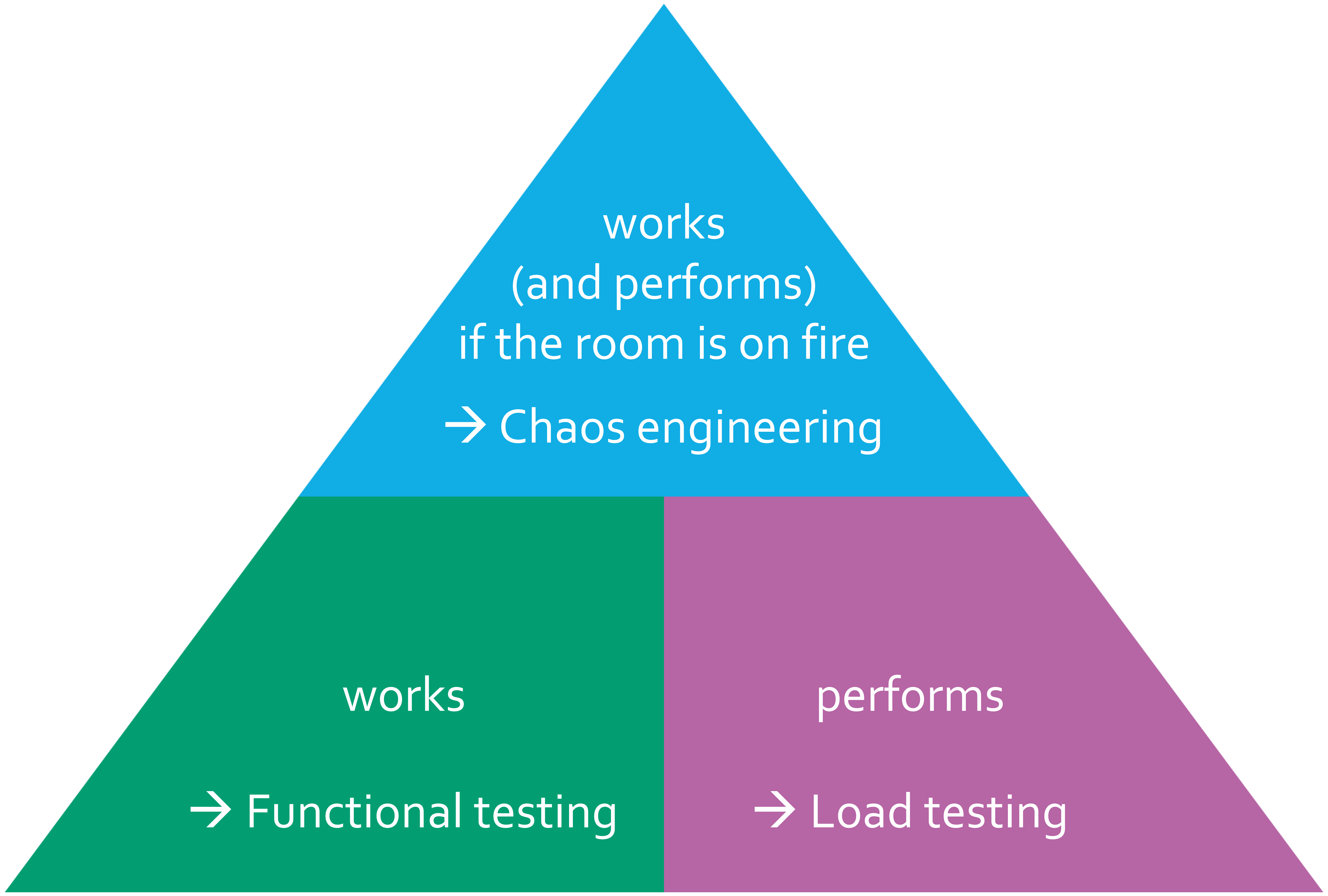
Quality

IEEE says: Software quality refers to the degree to which software conforms to its requirements and meets the needs of its users. It is formally defined as “the capability of a software product to satisfy stated and implied needs when used under specified conditions.” Another definition states that software quality depends on “the degree to which those established requirements accurately represent stakeholder needs, wants, and expectations.” High quality software meets its requirements, which in turn should accurately reflect stakeholder needs. Quality is about aligning the software with both its formal requirements as well as true user needs.

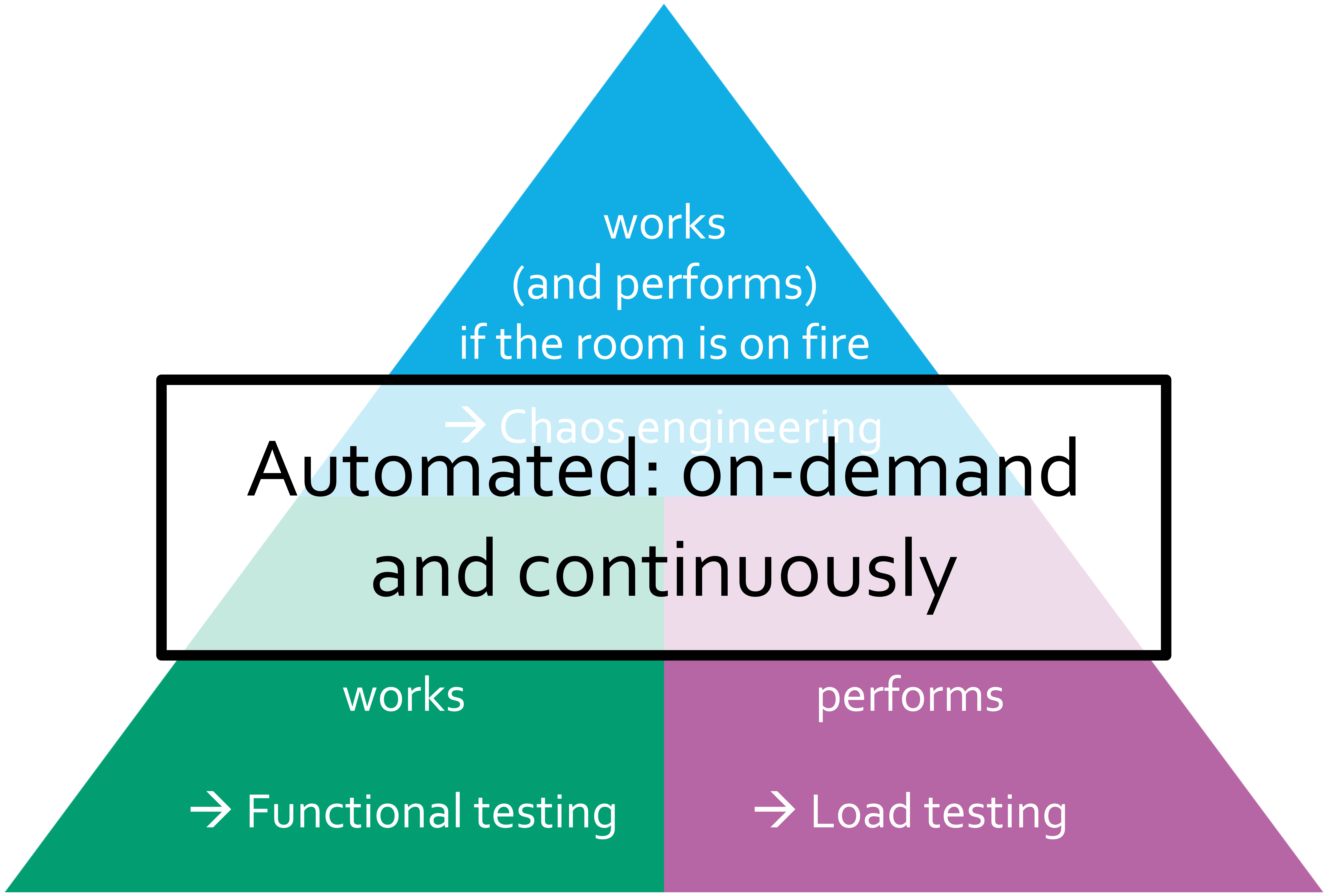
Quality



Quality


works
(and performs)
if the room is on fire
→ Chaos engineering
works
→ Functional testing
performs
→ Load testing

Quality



Automated: on-demand and continuously

Structure

Functional testing: **Playwright**

Load testing: **Azure Load Testing (JMeter)**

Chaos engineering: **Azure Chaos Studio**

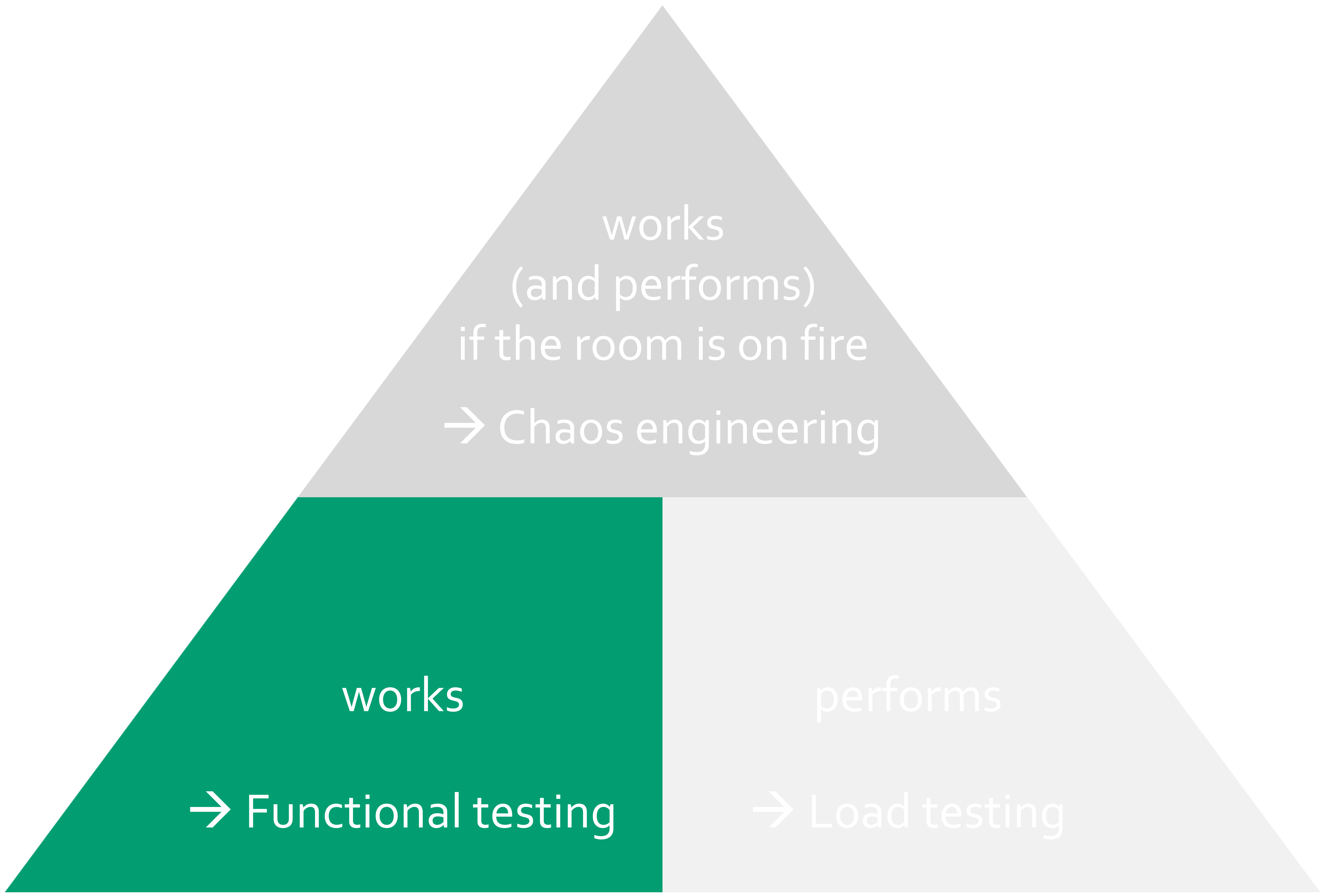
For each of the topics

- Intro
- Practical scenario and demo incl. automation
- How to repro at home

Combined questions in the end (or in between for the very unlikely and surprising case that something in Azure is slower than expected)

Warning: Some experimentation on my side, not a ton of practical prod experience

Quality


works
(and performs)
if the room is on fire
→ Chaos engineering
works
→ Functional testing
performs
→ Load testing

Functional testing: what

“Make sure the application provides the expected functionality”

Validate

- (core) processes and features
- status, e.g. what is visible / available and what isn’t
- results: success and failure
- using different datasets, typically inputs

Key requirement: Know what to test → test plan. Talk to my colleague Luc van Vugt if you want to know how

Functional testing: how

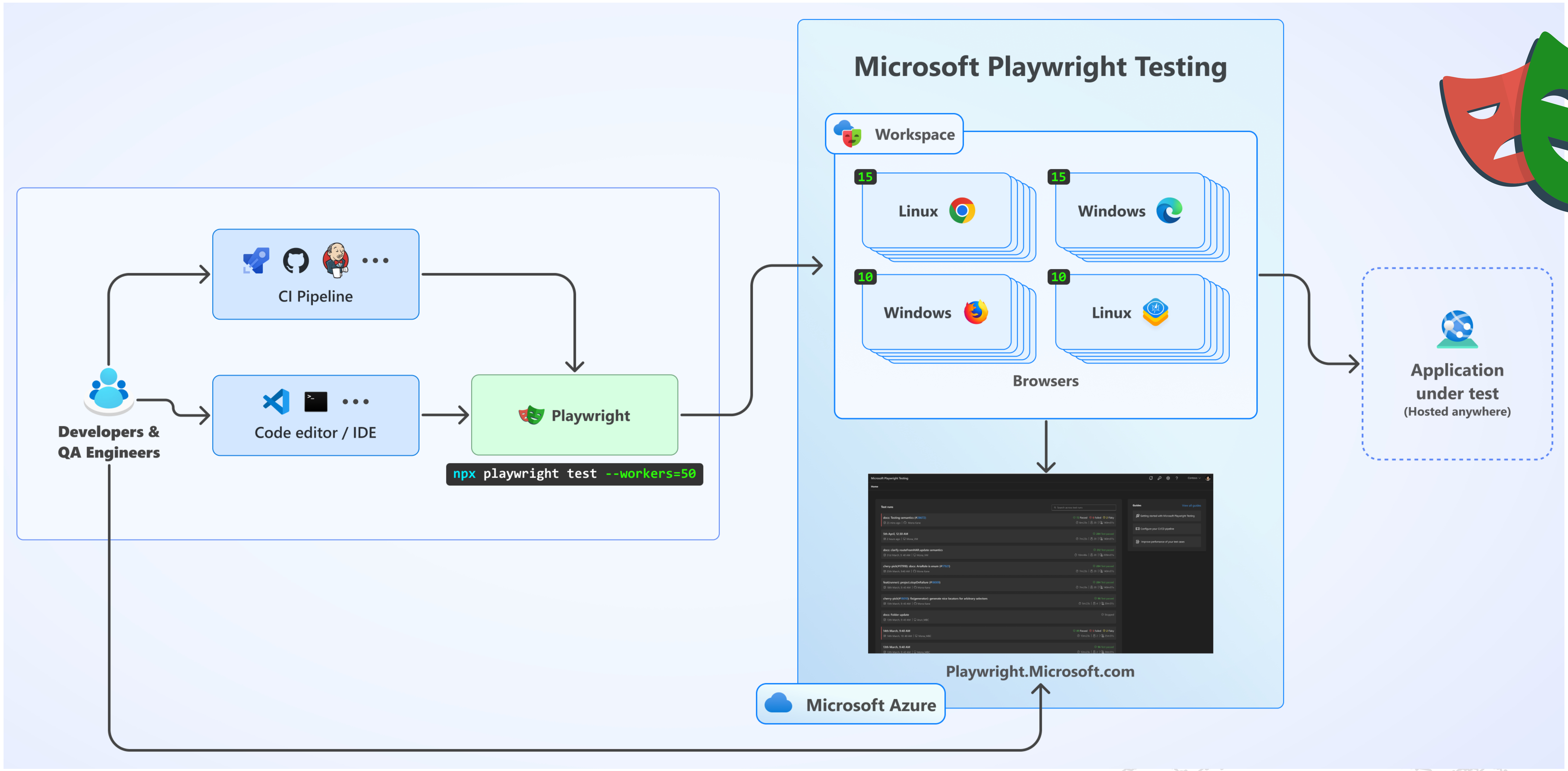
Playwright

- Open-source test framework for browser-based testing built and maintained by Microsoft
- Cross-browser: Chromium (Chrome, Edge), WebKit (Safari), Firefox
- Cross-OS: Windows, MacOS, Linux
- Cross-language: Node.js, .NET, Java, Python

Key features

- Code generation through recording
- Playwright inspector to help with target analysis
- Trace Viewer to get all execution information
- Easy headless execution in pipelines

Functional testing: how



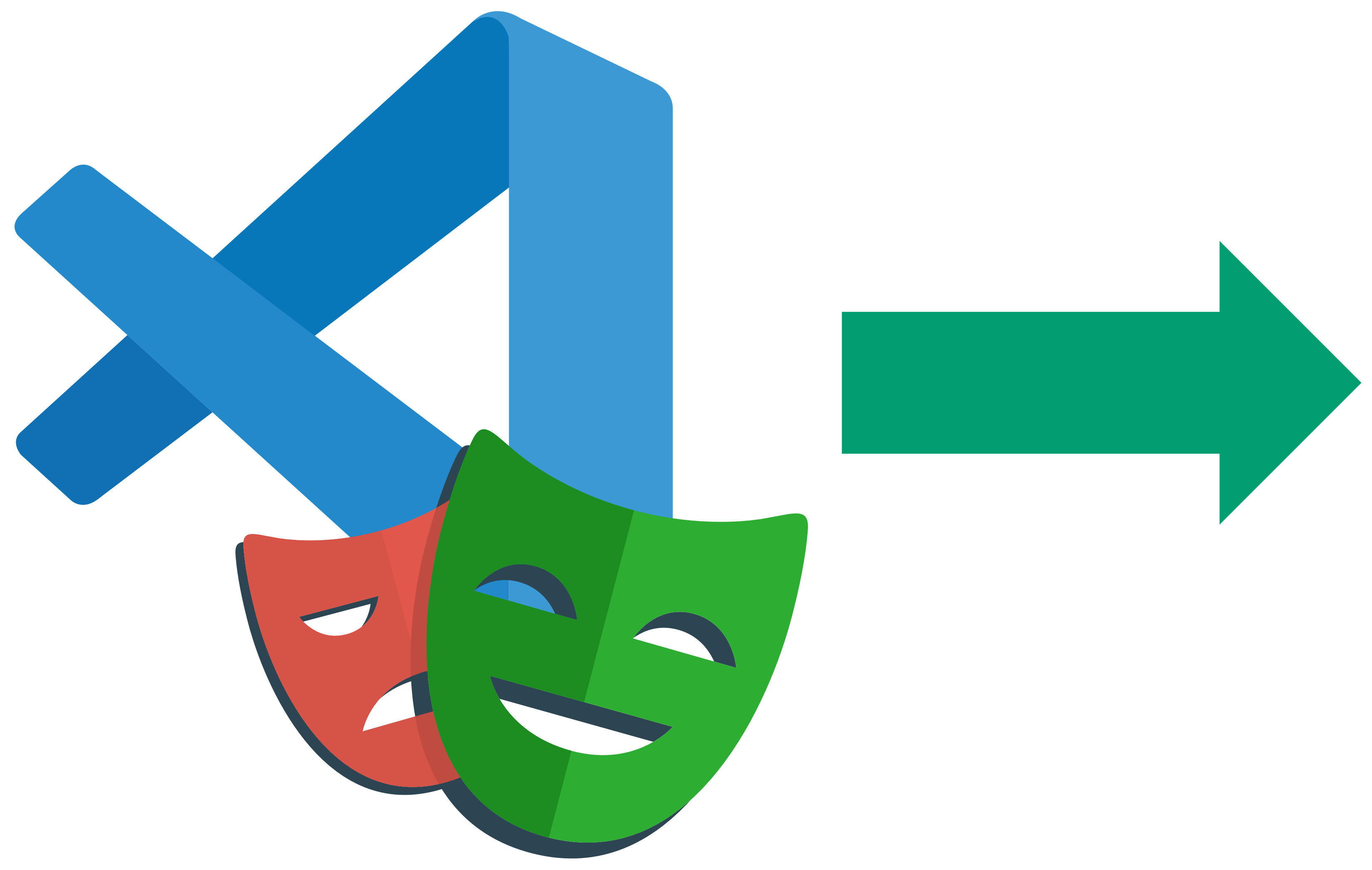
```
npx playwright test --workers=50
```

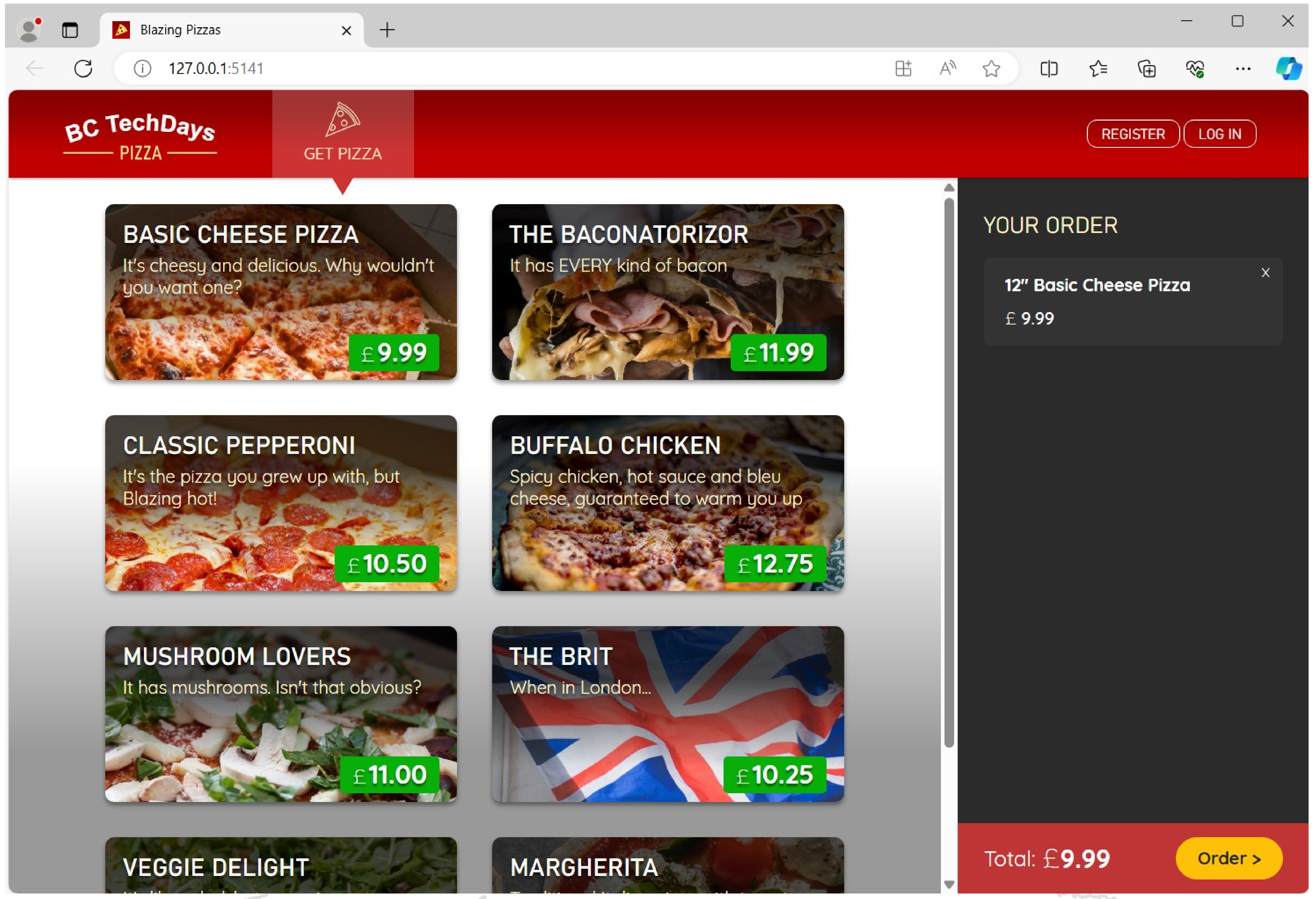
https://learn.microsoft.com/en-us/azure/playwright-testing/overview-what-is-microsoft-playwright-testing

Functional testing: demo time

Scenario

- Blazor web app: BC TechDays Pizza! Based on Blazor Workshop by Jeffrey Fritz (github.com/csharpfritz/BlazingPizzaWorkshop)
- Non-trivial dynamic website
- Do some recording, coding, playback





Functional testing: automation

Easy to set up on Azure Pipelines, Github Actions and others

Continuous Integration | Playwright

Functional testing: try at home

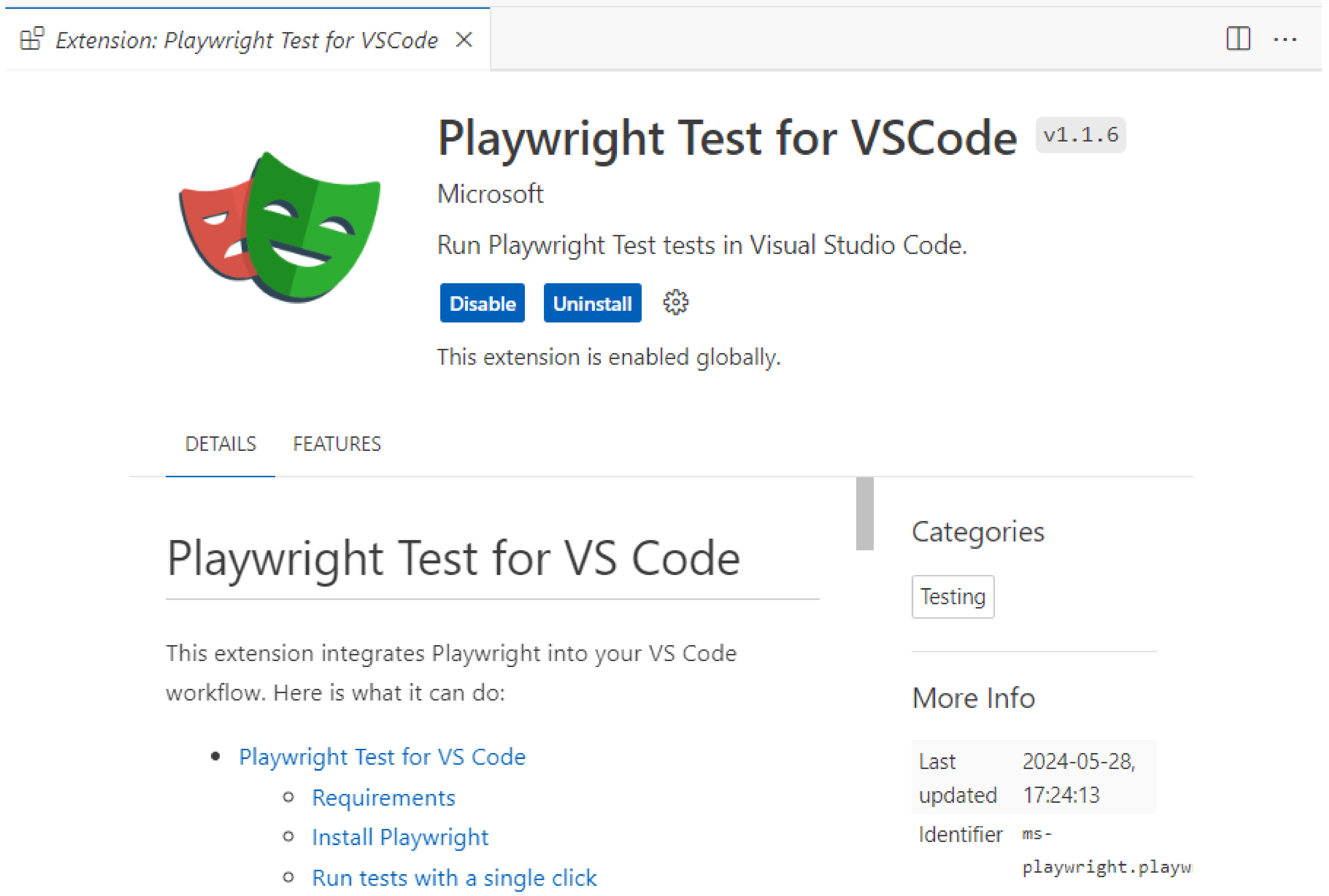
Run application in dev container: https://github.com/tfenster/BlazingPizzaWorkshop

Install Playwright through the VS Code extension

- Natively, not in dev container

Get started!

- Record, code, playback
- Push to Github → CI!


Extension: Playwright Test for VSCode

Playwright Test for VSCode v1.1.6

Microsoft

Run Playwright Test tests in Visual Studio Code.

This extension is enabled globally.

Playwright Test for VS Code

This extension integrates Playwright into your VS Code workflow. Here is what it can do:

- Playwright Test for VS Code
  - Requirements
  - Install Playwright
  - Run tests with a single click

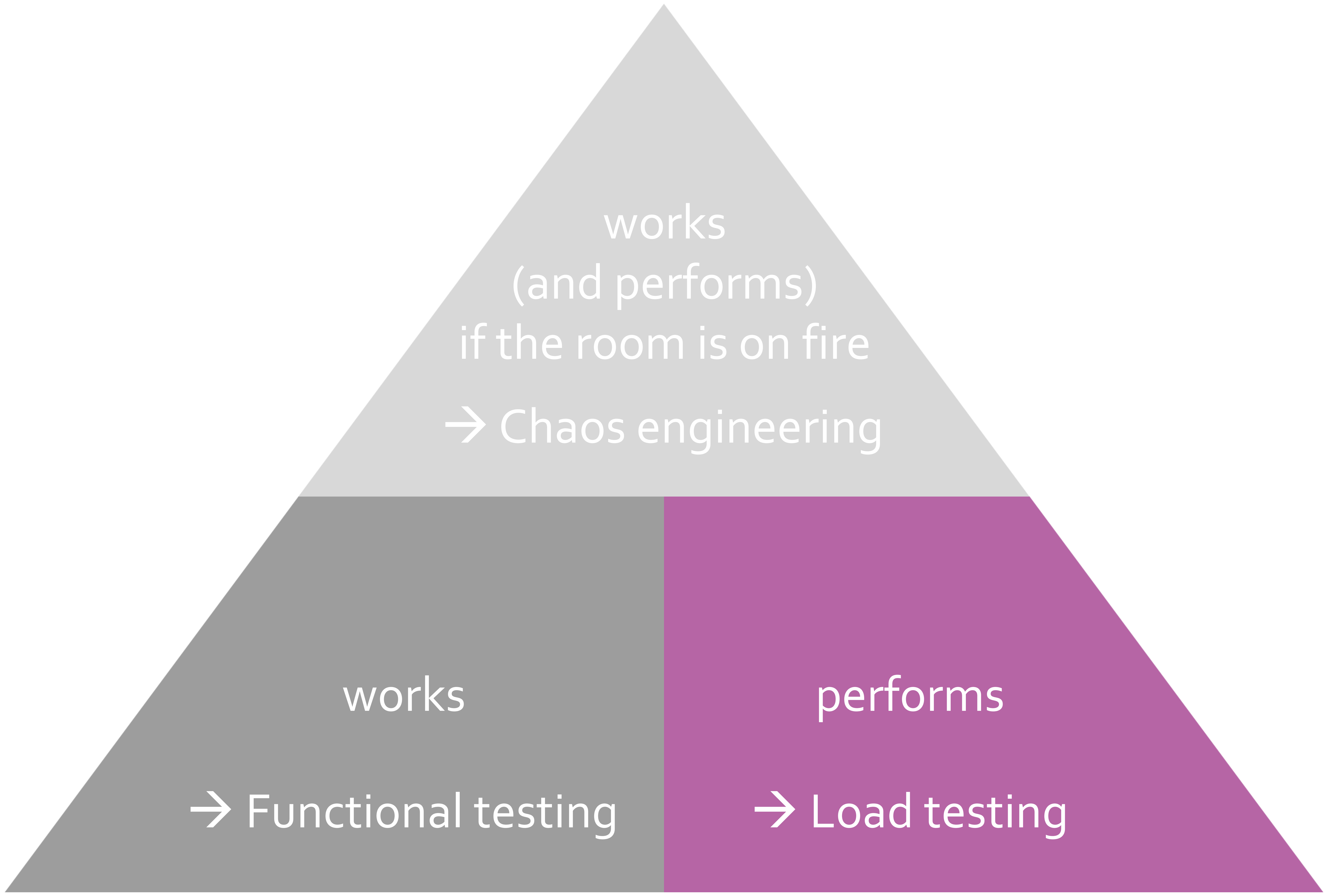
Categories

Testing

More Info

Last updated	2024-05-28, 17:24:13
Identifier	ms-playwright.playw

Quality


works
(and performs)
if the room is on fire
→ Chaos engineering
works
→ Functional testing
performs
→ Load testing

Performance testing: what

“Make sure the application can handle a load of X using Y resources with a response time of Z (and an acceptable level or errors)”

Test

- (core) processes
- expected, realistic load scenarios
- more than expected → identify bottlenecks and breaking points
- establish a base line and compare against it

Test critical (initial screen, payment, posting, ...) or highly resource intensive parts (complex calculation, data analysis, batch writing, ...)

Performance testing: how

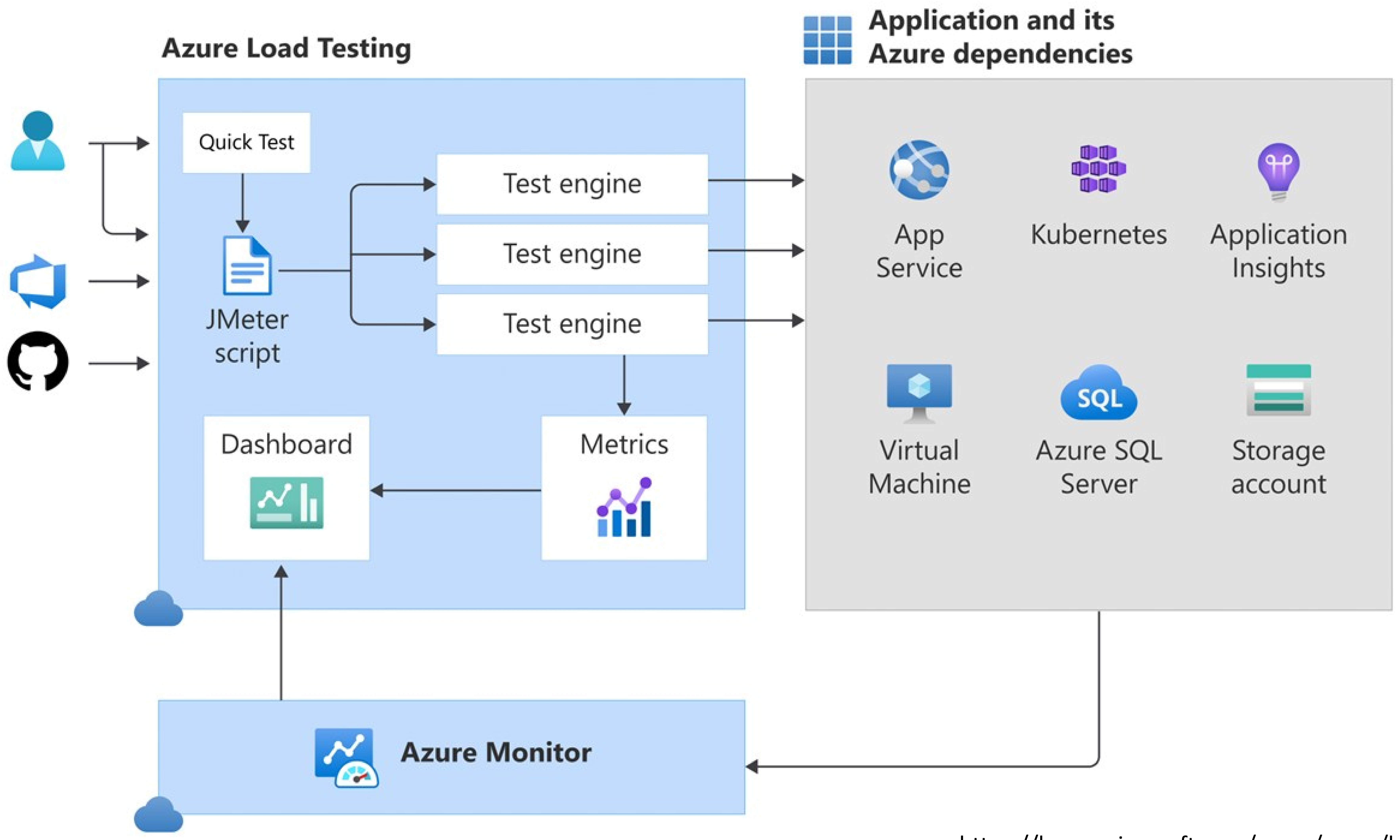
Azure Load Testing

- Cloud-based load-testing service
- Generate load on demand with very little setup
- Generate geographically distributed load to match real-world scenarios

Key features

- Quick-start features for simple to medium scenarios
- Full power of established load-testing framework JMeter if needed
- Integrated in Azure to simultaneously get metrics of system under test
- Azure CLI or IaC tools to create tests

Performance testing: how

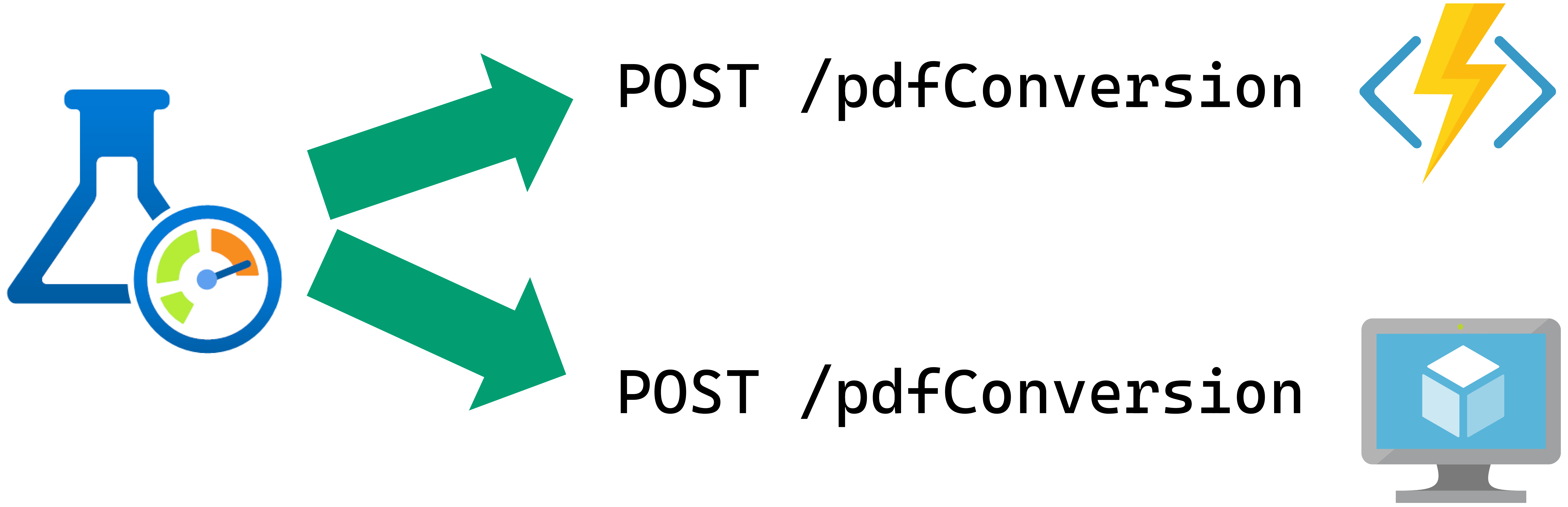


https://learn.microsoft.com/en-us/azure/load-testing/overview-what-is-azure-load-testing

Performance testing: demo time

Scenario

- Service to convert PDF to PNG (and a simple GET to begin)
- Containerized code running on Azure Function and on Azure VM
- Generate load, find bottlenecks, scale infrastructure

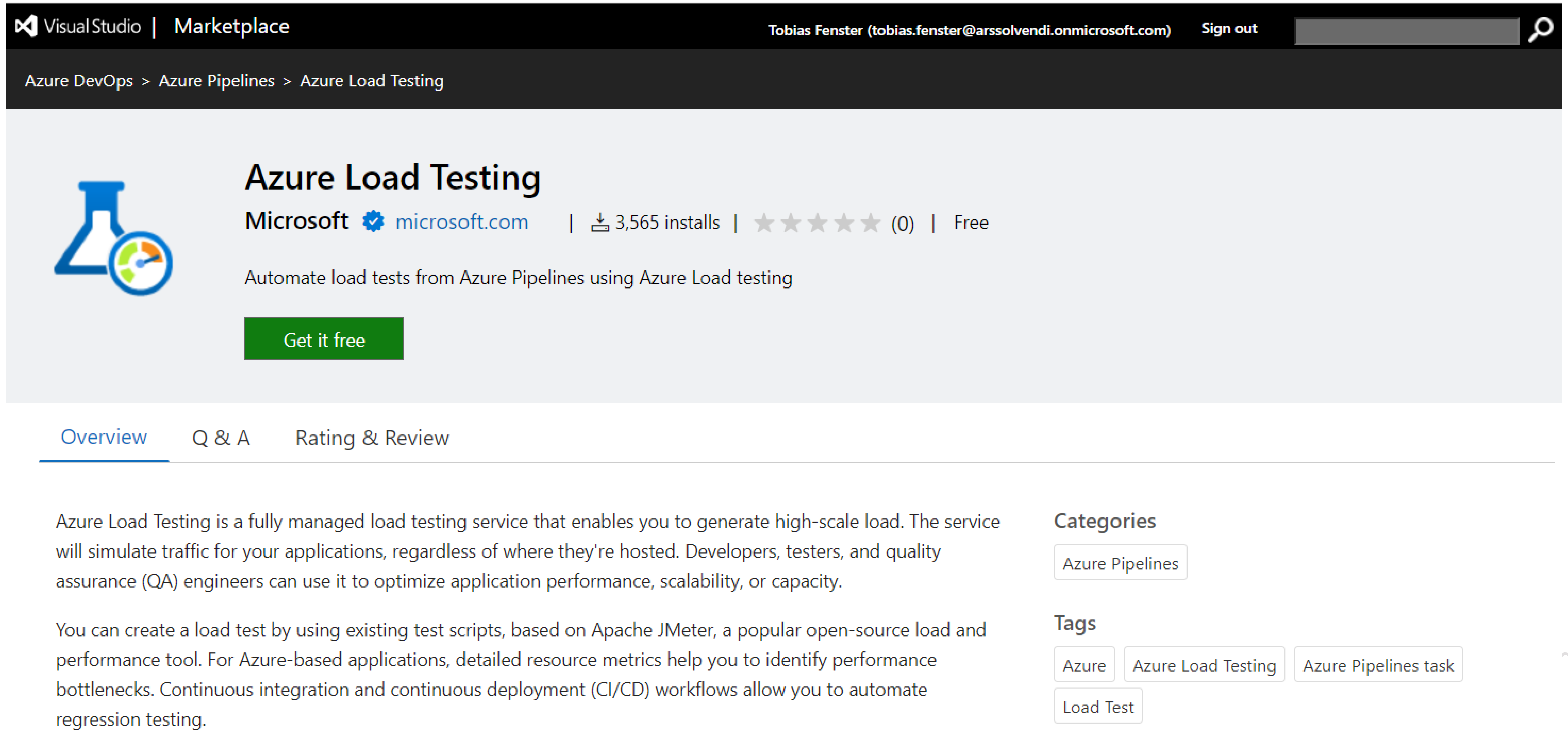


Performance testing: automation

Easy to set up on Azure Pipelines and Github Actions

[Quickstart: Automate load tests with CI/CD - Azure Load Testing | Microsoft Learn]()

[Manually configure CI/CD for load tests - Azure Load Testing | Microsoft Learn]()


Visual Studio | Marketplace
Tobias Fenster (tobias.fenster@arssolvendi.onmicrosoft.com)
Sign out
Azure DevOps > Azure Pipelines > Azure Load Testing
Get it free
Overview
Q & A
Rating & Review

Azure Load Testing

Microsoft microsoft.com | 3,565 installs | (0) | Free

Automate load tests from Azure Pipelines using Azure Load testing

Azure Load Testing is a fully managed load testing service that enables you to generate high-scale load. The service will simulate traffic for your applications, regardless of where they're hosted. Developers, testers, and quality assurance (QA) engineers can use it to optimize application performance, scalability, or capacity.

You can create a load test by using existing test scripts, based on Apache JMeter, a popular open-source load and performance tool. For Azure-based applications, detailed resource metrics help you to identify performance bottlenecks. Continuous integration and continuous deployment (CI/CD) workflows allow you to automate regression testing.

Categories

Azure Pipelines

Tags

Azure | Azure Load Testing | Azure Pipelines task | Load Test

Performance testing: try at home

Run conversion service in Azure Function or VM:
https://github.com/tfenster/presentation-src/tree/bctechdays-24-loadtest

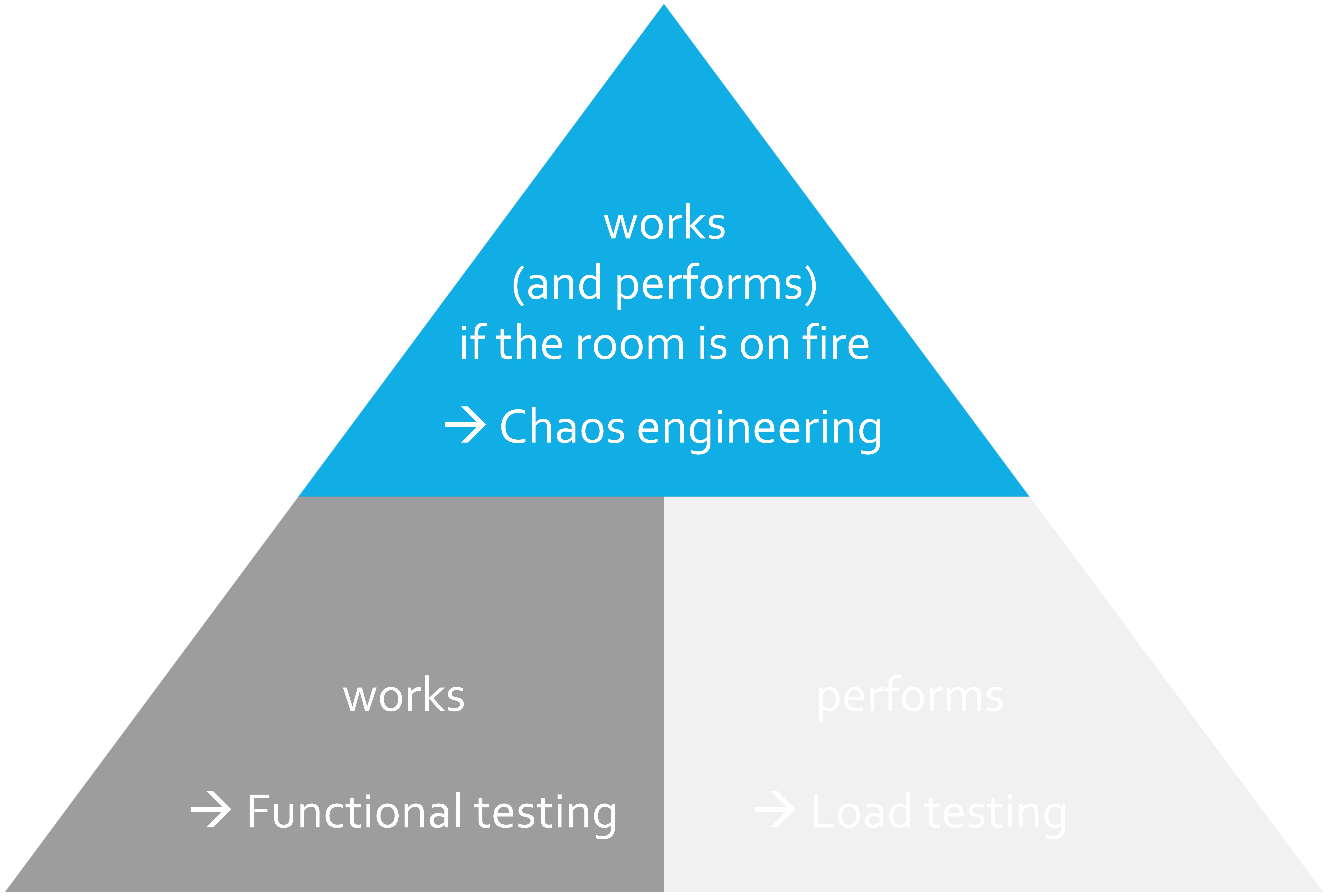
- createAzInfra*.sh

Create load tests based on resources in same repo

- load-test.*
- createLoadTests.sh

Install JMeter (Java!) to adjust tests

Quality


works
(and performs)
if the room is on fire
→ Chaos engineering
works
→ Functional testing
performs
→ Load testing

Chaos engineering: what

"Make sure the application works and performs even if something drastic goes wrong"

Validate
- (core) processes and features
- resiliency against things that shouldn't happen
- acceptable fallback behavior

Tricky: What to test?
- the things you already planned resiliency for
- the things you haven't planned for…

Chaos engineering : how

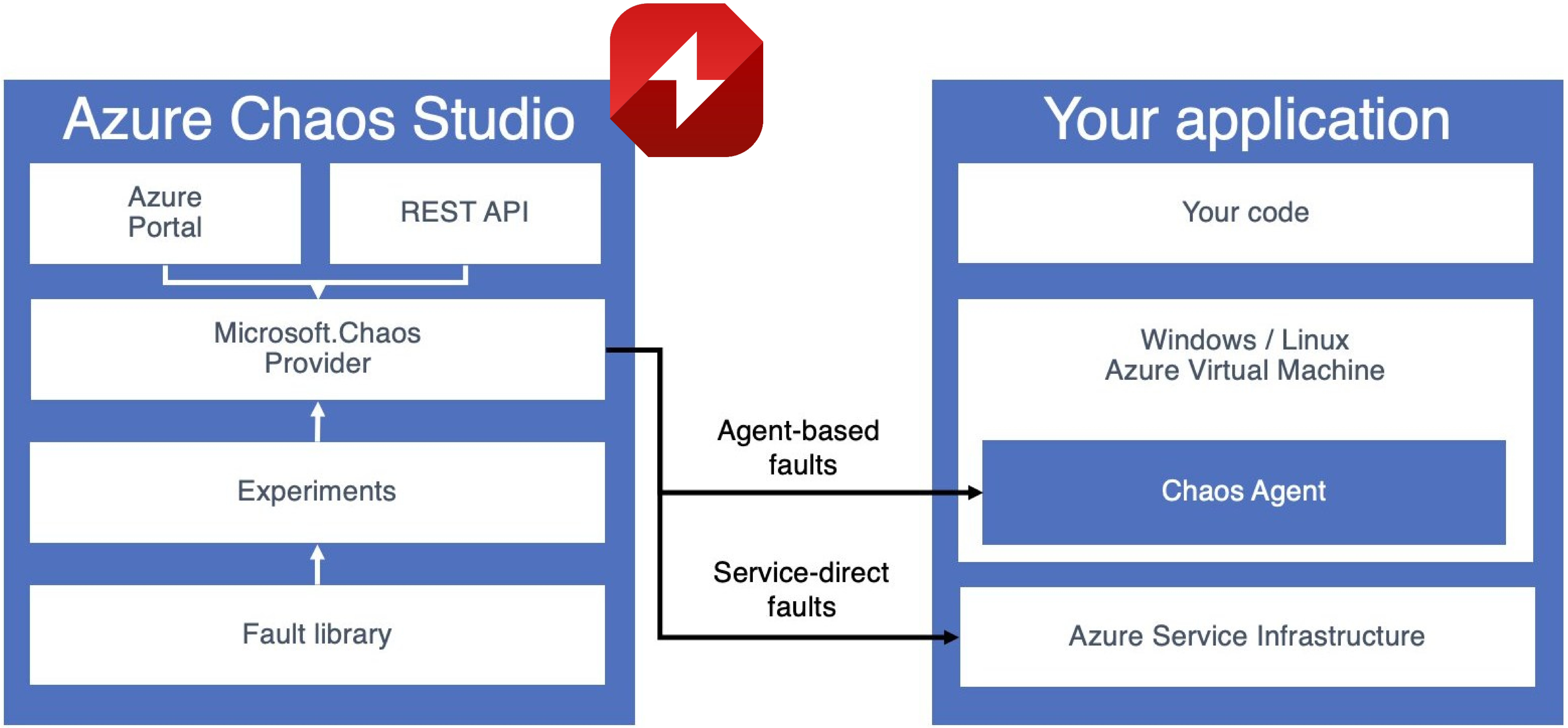
Azure Chaos Studio

- Experimentation platform to introduce faults and stresses
- Managed service by Microsoft
- Allows you to design and run experiments

Key features

- Running an experiment (“simulation”) means actually injecting the problems → Understand what you do!
- Either service-direct or agent-based faults
- Multiple security measures in place to make sure you don’t accidentally break anything
- Can be used in both shift left and shift right approaches

Chaos engineering : how

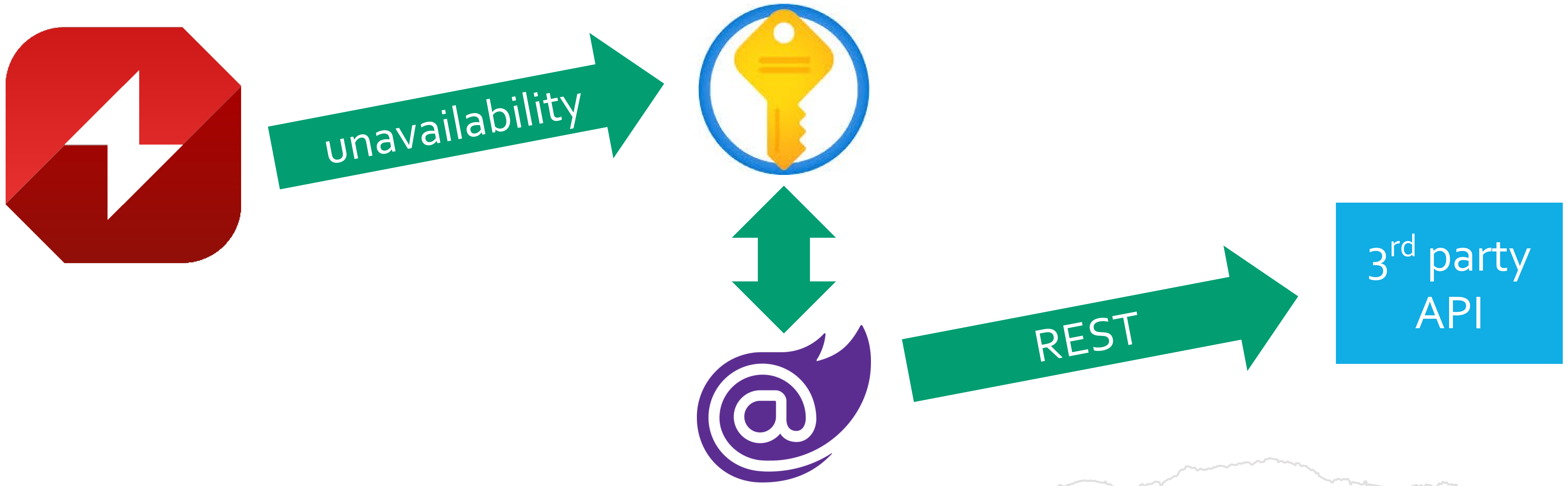


https://thomasvanlaere.com/posts/2021/12/azure-chaos-studio-and-powershell/

Chaos engineering: demo time

Scenario 1

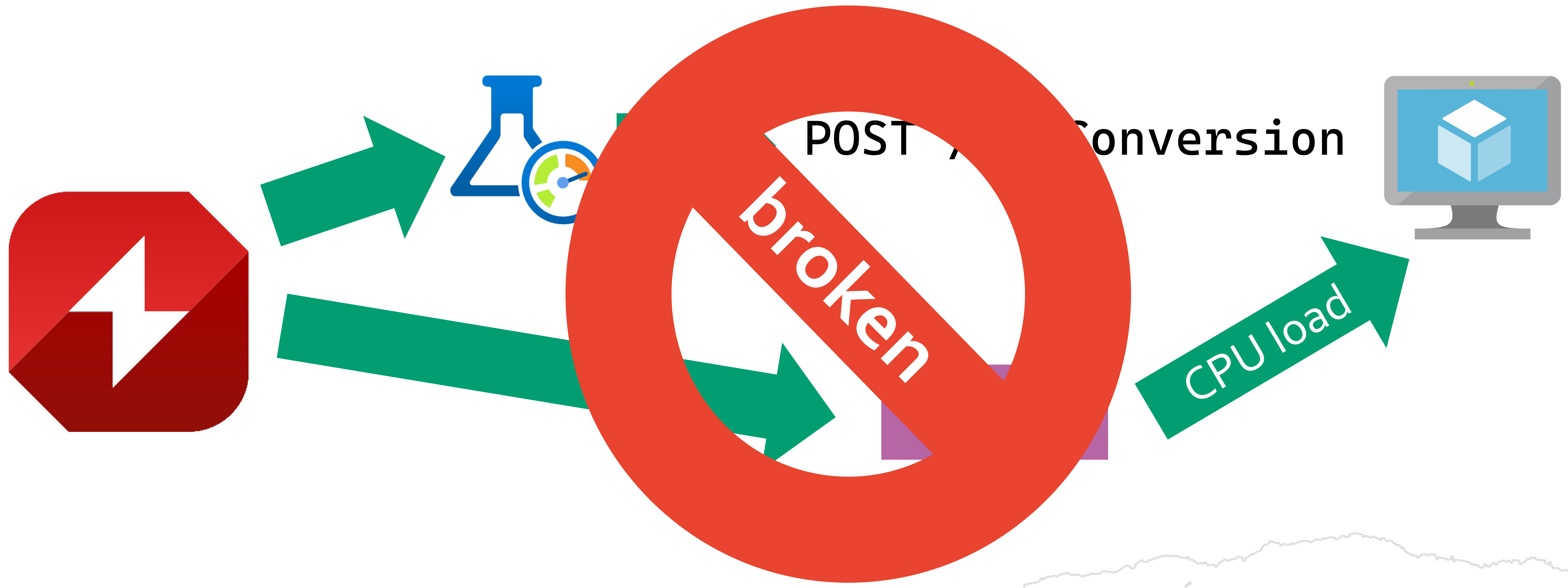
- Web application uses Azure Key Vault to access a service key that is needed to call an external API
- Azure Chaos Studio is used to introduce unavailability of the Key Vault
- Check application behavior



Chaos engineering: demo time

Scenario 2

- Same as in load testing: Web API hosted on an Azure VM to convert PDFs to PNGs
- Azure Chaos Studio is used to introduce high CPU load on the Azure VM
- Check application behavior with load test

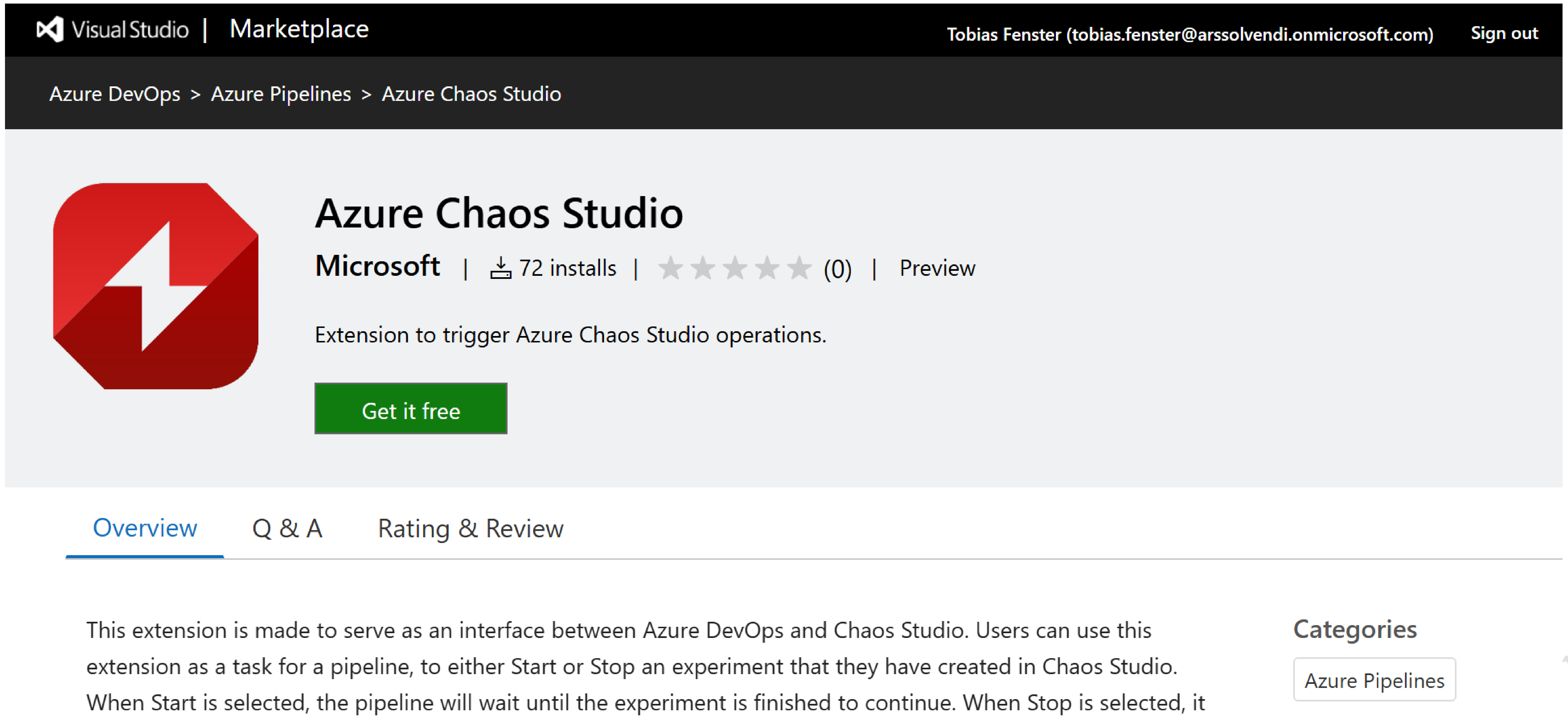


BC/NAV TECH DAYS 2024 mibuso.com

Chaos engineering: automation / plan B for scenario 2

Possible to set up using Azure CLI or IaC tools like Terraform or Bicep

Automated performance test included in a chaos engineering experiment can be useful


Visual Studio | Marketplace
Tobias Fenster (tobias.fenster@arssolvendi.onmicrosoft.com) Sign out
Azure DevOps > Azure Pipelines > Azure Chaos Studio

Azure Chaos Studio

Microsoft | 72 installs | ★★★★★ (0) | Preview

Extension to trigger Azure Chaos Studio operations.

Get it free
Overview Q & A Rating & Review

This extension is made to serve as an interface between Azure DevOps and Chaos Studio. Users can use this extension as a task for a pipeline, to either Start or Stop an experiment that they have created in Chaos Studio. When Start is selected, the pipeline will wait until the experiment is finished to continue. When Stop is selected, it

Chaos engineering: try at home

Scenario 1:

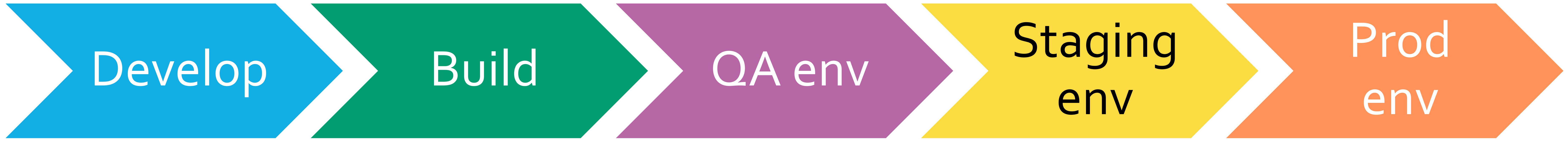
- Run web application using KV in dev container: https://github.com/tfenster/presentation-src/tree/bctechdays-24-kv
- Use scripts (createCert.sh and createAzInfra.sh) to create required resources
- Inject KV unavailability through Chaos Studio
- Observe application

Scenario 2:

- Create Azure infrastructure using same scripts as for load test: https://github.com/tfenster/presentation-src/tree/bctechdays-24-loadtest
- Deploy containerized Web API to VM
- Inject CPU stress to VM and run load test in parallel (or use pipeline as fallback)
- Observe application

When and where to test

Shift left / shift right?



Functional testing: Shift left as far as possible

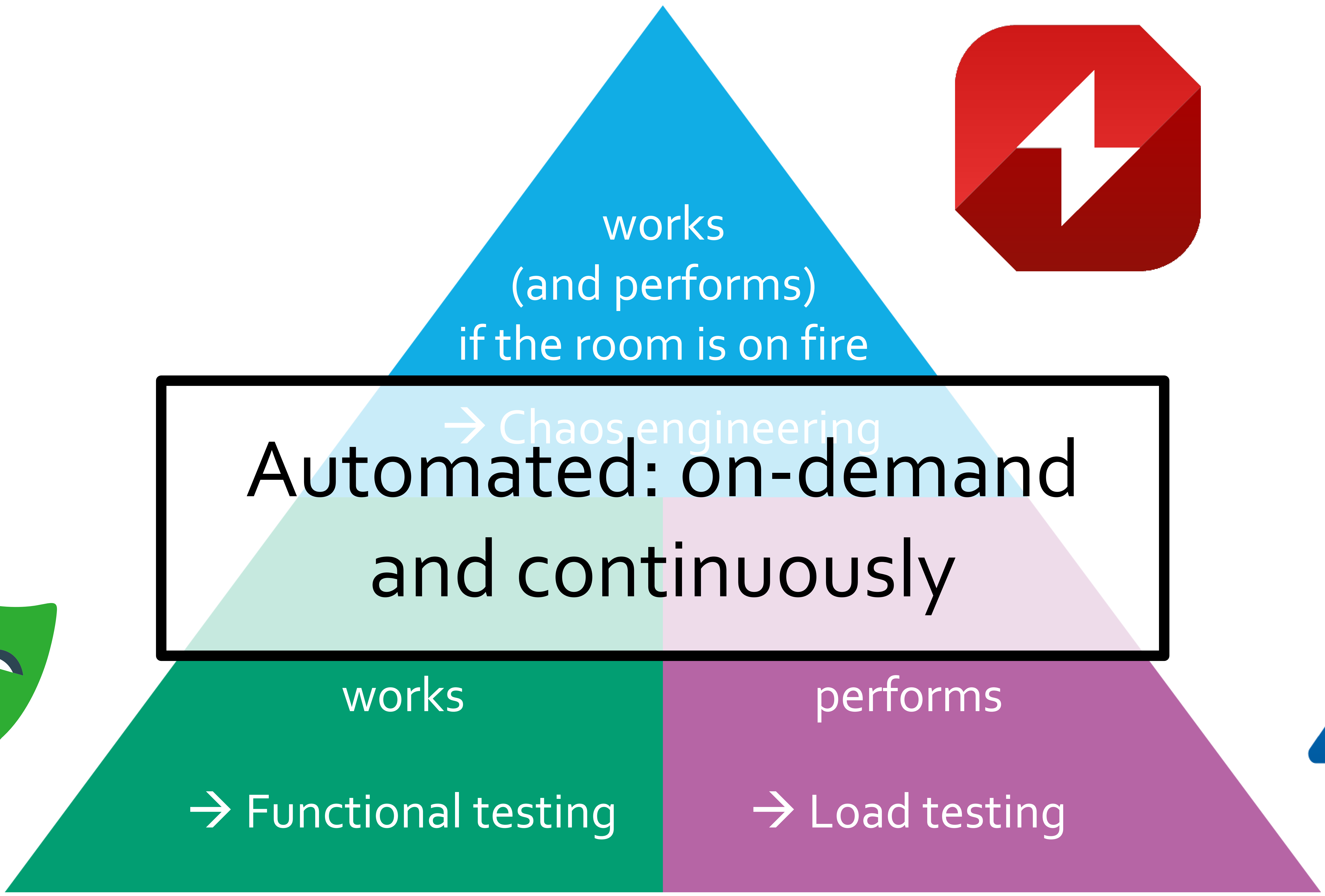
Performance testing: Shift left to catch relative problems early, shift right to validate absolute numbers

- Requires QA / Staging scale as close as possible to Prod

Chaos engineering: Shift left to catch problems early. Once confident, shift right to validate

- Requires QA / Staging on architecture as close as possible to Prod

Recap



Automated: on-demand and continuously

Q&A

Any Questions?

BC/NAV TECH DAYS 2024
mibuso.com

Thank You