Grafana Labs

Evaluating Performance

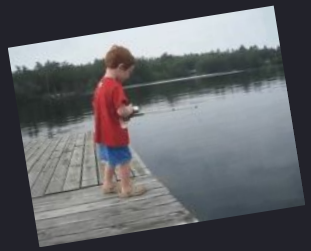
At a Massive Scale

Γειά σας!

Giannis Papadakis
Director QA, GWI

About me

- Based in Athens, Greece
- I love working in all things that touch Product and Engineering
- Grafana Champion
- I love hiking and fishing



Contact

- Email: gpapadakis@gwi.com

Overview

1. Introduction
2. Foundations
3. K6 in CI
4. Advanced
5. Remarks / Q&A

What is load testing?

"Load testing is the process of putting demand on a system and measuring its response."

Load Testing

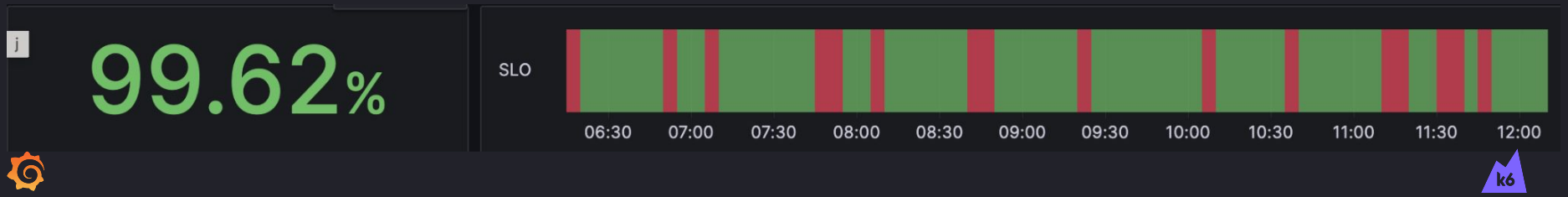
✅ ❌

Load Testing

Accept Errors: 9s of Reliability

Accept Outliers: percentiles 90/95/99/..

✅



Latency 🏎️

Availability 🟢

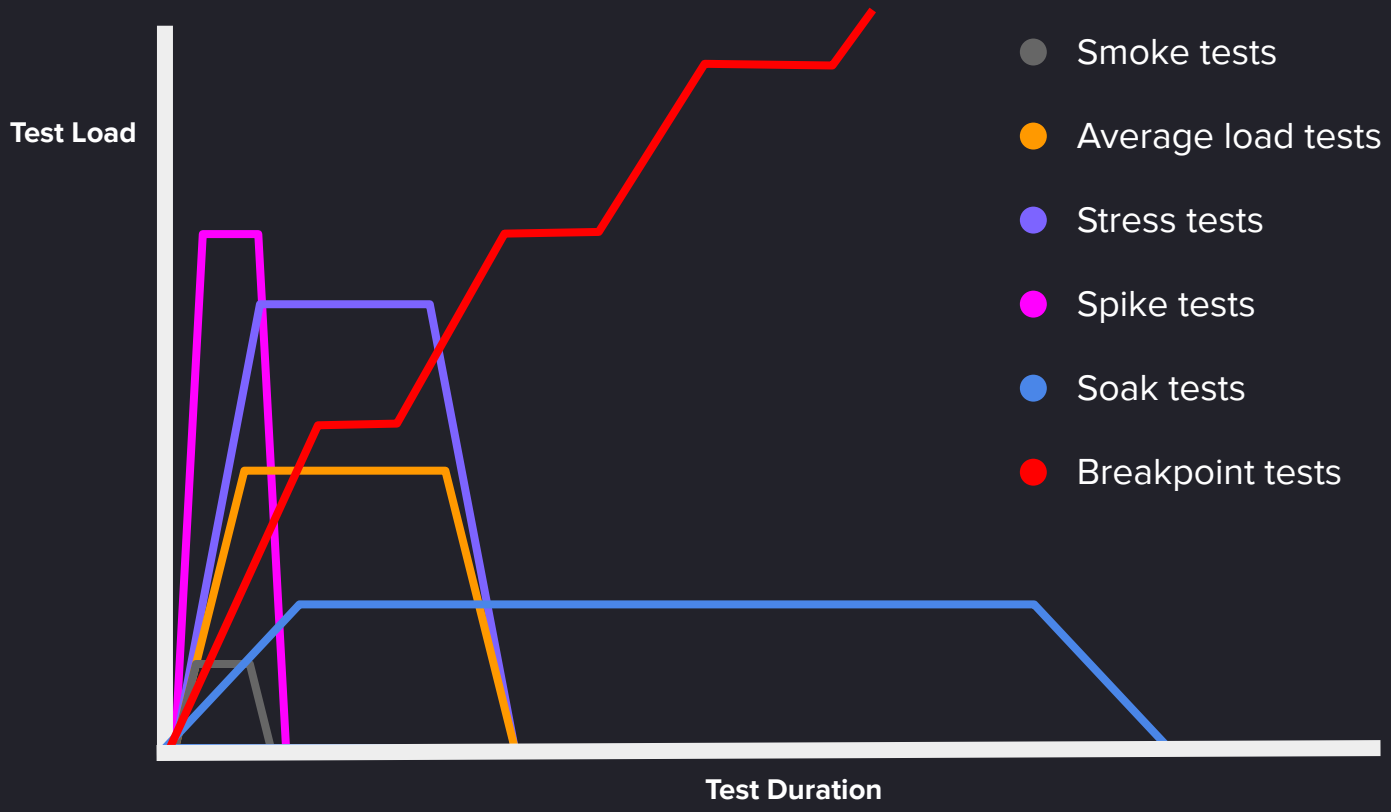
Reliability ✔️

Resilience 💥

Scalability/Elasticity 📈

“Load testing is the process of putting demand on a system and ***measuring its response****”*

Common types of load tests



Myths about load testing

- Performance testing = load testing.
- Load testing is only for large companies.
- Load testing is expensive.
- Load testing should only be done in production.
- You don't need load testing if you have observability.

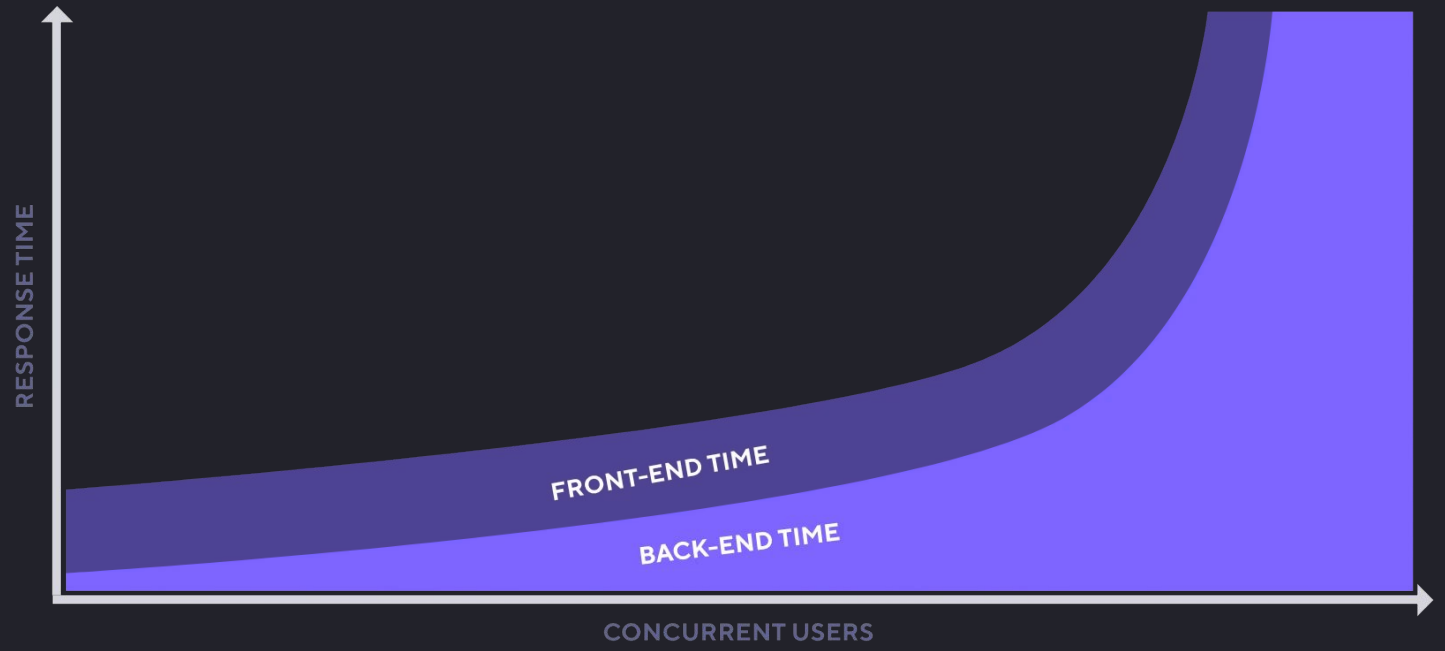
Why load testing today?

UX

SLOs

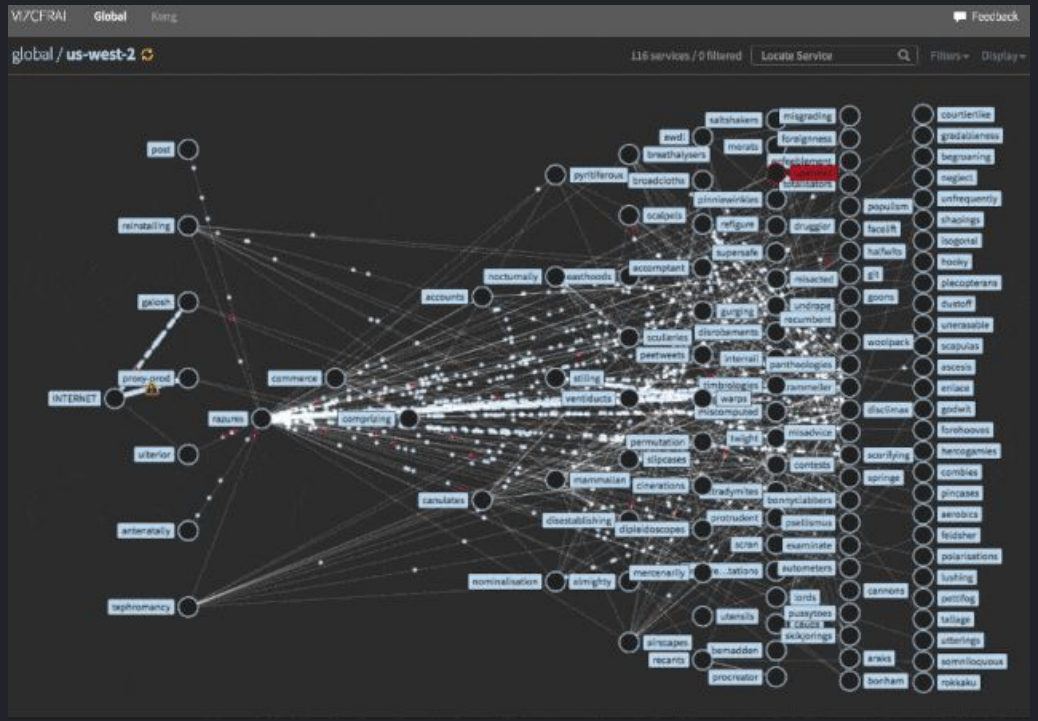
Why do load testing today?

UX



Why load testing today?

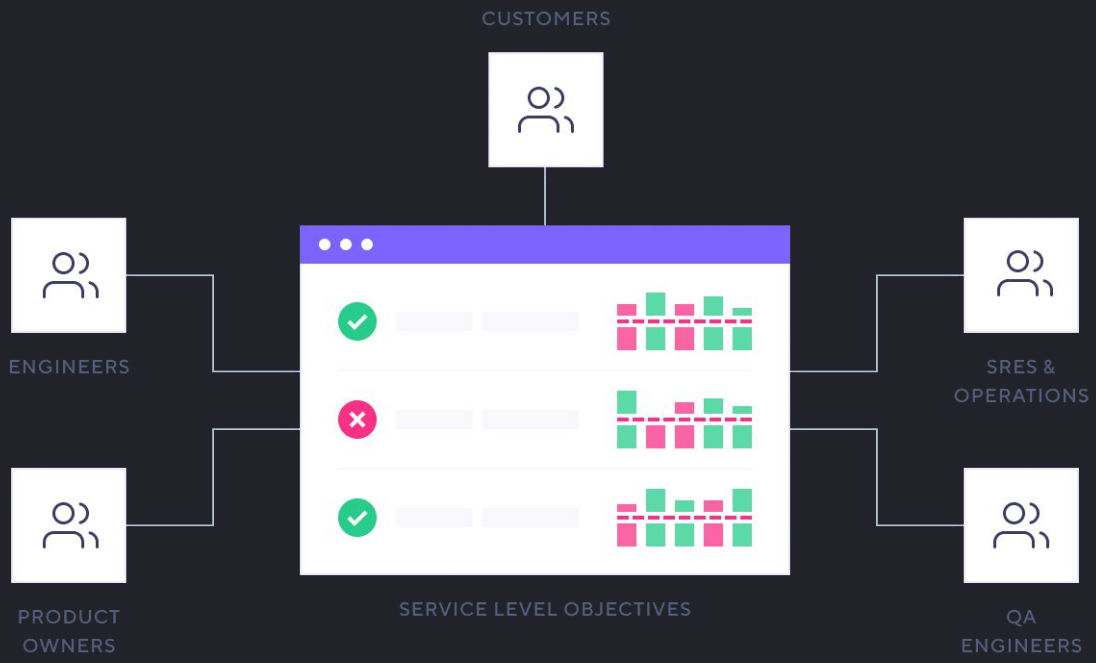
Distributed systems



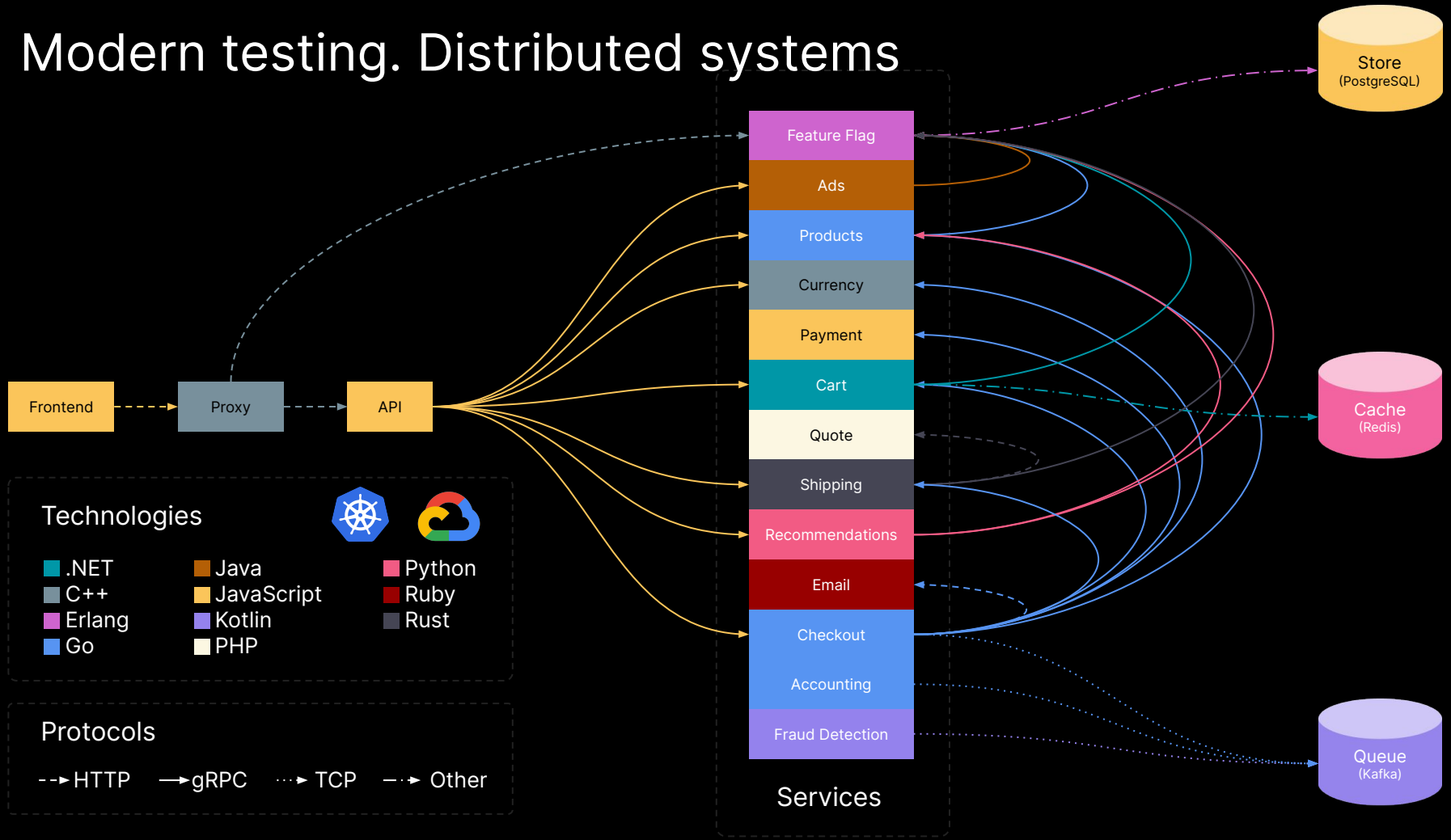
Why load testing today?

Proactively test

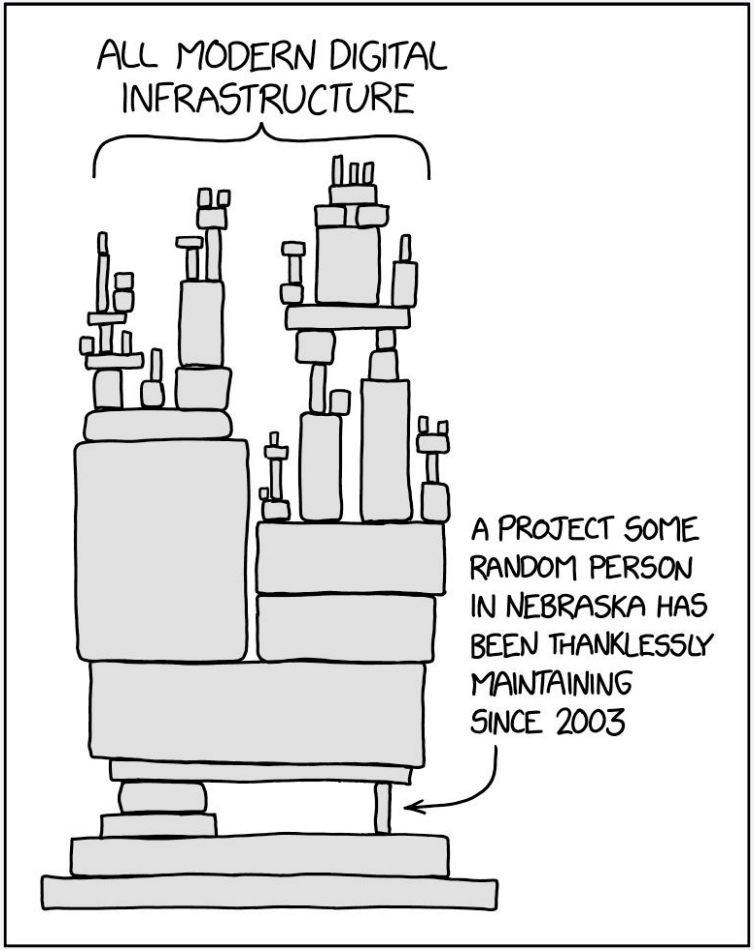
SLOs



Modern testing. Distributed systems



Complexity



Mandatory xkcd #2347




Things will break

Everywhere

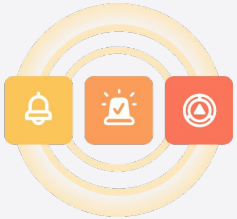

Joining the dark side

Embracing failures as something normal

Be prepared. Please!

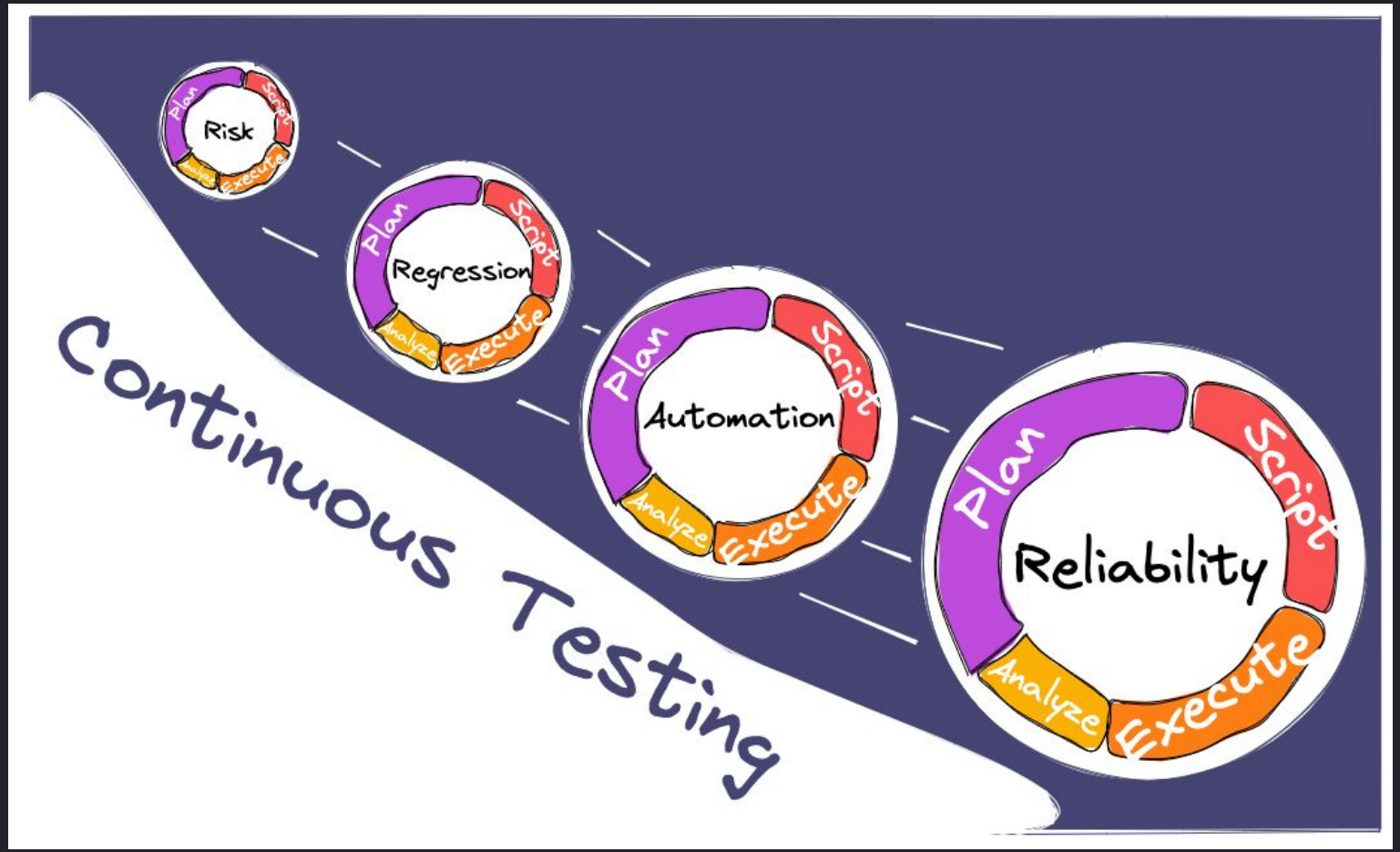
Observe

Act



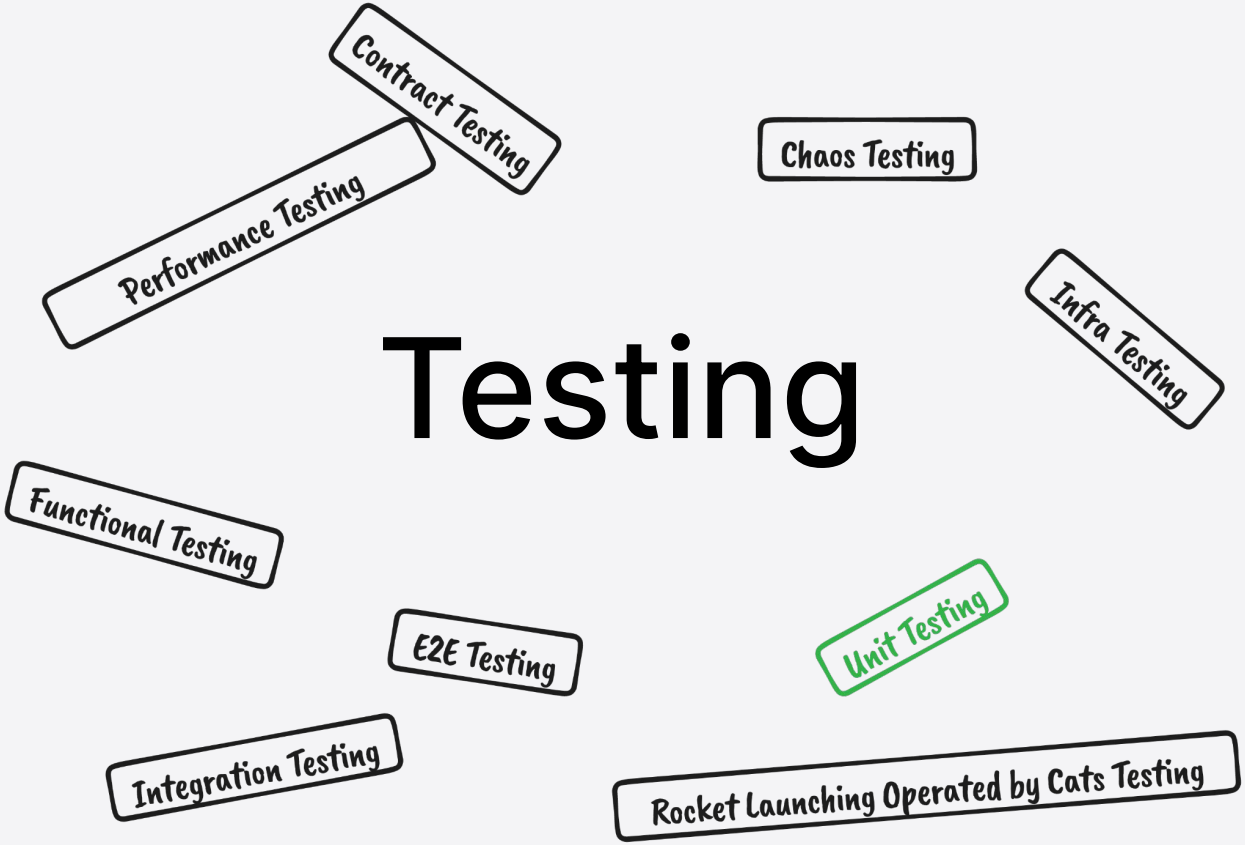
Fail early. Fail often.

✨Testing ✨


Risk
Plan
Script
Execute
Regression
Plan
Script
Analyze
Execute
Automation
Plan
Script
Analyze
Execute
Reliability
Plan
Script
Analyze
Execute
Continuous Testing

But not any kind of testing

- It needs to:
  - Cover the right places.
  - Be easy to maintain and up-to-date.
    - Tests must be own by folks who develop and maintain the software.
  - Be runnable at any step of the software development cycle.
    - For example, to:
      - Aid devs while creating a new feature.
      - Fail CI on pull requests if something isn't OK.
      - Stop a release on the canary system if something pops up.


Contract Testing
Performance Testing
Chaos Testing
Infra Testing
Testing
Functional Testing
E2E Testing
Unit Testing
Integration Testing
Rocket Launching Operated by Cats Testing

Problem: Bad developer experience

Grafana k6

The best developer experience for performance testing

Why k6?

1. Open source
2. Programmable
3. Performant
4. Extensible
5. Docs, Docs, Docs...

Grafana Labs

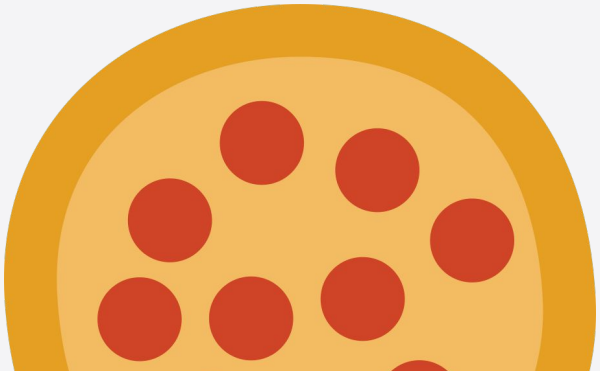
Our workshop steps

Any questions?

https://github.com/Athens-SDET-Meetup/performance_workshop

QuickPizza

https://github.com/grafana/quickpizza



Getting started (yes, in one slide)

```
import http from 'k6/http';
import { check, sleep } from 'k6';

export const options = {
  vus: 10,
  duration: '30s',
};

export default function () {
  const res = http.get('https://httpbin.test.k6.io/');
  check(res, { 'status was 200': (r) => r.status == 200 });
  sleep(1);
}
```

```
k6 run script.js
```

Simple testing is better than no testing

Scenarios

Simulating user flows and workload modeling, adding think time

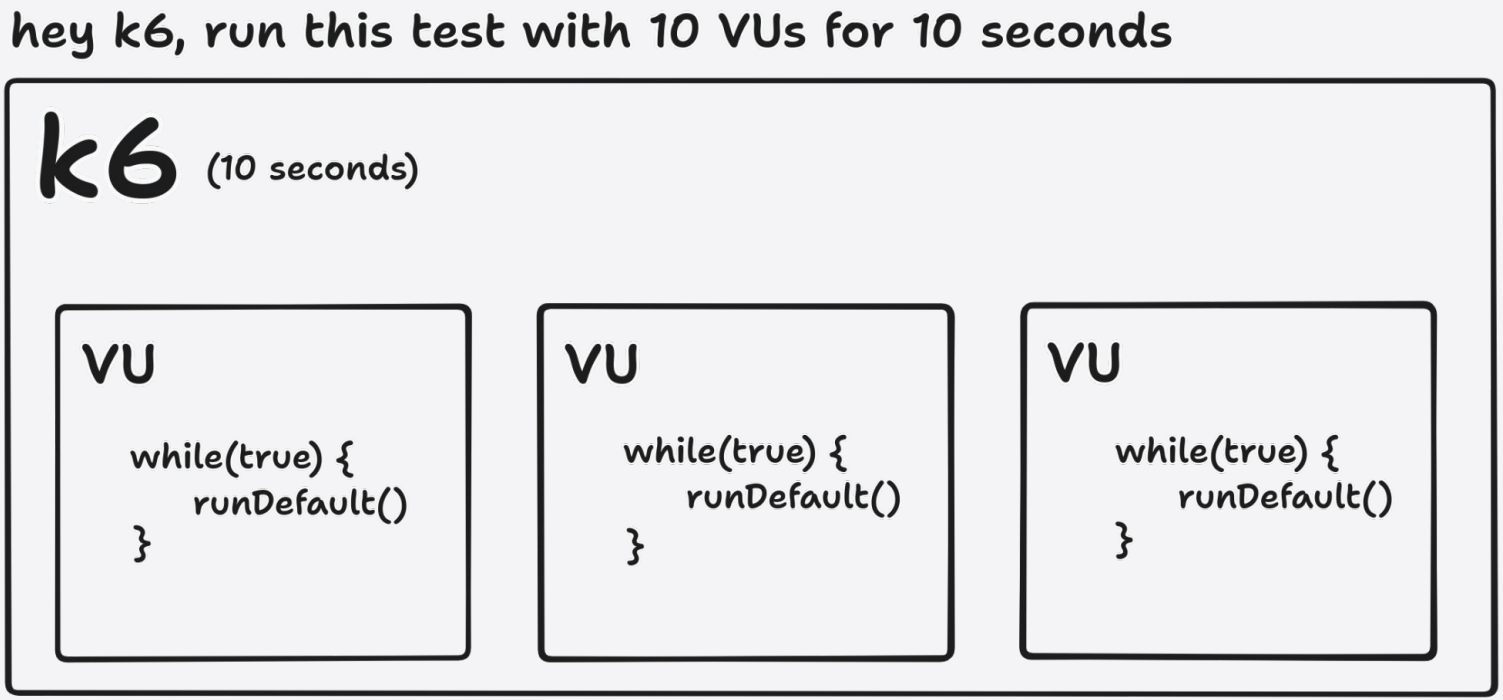
Thresholds

Setting SLOs for the system under load

Test data

Using dynamically generated or random values

VUs and iterations



Scenarios: Test parameters

VUs, iterations

```
export const options = {
    vus: 100,
    iterations: 200,
}
```

duration

```
export const options = {
    vus: 100,
    duration: '30m',
}
```

think time

```
export default function() {
  http.get('https://myapi.com/products/');
  sleep(5);
}
```

Thresholds ✅

Testing should be goal oriented

Checks

Thresholds

Like Unit testing, for Performance

```
import { check } from 'k6';
import http from 'k6/http';

export default function () {
  const res = http.get('http://test.k6.io/');
  check(res, {
    'is status 200': (r) => r.status === 200,
  });
}
```

```
export const options = {
  thresholds: {
    http_req_failed: ['rate<0.01'],
    http_req_duration: ['p(95)<400'],
    'http_req_duration{type:API}': ['p(95)<200'],
   my_custom_metric: ['p(99)<200']
  },
};
```

Results output: Real time

```
k6 run --out experimental-prometheus-rw script.js
```

https://k6.io/docs/results-output/real-time/

Loading and using data

```
import { SharedArray } from 'k6/data';

const data = new SharedArray('some data name', function () {
  return JSON.parse(open('./data.json')).users;
});

export default function () {
  const user = data[0];
  console.log(data[0].username);
}
```

https://k6.io/docs/examples/data-parameterization/

Write once, run as you need

Scenarios

- By number of iterations:
  - Shared-iterations
  - Per-vu-iterations
- By number of VUs:
  - Constant-VUs
  - Ramping-vus
- By iteration rate.
  - Constant-arrival-rate
  - Ramping-arrival-rate

```
import http from "k6/http";

export const options = {
  scenarios: {
    shared_iter_scenario: {
      executor: "shared-iterations",
      vus: 10,
      iterations: 100,
      startTime: "0s",
    },
    per_vu_scenario: {
      executor: "per-vu-iterations",
      vus: 10,
      iterations: 10,
      startTime: "10s",
    },
  },
};

export default function () {
  http.get("https://test.k6.io/");
}
```

https://k6.io/docs/using-k6/scenarios/

But HTTP is not enough!

We want to test all the things. Full-stack.

- Native support for:
  - Protocol:
    - HTTP/1.1
    - HTTP/2
    - WebSockets
    - gRPC (unary)
    - Redis - Experimental
  - Browser:
    - Chromium - Experimental

```
import ws from 'k6/ws';
import { check } from 'k6';

export default function () {
  const url = 'ws://echo.websocket.org';
  const params = { tags: { my_tag: 'hello' } };

  const res = ws.connect(url, params, function (socket) {
    socket.on('open', () => console.log('connected'));
    socket.on('message', (data) => console.log('Message received: ', data));
    socket.on('close', () => console.log('disconnected'));
  });

  check(res, { 'status is 101': (r) => r && r.status === 101 });
}
```

https://k6.io/docs/using-k6/protocols/

Browser

```
import { chromium } from 'k6/experimental/browser';

export default async function () {
  const browser = chromium.launch({ headless: false });
  const page = browser.newPage();

  try {
    await page.goto('https://test.k6.io/my_messages.php', { waitUntil: 'networkidle' });

    // Enter login credentials
    page.locator('input[name="login"]').type('admin');
    page.locator('input[name="password"]').type('123');

    page.screenshot({ path: 'screenshot.png' });
  } finally {
    page.close();
    browser.close();
  }
}
```

https://k6.io/docs/using-k6-browser/overview/

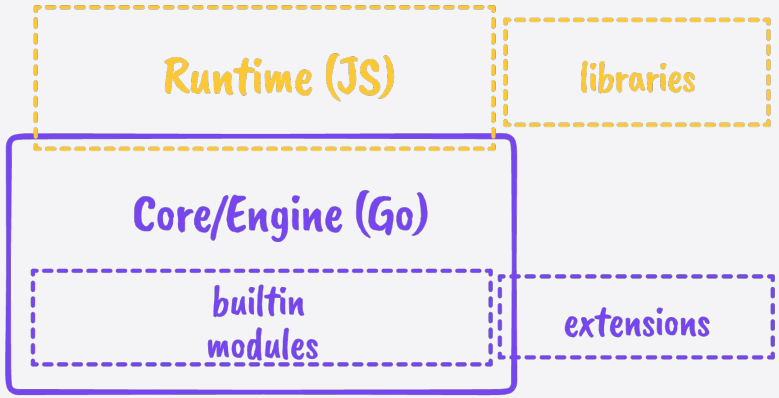
Composability 

What about X, Y and Z?

Doing more things with k6

Libraries and Extensions

- Libraries: Write JS, call from JS.
- Extensions: Write Go, call from JS.
  - Thriving ecosystem:
    - Kafka, SQL, SQS, MQTT, Kubernetes
    - More: https://github.com/topics/xk6



https://k6.io/docs/javascript-api/jslib/

https://k6.io/docs/extensions/

Testing
is a team sport

Extension: SQL

```
import sql from 'k6/x/sql';

const db = sql.open("sqlite3", "./test.db");

export function setup() {
  db.exec(`CREATE TABLE IF NOT EXISTS keyvalues (
            id integer PRIMARY KEY AUTOINCREMENT,
            key varchar NOT NULL,
            value varchar);`);
}

export function teardown() {
  db.close();
}

export default function () {
  db.exec("INSERT INTO keyvalues (key, value) VALUES('plugin-name', 'k6-plugin-sql');");

  let results = sql.query(db, "SELECT * FROM keyvalues;");
  for (const row of results) {
    console.log(`key: ${row.key}, value: ${row.value}`);
  }
}
```

https://github.com/grafana/xk6-sql

Extension: Kubernetes

```
import { Kubernetes } from 'k6/x/kubernetes';

const podSpec = {
    apiVersion: "v1",
    kind:       "Pod",
    metadata: {
        name:      "busybox",
        namespace: "testns"
    },
    spec: {
        containers: [
            {
                name:    "busybox",
                image:   "busybox",
                command: ["sh", "-c", "sleep 30"]
            }
        ]
    }
}
```

```
export default function () {
  const kubernetes = new Kubernetes();

  kubernetes.create(pod)

  const pods = kubernetes.list("Pod", "testns");

  console.log(`${pods.length} Pods found:`);
  pods.map(function(pod) {
    console.log(`  ${pod.metadata.name}`)
  });
}
```

https://github.com/grafana/xk6-kubernetes

Extension: Prometheus Client Remote Write

```
import { check, sleep } from 'k6';
import remote from 'k6/x/remotewrite';

const client = new remote.Client({
    url: "<your-remote-write-url>"
});

export default function () {
    let res = client.store([{
        "labels": [
            { "name": "__name__", "value": `my_cool_metric_${__VU}` },
            { "name": "service", "value": "bar" }
        ],
        "samples": [
            { "value": Math.random() * 100, }
        ]
    }]);
    check(res, {
        'is status 200': (r) => r.status === 200,
    });
    sleep(1)
}
```

https://github.com/grafana/xk6-client-prometheus-remote

Creating an extension is pretty easy

https://k6.io/docs/extensions/get-started/create/javascript-extensions/

Grafana k6

The best developer experience for **reliability** testing

Remarks

- Start with a PoC and focus on areas that you have good insights that are up for improvement
- Be sure that the tech stack is widely known across QA and Engineering
- Model your test based on real-life scenarios
- Observability from day one

Grafana Labs

That’s all! Thank you :)

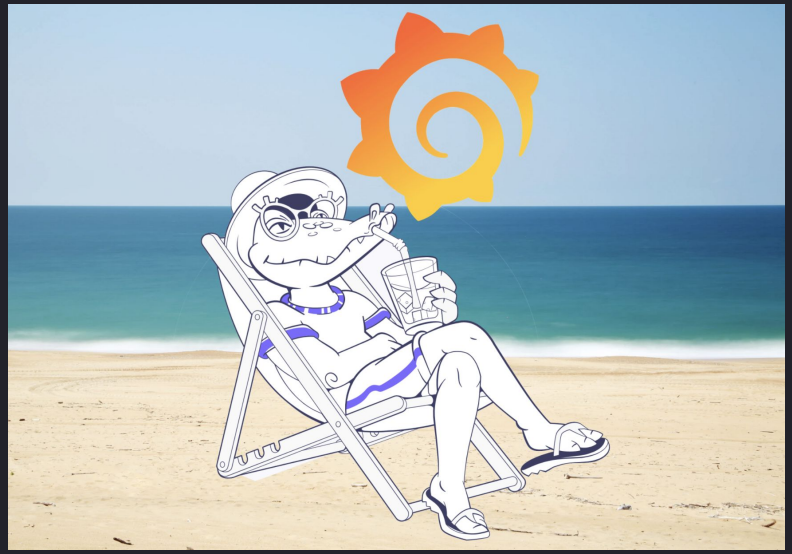
Any questions?

https://github.com/grafana/quickpizza

Grafana Labs

Get involved

- k6: k6.io / github.com/grafana/k6
- k6 documentation: k6.io/docs
- k6 YouTube Channel: youtube.com/k6test



k6.io/slack

grafana/k6

community.k6.io