Avantgarde Labs
Wir lieben Entwicklung.

9 May 2025

SCHNELLER ALS DAS BOTTLENECK

Performance-Optimierung mit K6

Dr. Markus Frank

DR. MARKUS FRANK

ARCHITECT @ AGVL

WIRTSCHAFTS-INFORMATIK
B.SC

UNIVERSITÄT MANNHEIM

WIRTSCHAFTS-INFORMATIK
M.SC

TECHNISCHE UNIVERSITÄT DRESDEN

PROMOTION

University of Stuttgart
Germany

QUALITY AND ARCHITECTURE

Model-Based Performance Prediction for Concurrent Software on Multicore Architectures

WISSENSTRANSFER-INSTITUT

3 JAHRE

TECHNOLOGIE TRANSFER INITIATIVE

FREELANCER

5 JAHRE

markuskfrank.de

mfrank@avantgarde-labs.de
mfrankTUC
dr-markus-frank-a12a83ba

Avantgarde Labs

ABOUT US

- Founded in 2007 by four students coding AI trading software
- Evolved into a 50-strong team of developers, data scientists, and consultants
- We tackle complex business problems that off-the-shelf software can't handle
- Because one size never fits all, we work with a variety of languages, frameworks, and tools

www.avantgarde-labs.de

Avantgarde Labs

ABOUT US

- Founded in 2007 by four students coding AI trading software
- Evolved into a 50-strong team of developers, data scientists, and consultants
- We tackle complex business problems that off-the-shelf software can't handle
- Because one size never fits all, we work with a variety of languages, frameworks, and tools

www.avantgarde-labs.de

LET ME TELL YOU A STORY …

Once upon a time

WHAT IS IT ABOUT?

Running example

Product Owner

ONLINE
LIBRARY
SYSTEM

ME

WHAT

Running examp

UC1: Our **USER** need to be able to lend **books**.

User

Product Owner

ONLINE LIBRARY SYSTEM

ME

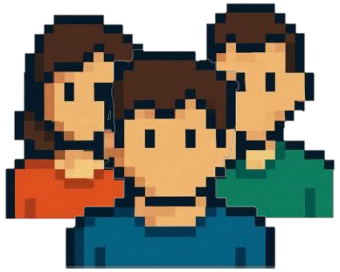
WHAT IS IT ABOUT?

Running example

Product Owner

Stakeholder:
Legal, Ops, …

6am / 8pm

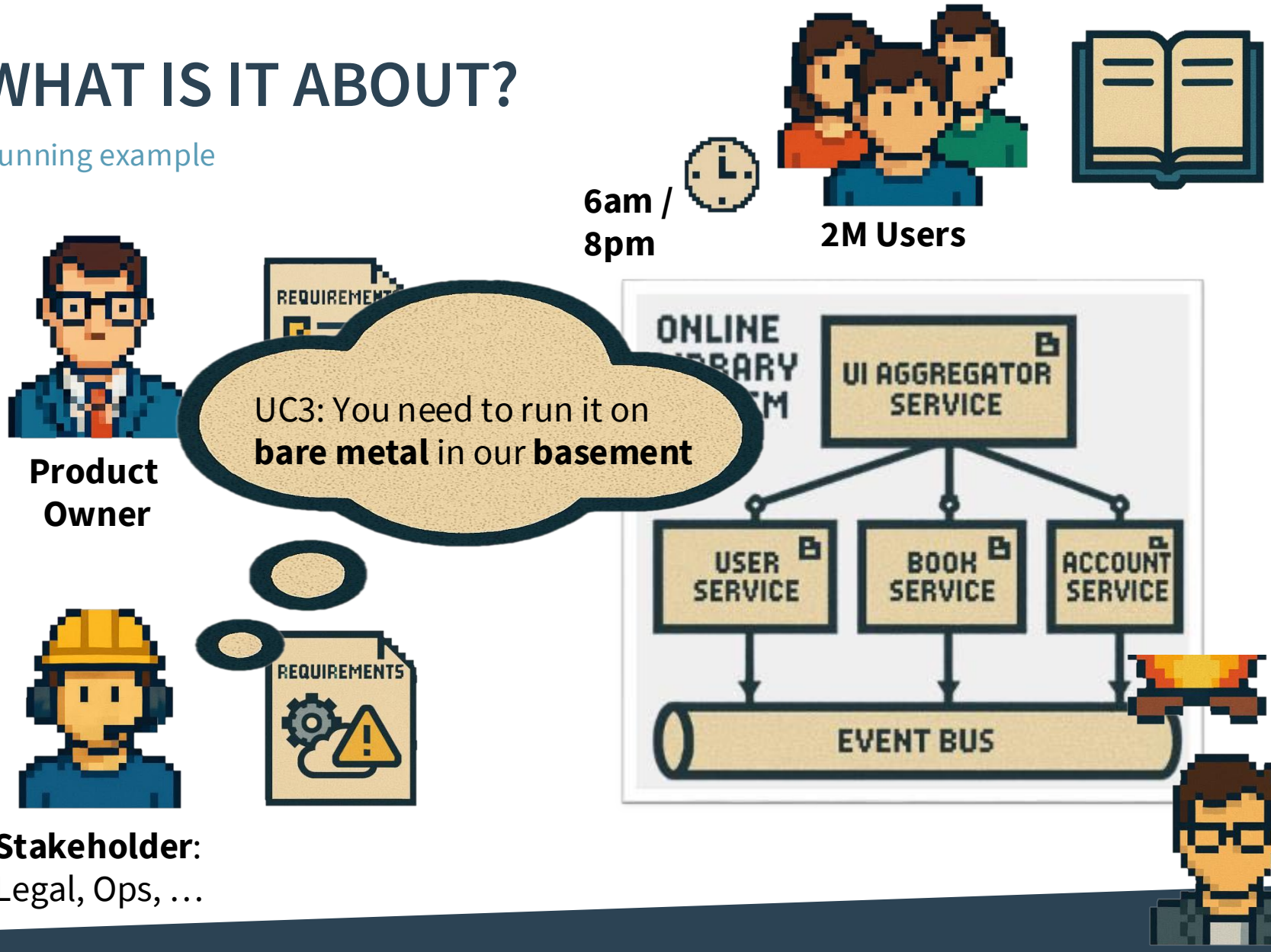


2M Users

ME

WHAT IS IT ABOUT?

Running example



WHAT IS IT ABOUT?

Running example

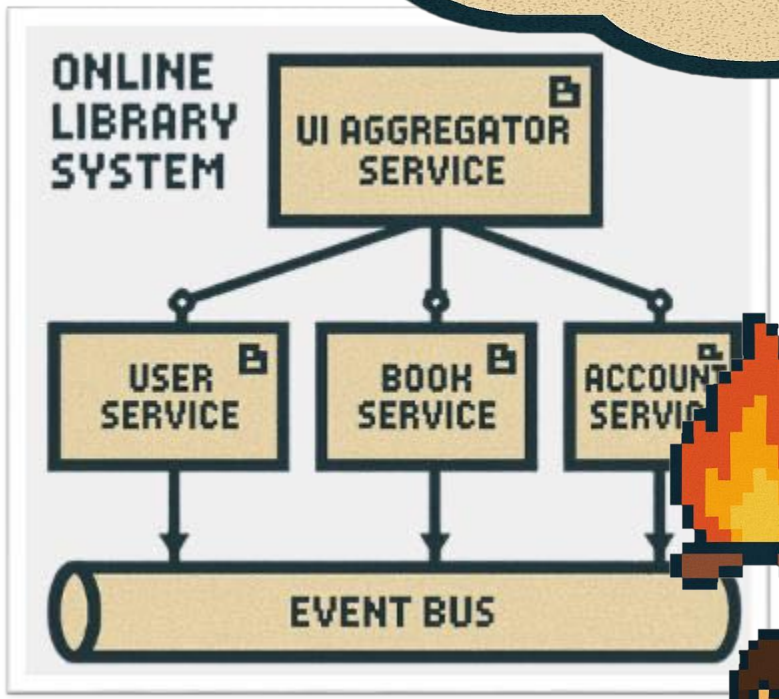
Product Owner

Stakeholder:
Legal, Ops, …

6am / 8pm

2M U

UC4: It should **always** work **without errors**



Manager

ME

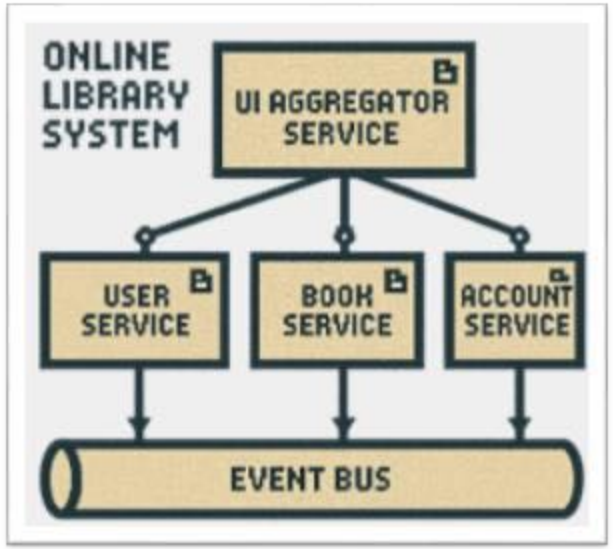
DEFINITION OF SLO‘S

Running example

Manager

Product Owner

Stakeholder:
Legal, Ops, …



Stop!
Let’s talk about SLAs first

90% of all **requests** need to have an average response time below **500 ms**

Not more than **5%** of requests are allowed to **fail**

ME

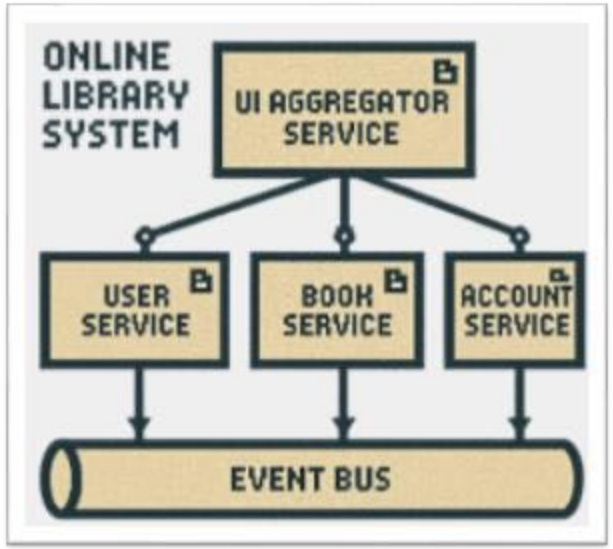
DEFINITION OF SLO‘S

Running example

Manager

Product Owner

Stakeholder:
Legal, Ops, …



90% of all **requests** need to have an average response time below **500 ms**

Not more than **5%** of requests are allowed to **fail**

What's the overall QUALITY of our software?

How can we test and ensure the quality requirements?

How can we prevent performance erosion?

ME

PERFORMANCE VS REGRESSION TESTING

Performance Testing

How can we test and ensure the quality requirements?

- Assess **responsiveness** and **stability** of a system.
- Simulates **user behavior**.
- high/specific number of **requests**.
- Tests observe:
  - System behavior
  - Collapse **thresholds**
  - Performance bottlenecks
  - **Recovery** capabilities
- Should be repeated initially and at regular intervals

Regression Testing

How can we prevent performance erosion?

- Compare method calls in a controlled, fixed environment.
- **Goal** to identify **improvements** or **regressions** in response time
- Must be executed under **consistent conditions**:
  - Same hardware
  - Same system load
  - Same test data
  - No external influences or dependencies
- Should be repeated at regular intervals

TOOLS AND FRAMEWORKS

- Quick ramp up
- Integration in Grafana
- Simple CI integration
- Lightweight

- Complex scenarios become tricky
- No build-in UI
- Growing community
- Cloud version gets pricey

k6

Grafana Labs

- Extensive protocol support
- Large community
- Mature and well established
- GUI for beginners

APACHE JMeter™

- Steep learning curve
- Resource-heavy
- No native CI/CD support

- Complex scenarios
- Support for event-driven scenarios
- Performance and low resource usage

Gatling
LOAD TESTING

- Steep learning curve
- Complex framework
- (Scala?)

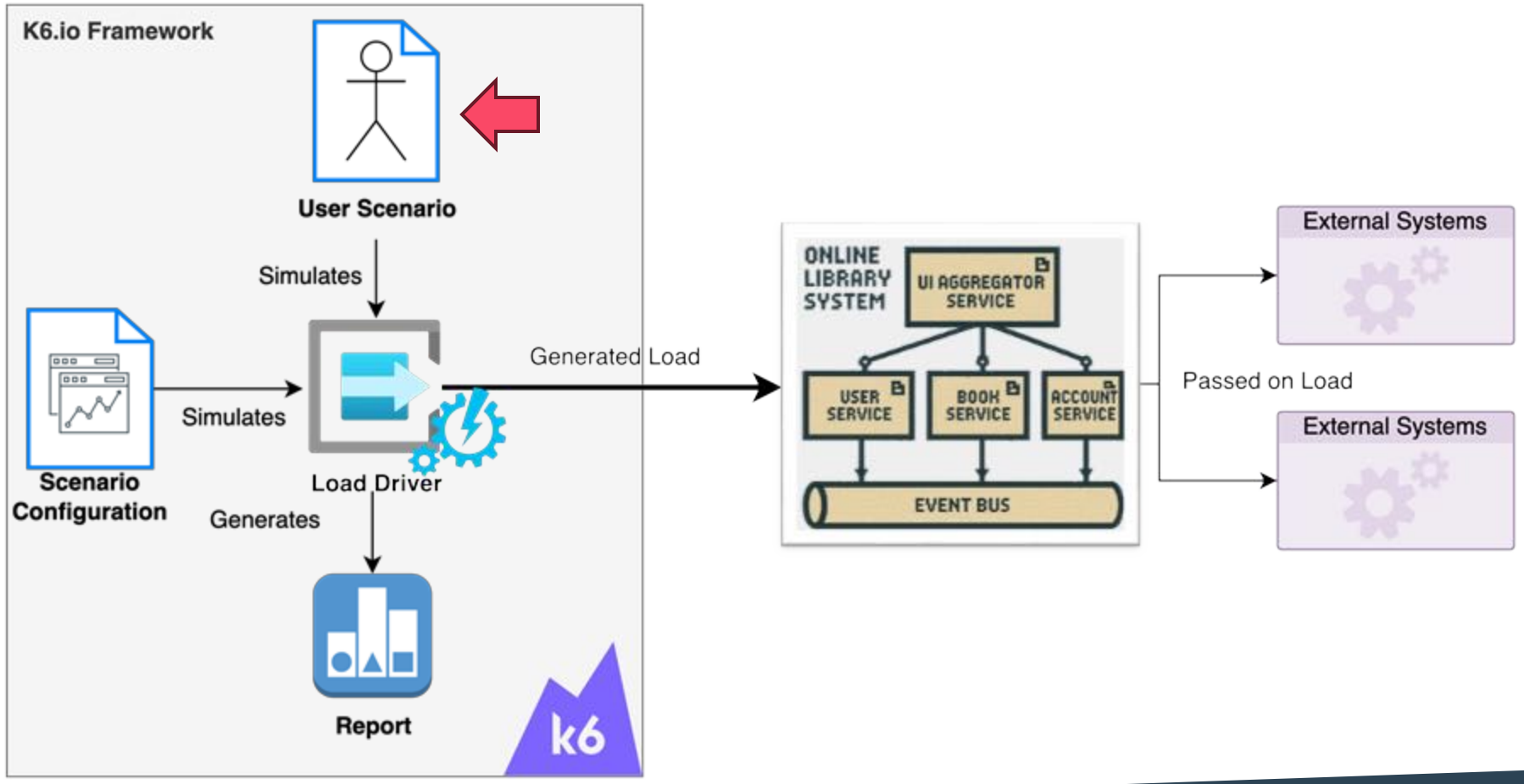
TOOLS AND FRAMEWORKS

USE WHEN:

- Dev-first, CI/CD native: K6
- Legacy systems, broad protocol: JMeter
- Complex UC + performance: Gatling

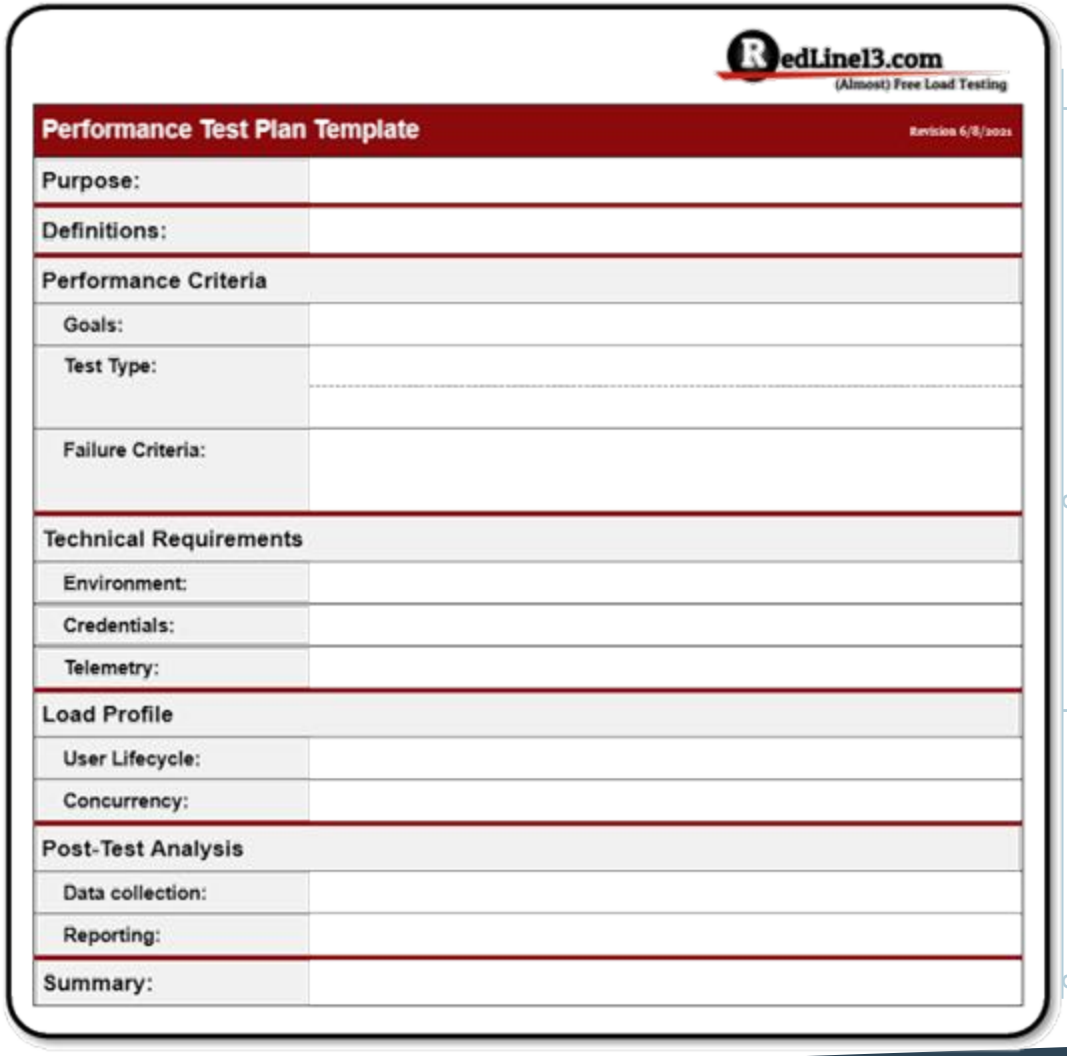
Grafana Labs

COMPONENTS OF LOAD TESTING



A GOOD TEST DESIGN

- „*Realistic*“ test scenarios
- Measurable goals
- Abstracting to reduce complexity
- Automatisation
- Have a feedback loop



RedLine13.com
(Almost) Free Load Testing

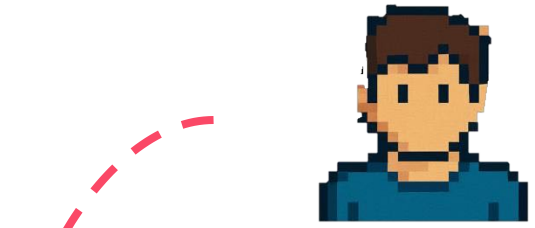
Performance Test Plan Template Revision 6/8/2021

Purpose:	
Definitions:	
Performance Criteria	
Goals:	
Test Type:	
Failure Criteria:	
Technical Requirements	
Environment:	
Credentials:	
Telemetry:	
Load Profile	
User Lifecycle:	
Concurrency:	
Post-Test Analysis	
Data collection:	
Reporting:	
Summary:	

https://www.redline13.com/blog/2021/06/test-plan-for-load-testing/

EXAMPLE: USER SCENARIO

Defines the common behaviour of the user

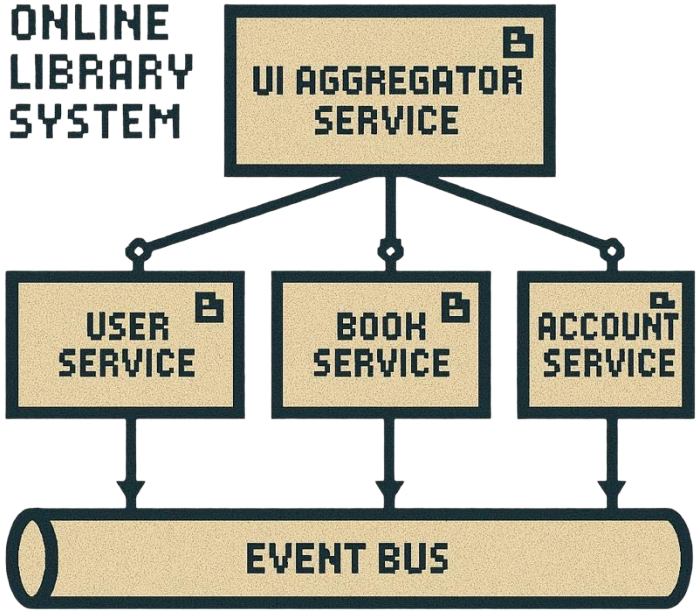


User

1. Login

Default Scenario

1. Login

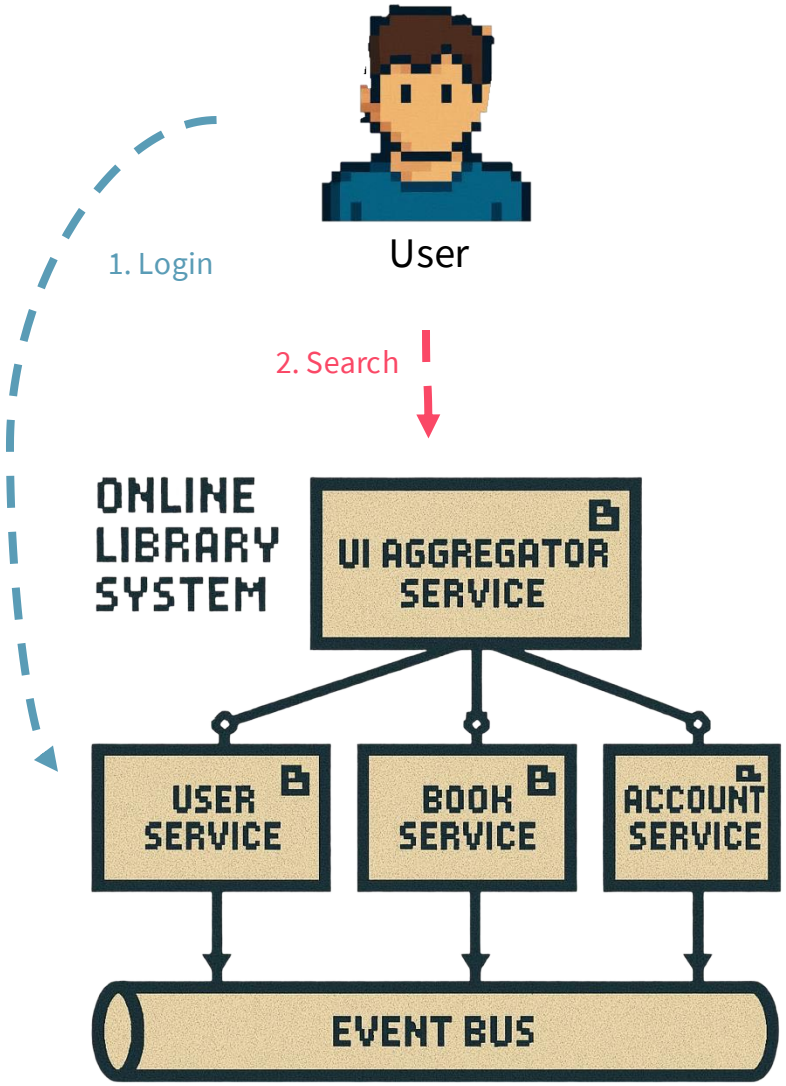


EXAMPLE: USER SCENARIO

Defines the common behaviour of the user

Default Scenario

1. Login
2. **Search**

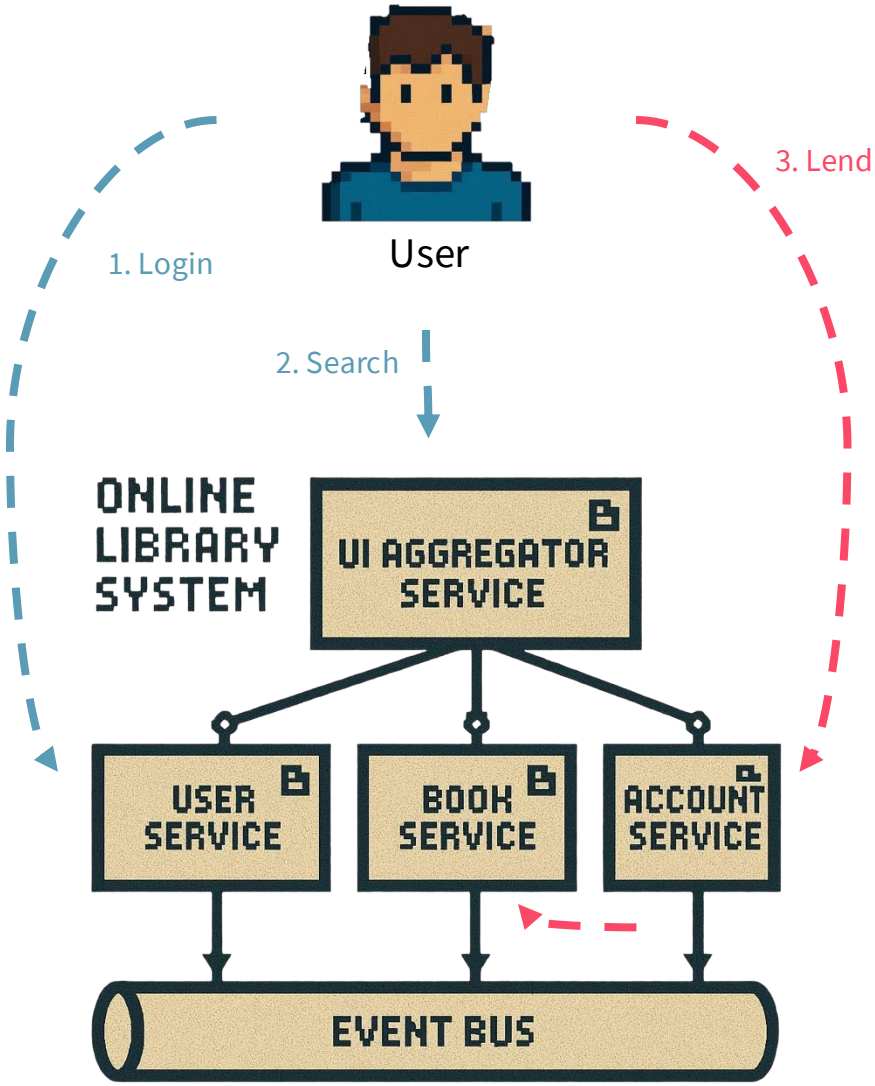


EXAMPLE: USER SCENARIO

Defines the common behaviour of the user

Default Scenario

1. Login
2. Search
3. **Lend**

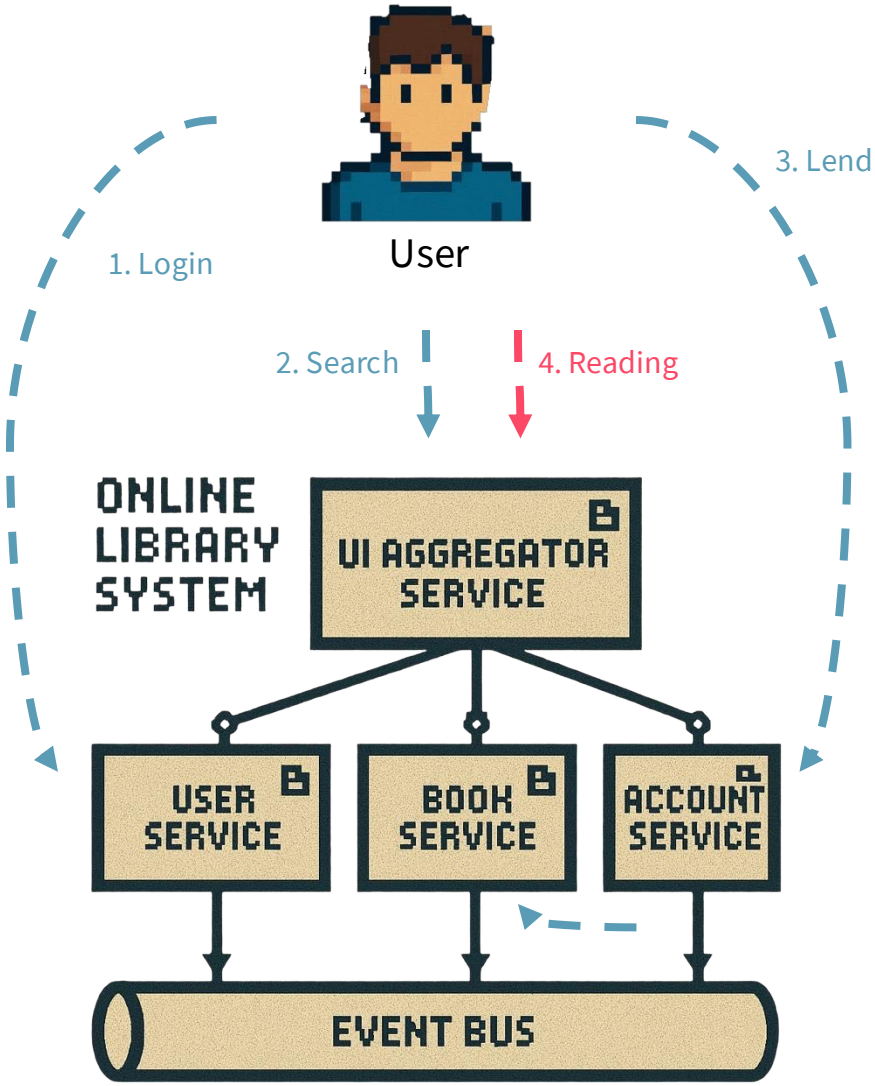


EXAMPLE: USER SCENARIO

Defines the common behaviour of the user

Default Scenario

1. Login
2. Search
3. Lend
4. **Reading**

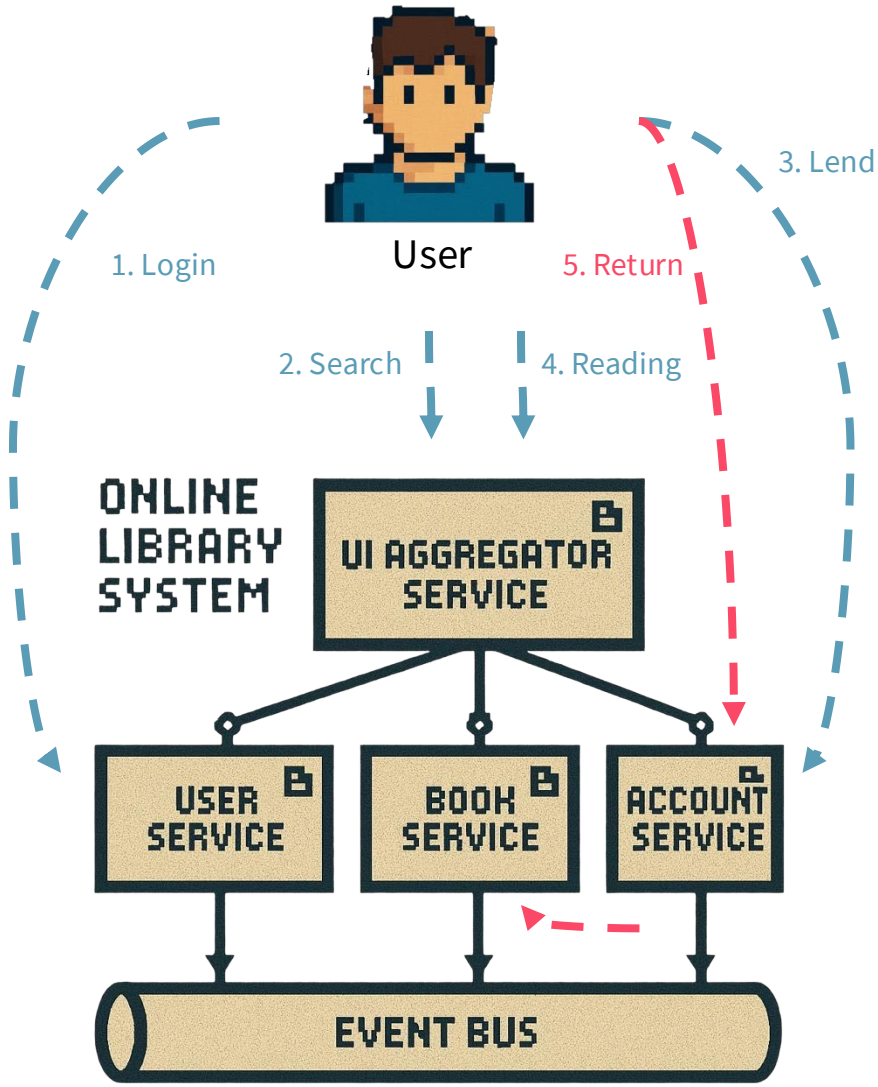


EXAMPLE: USER SCENARIO

Defines the common behaviour of the user

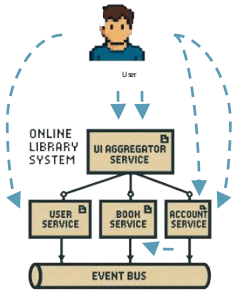
Default Scenario

1. Login
2. Search
3. Lend
4. Reading
5. **Return**



USER SCENARIO

Defines the common behaviour of the user



Scenarios define a user journey

```
export function scenarioA() {
  group('login', function () {})
  group('browse', function (){})
  group('search', function () {})
}
```

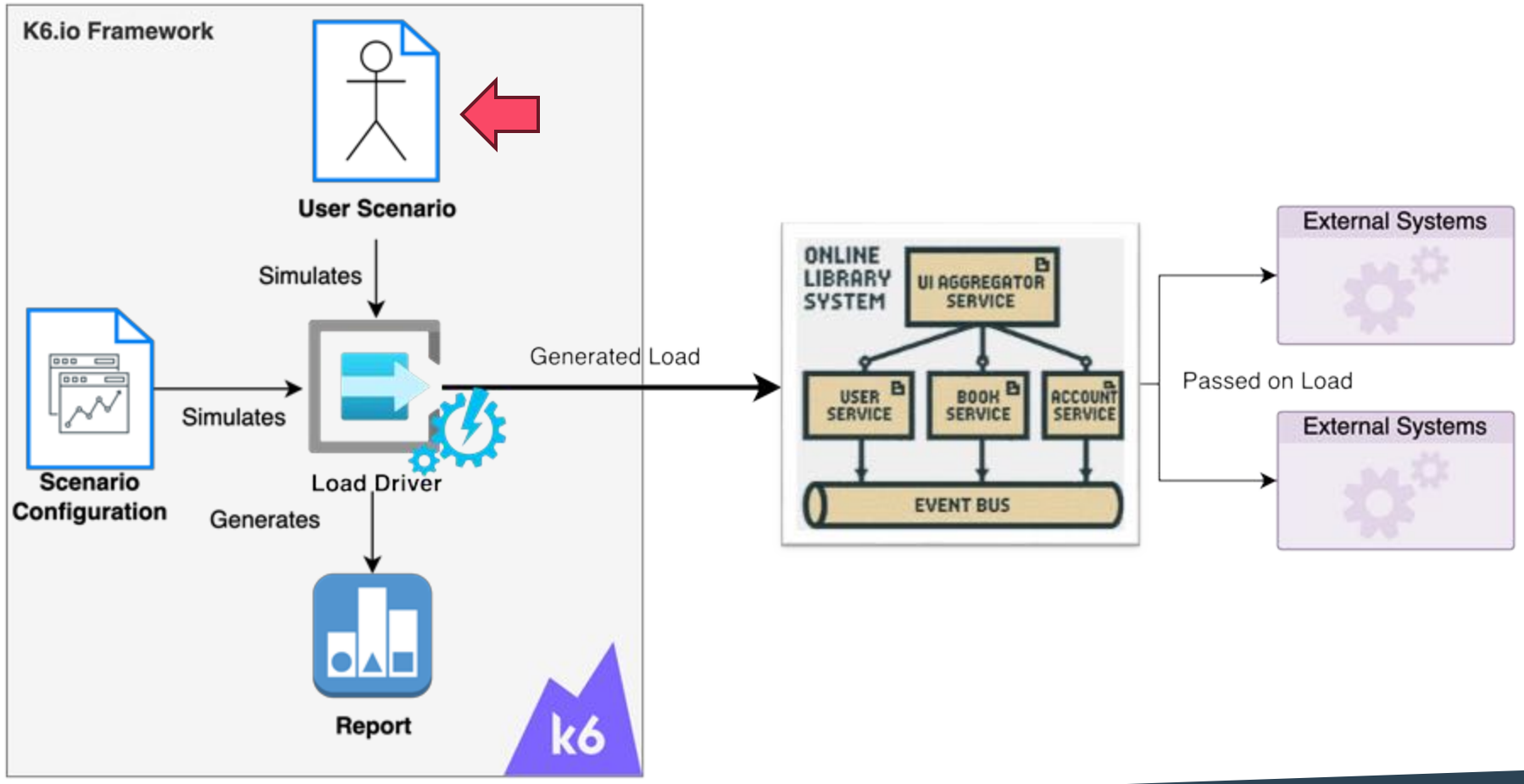
Groups help you structure code and reports

```
('login', function () {
  let login_user = getRandomData(loginUsers)
  let payload = JSON.stringify({
      n_user.username
      n_user.password,
      user.libraryId
  })

  let headers = {
    'Content-Type': 'application/json',
  }
  let response = http.post('https://${user_service_url}/v1/auth/login', payload, {
 headers: headers, tags: {name: "login"}});
  sleep(0.25); // after making the request wait
})
```

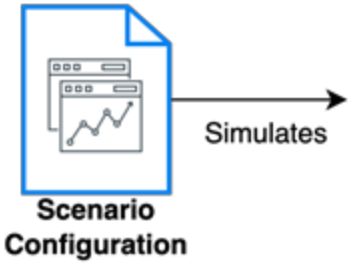
Tip: use `tags` to group requests in reports.

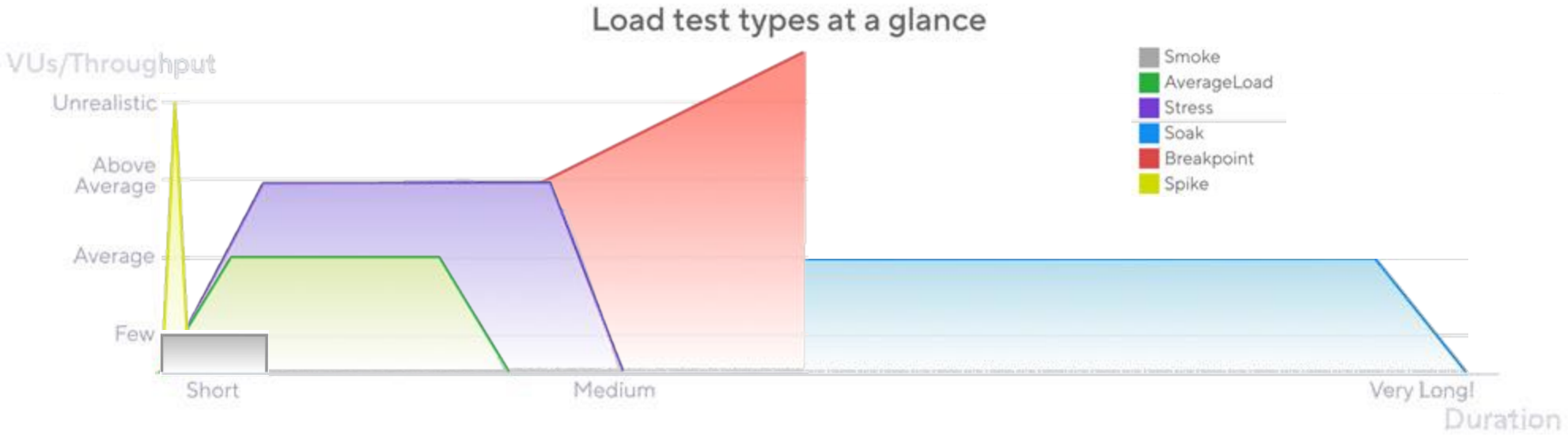
COMPONENTS OF LOAD TESTING



SCENARIO CONFIGURATION

Defines the test configuration, behaviour, and type

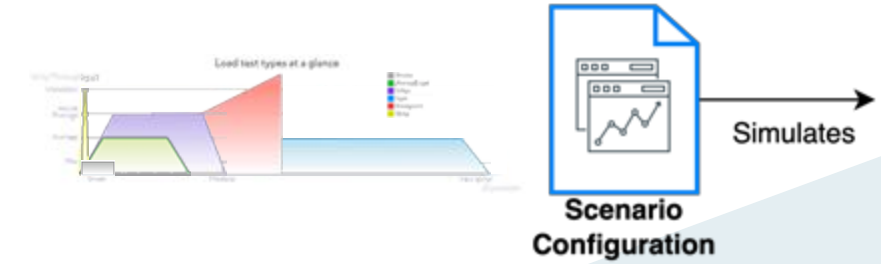


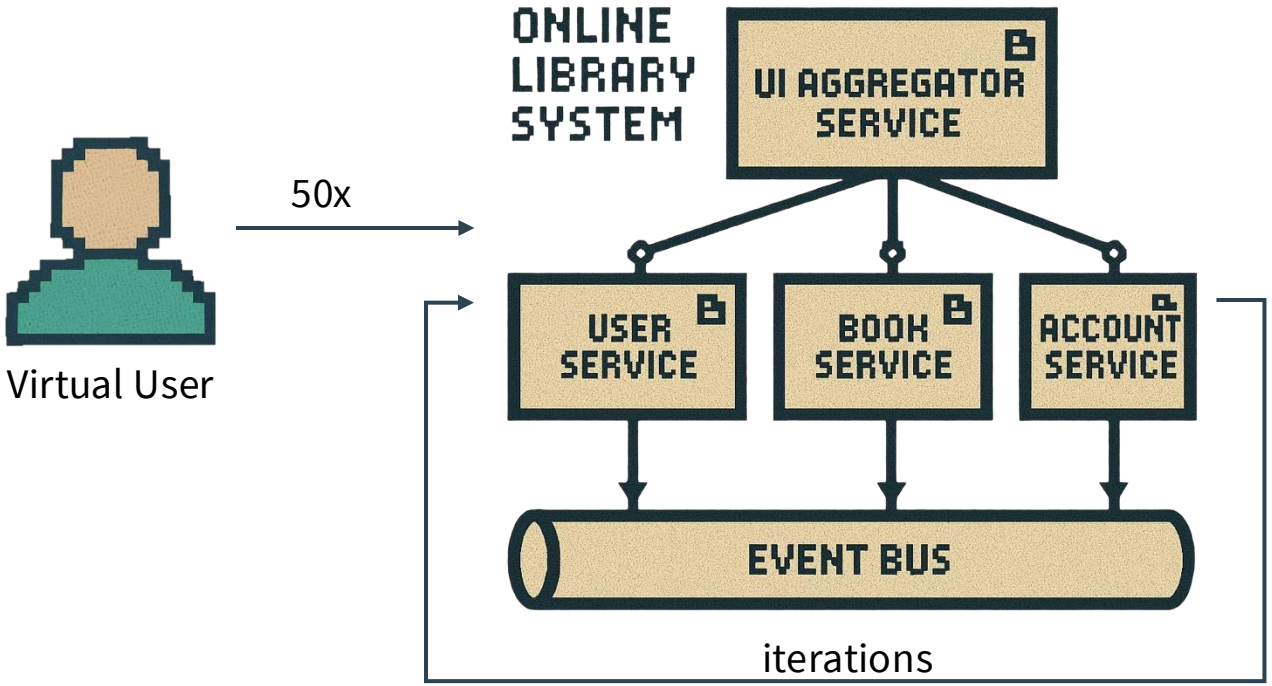


https://k6.io/docs/test-types/load-test-types/

SCENARIO CONFIGURATION

Open vs **Closed Workload**





Closed Workload

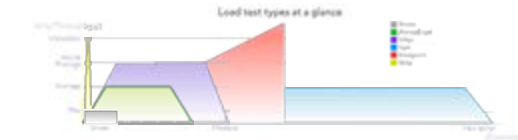
Characteristics:

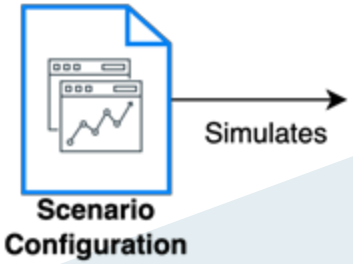
- Fixed number of **VU**
- Waits for response
- Repeat
- Simulate parallel users

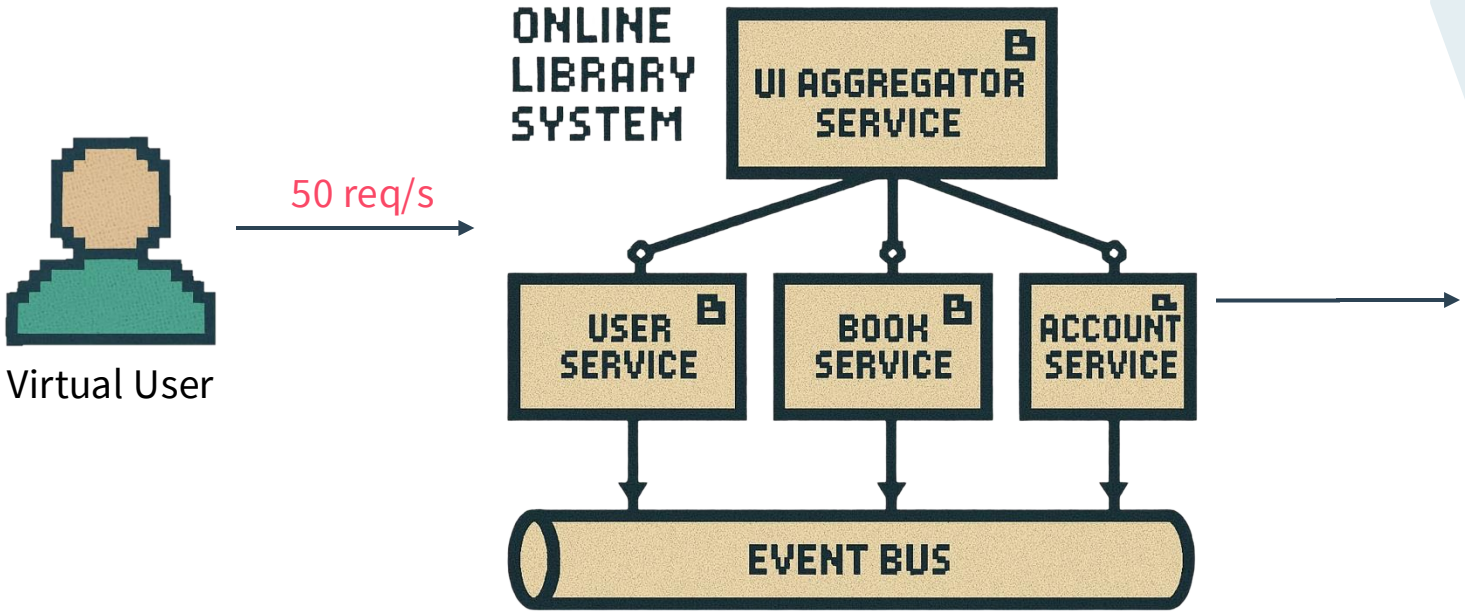
```
constant-vus
ramping-vus
```

SCENARIO CONFIGURATION

Open vs Closed Workload







Open Workload

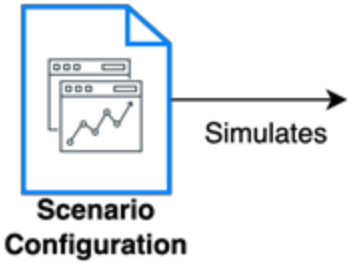
Characteristics:

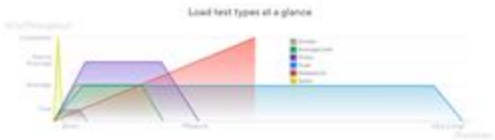
- Fixed number of **arrival rate**
- Focus on request volume
- Simulates traffic

`constant-arrival-rate`, `ramping-arrival-rate`

SCENARIO CONFIGURATION

Stress Test – Closed Workloa





Alternatives: `constant-vus`

```
stress_test: {
    executor: 'ramping-vus', // slowly
    stages: [
        { target: 20, duration: '2m' }, //
        { target: 50, duration: '2m' }, // load Stage one
        { target: 100, duration: '2m' }, // load Stage two
        { target: 100, duration: '3
    ],
    gracefulRampDown: '30s', // slow ramp down
    gracefulStop: '1m', // wai
    exec: 'scenarioA',
},
```

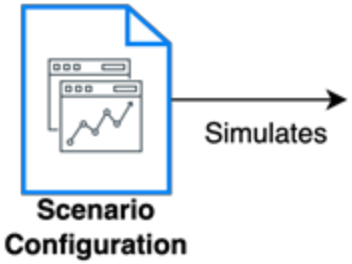
Load Test Specifications

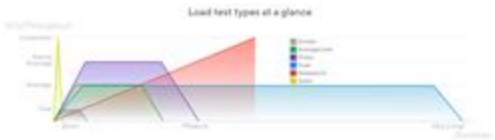
Slowly ramps down user at the end of the test.

Time to wait for open requests at the end of the test

SCENARIO CONFIGURATION

Breakpoint Test – Open Wor





Open Workload

```
break_through_test: {
  executor: 'ramping-arrival-rate', /                    n slows
  startRate: 10, // Start iterations
  timeUnit: '10s', // Start `startRate` iterations per unit
  preAllocatedVUs: 1000, // how                            e

  stages: [
    { target: 10, duration: '30s' },
    { target: 100, duration: '1m' },
    { target: 1000, duration: '2m' },
  ],

  exec: 'scenarioA',
},
```

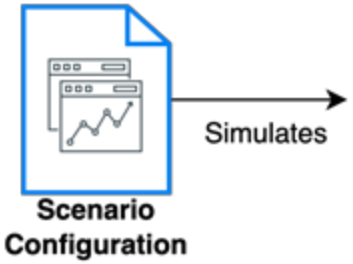
Warmup

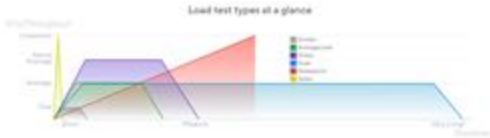
preAllocatedVUs

Slowly ramps up arrival rate

SCENARIO CONFIGURATION

Thresholds






```
thresholds: {
    http_req_failed: ['rate<0.05'], // http errors should be less than 5%
    http_req_duration: ['p(90)<500'], // 90% of requests should be below 500ms
},
```

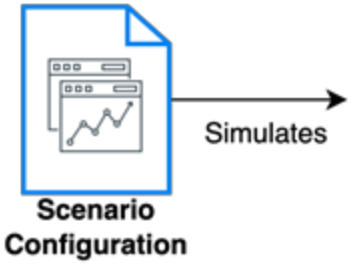
```
break_through_test: {
  executor: 'ramping-arrival-rate', //Assure load increase if thes ystem slows
  startRate: 10, // Start iterations per `timeUnit`
  timeUnit: '10s', // Start `startRate` iterations per minute
  preAllocatedVUs: 1000, // how large the initial pool of VUs would be

  stages: [
    { target: 10, duration: '30s' },
    { target: 100, duration: '1m' },
    { target: 1000, duration: '2m' },
  ],

  exec: 'scenario_1',
},
```

SCENARIO CONFIGURATION

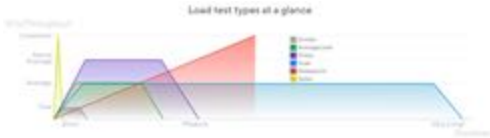
Custom Checks

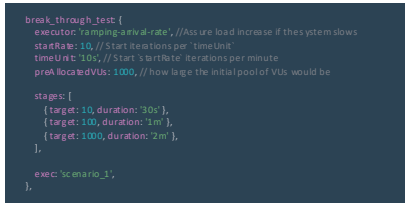


Checks do not fail tests by default

Definition of custom check:

```
check(response, {
   'Lend completed in time': (r) =>
      r.body.includes('DONE'),
});
```



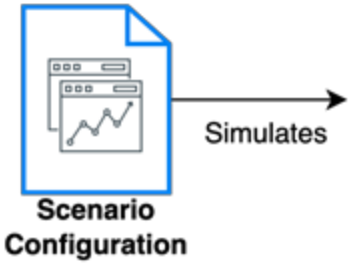


Ensuring Checks

```
thresholds: {
   http_req_failed: ["rate<0.05"], // http errors should be less than 5%
   http_req_duration: ["p(90)<500"], // 90% of requests should be below 1s
   checks: ['rate > 0.99'],  // Ensure high success rate for checks
}
```

SCENARIO CONFIGURATION

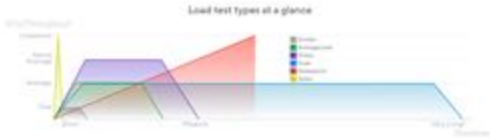
Custom Checks

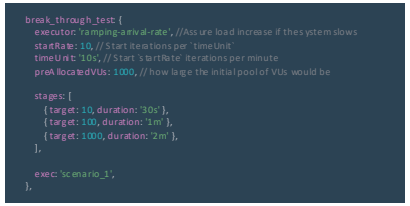


Checks do not fail tests by default

Definition of custom check:

```
check(response, {
   'Lend completed in time': (r) =>
      r.body.includes('DONE'),
} , { myTag: 'lendTime' });
```





Thresholds can be groupd by tags

Ensuring Checks

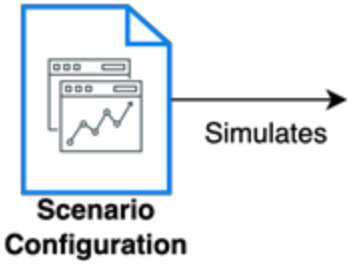
```
thresholds: {
   http_req_failed: ["rate<0.05"], // http errors should be less than 5%
   http_req_duration: ["p(90)<500"], // 90% of requests should be below 1s
   "checks{myTag:lendTime}": ["rate > 0.99"], // Ensure high success rate for checks
}
```

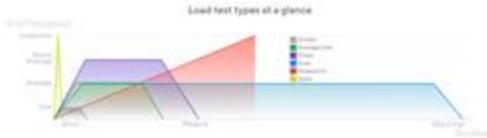
SCENARIO CONFIGURATION

Distribution

Only available in K6 Cloud

```
cloud: {
  name: 'Online Library Default User Test',
  projectID: 12345678,
  distribution: {
    distributionLabel1: { loadZone: 'amazon:de:frankfurt', percent: 50 },
    distributionLabel2: { loadZone: 'amazon:it:milan', percent: 50 },
  },
}
```



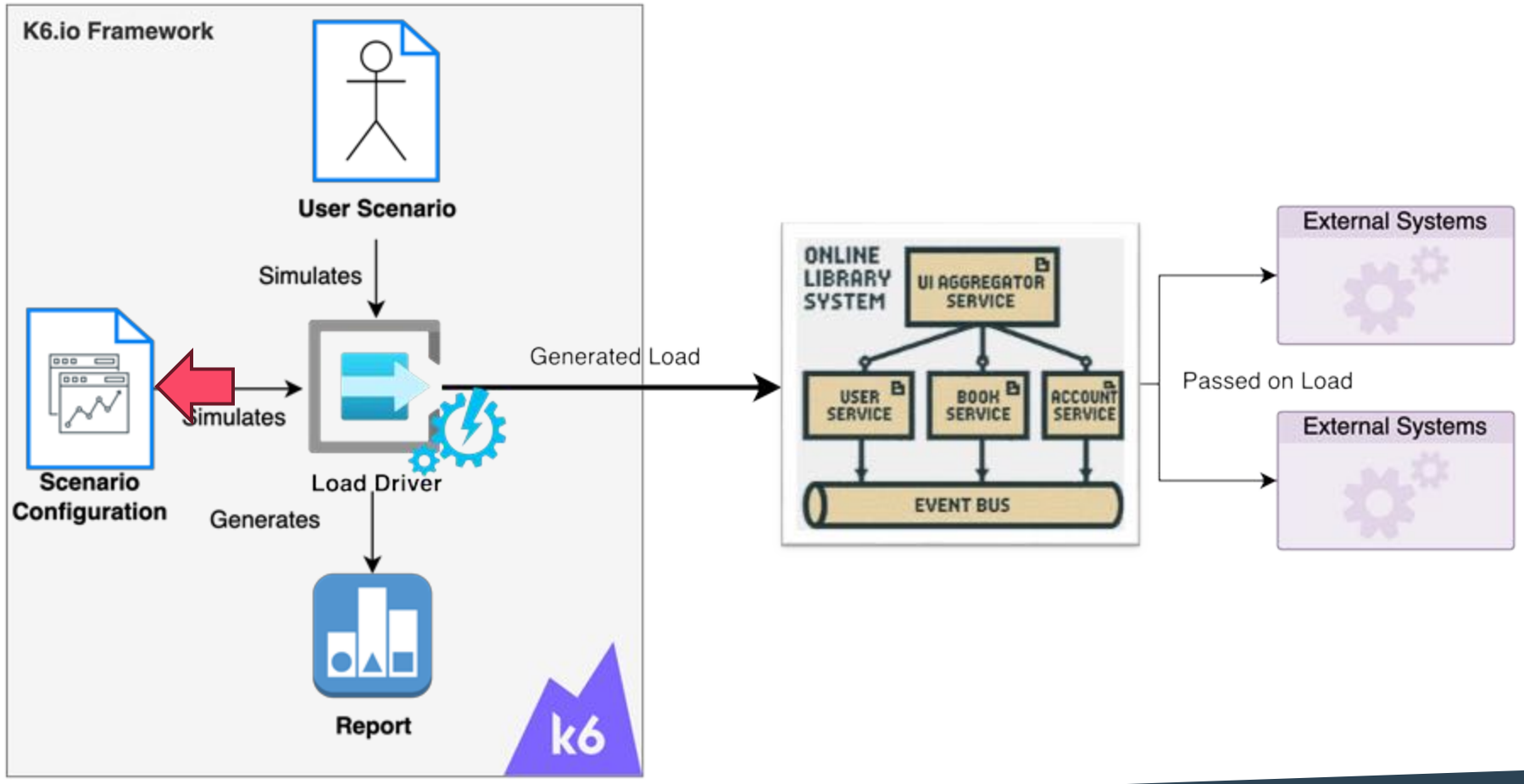



```
break_through_test {
  executor: 'ramping-arrival-rate', //Assure load increase if thes ystem slows
  startRate: 10, // Start iterations per `timeUnit`
  timeUnit: '10s', // Start `startRate` iterations per minute
  preAllocatedVUs: 1000, // how large the initial pool of VUs would be

  stages: [
    { target: 10, duration: '30s' },
    { target: 100, duration: '1m' },
    { target: 1000, duration: '2m' },
  ],

  exec: 'scenario_1',
},
```

COMPONENTS OF LOAD TESTING



LIVE DEMO

LIVE DEMO

End

GOOD TO KNOW

Twelve takeaways to remember

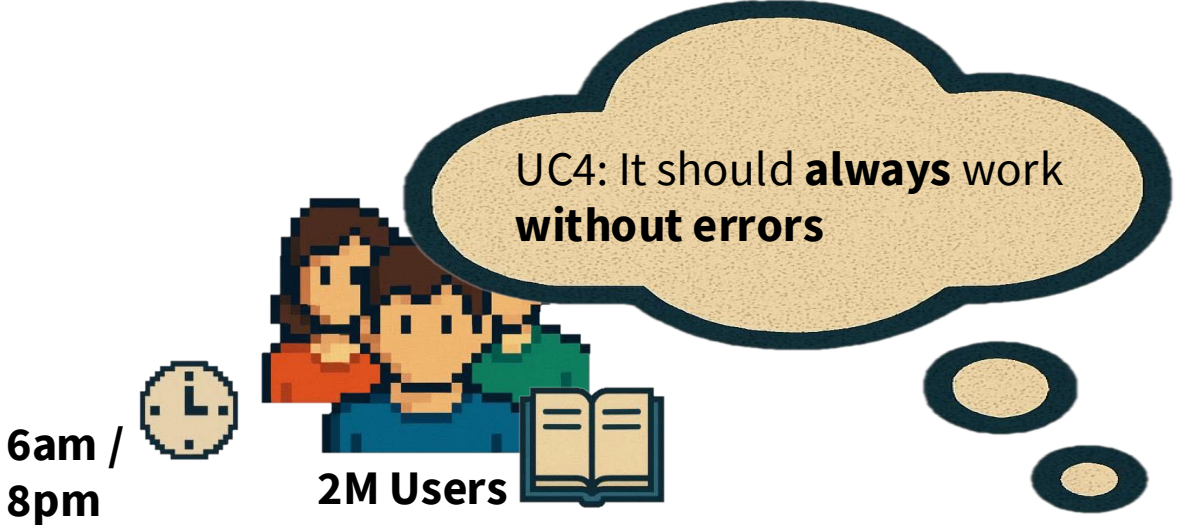
There are no guarantees …

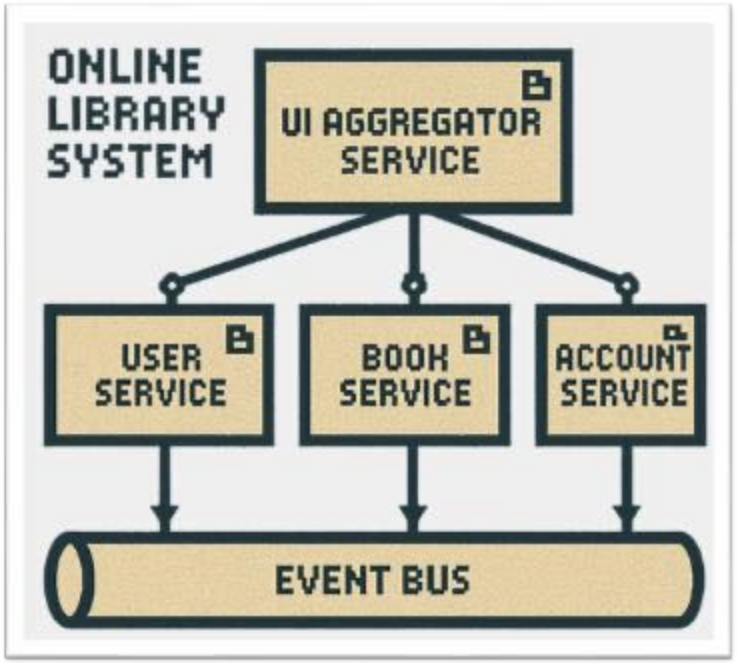
1. Keep test **simple**
    1. VUs are not real users
    2. Use sleep() wisely to simulate think times
2. Do **not** mistake for **functional tests**
3. Use **multiple** scenarios for different user journeys
4. Remember **3rd-party** service impact
5. Use **realistic** environments
6. **Ramp-Up** matters
7. Test data creation can be tricky
8. K6 Browers is available for **End2End** tests
9. K6 Browers Plugin can >**record**< **scenarios**
10. Yes, you can also write **TypeScript** (via esbuild)
11. **Limited npm** support (no Node.js core modules).
12. Remember to use **`setup()`** or **`teardown()`**

LESSONS LEARNED

90% of all requests need to have an average response time below 500 ms

Not more than 5% of requests are allowed to fail

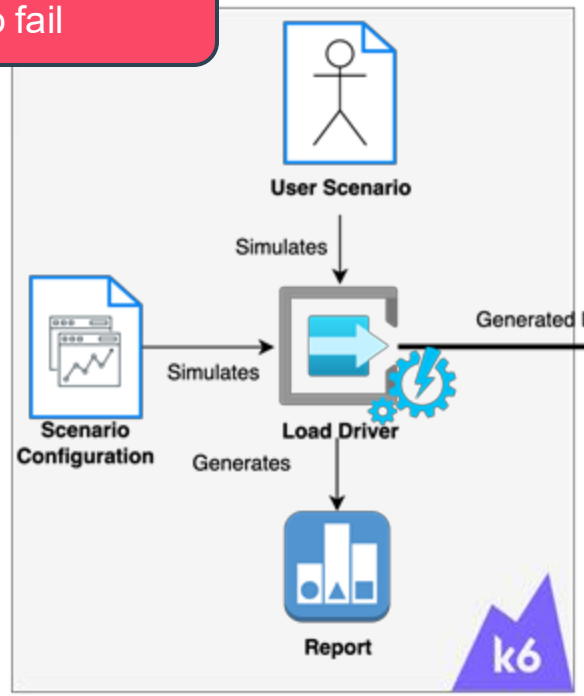


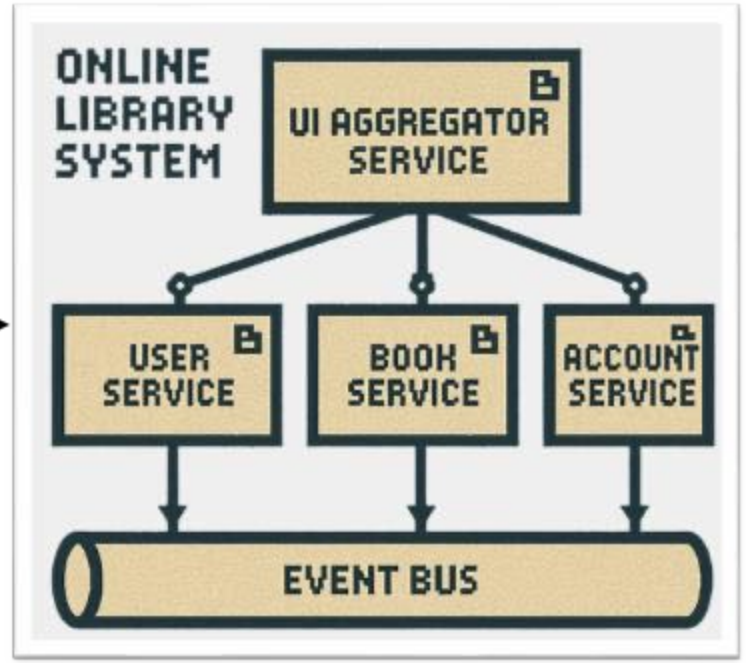


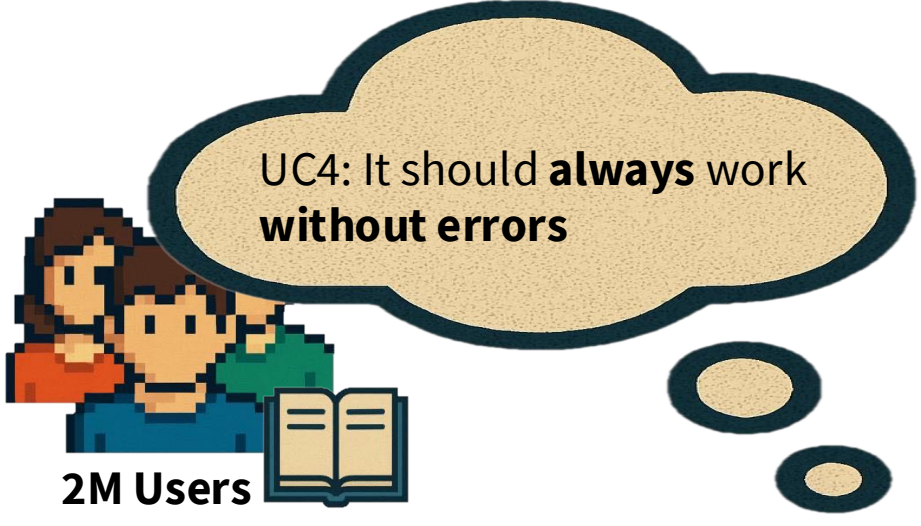
LESSONS LEARNED

90% of all requests need to have an average response time below 500 ms

Not more than 5% of requests are allowed to fail







6am / 8pm

2M Users

Manager

ME