SOFTWARE ARCHITECTURE

Escuela de Ingeniería Informática

Universidad de Oviedo

Lab 11

Monitoring and profiling: observability

2025-26

Jose Emilio Labra Gayo
Pablo González
Diego Martín
Celia Melendi

Monitoring and profiling

Quality attribute: Observability

Monitoring: Observe the behaviour at runtime while software is running

- Dashboards
- Usually in production (after deployment)

Profiling: Measure performance of a software while it is running

- Identify parts of a system that contribute to performance problems
- Find where to focus the efforts to improve performance
- Usually when developing/testing (before deployment)

Profiling

Monitors an application while it is running

Records performance (CPU & memory usage)

JavaScript:

- Chrome (Timeline), Firefox Developer Edition (Performance tool)

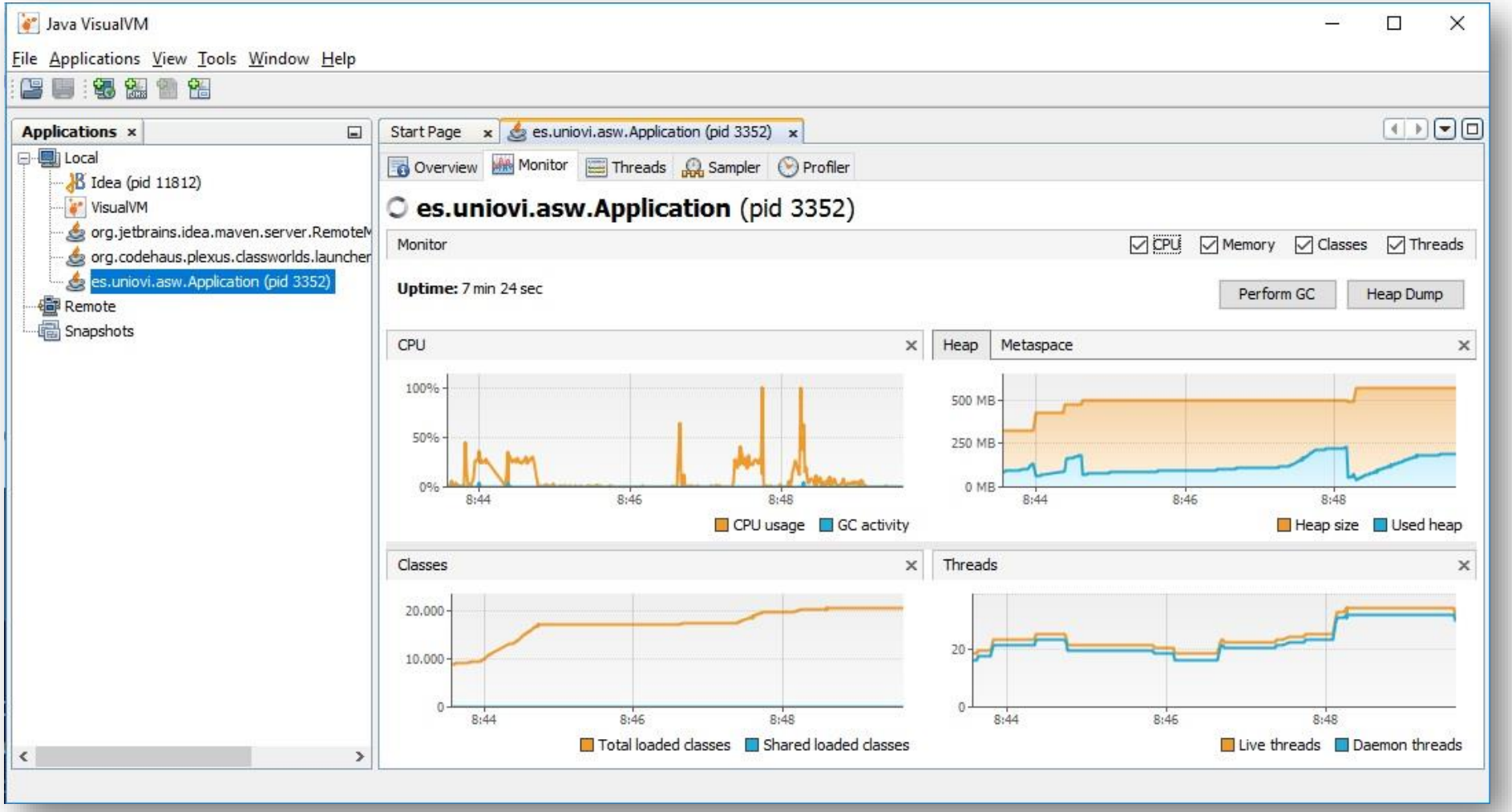
Server-side:

- JVisualVM, JProfiler, YourKit, JConsole
- Monitoring: Graphite, Datadog, Prometheus, Graphana

VisualVM

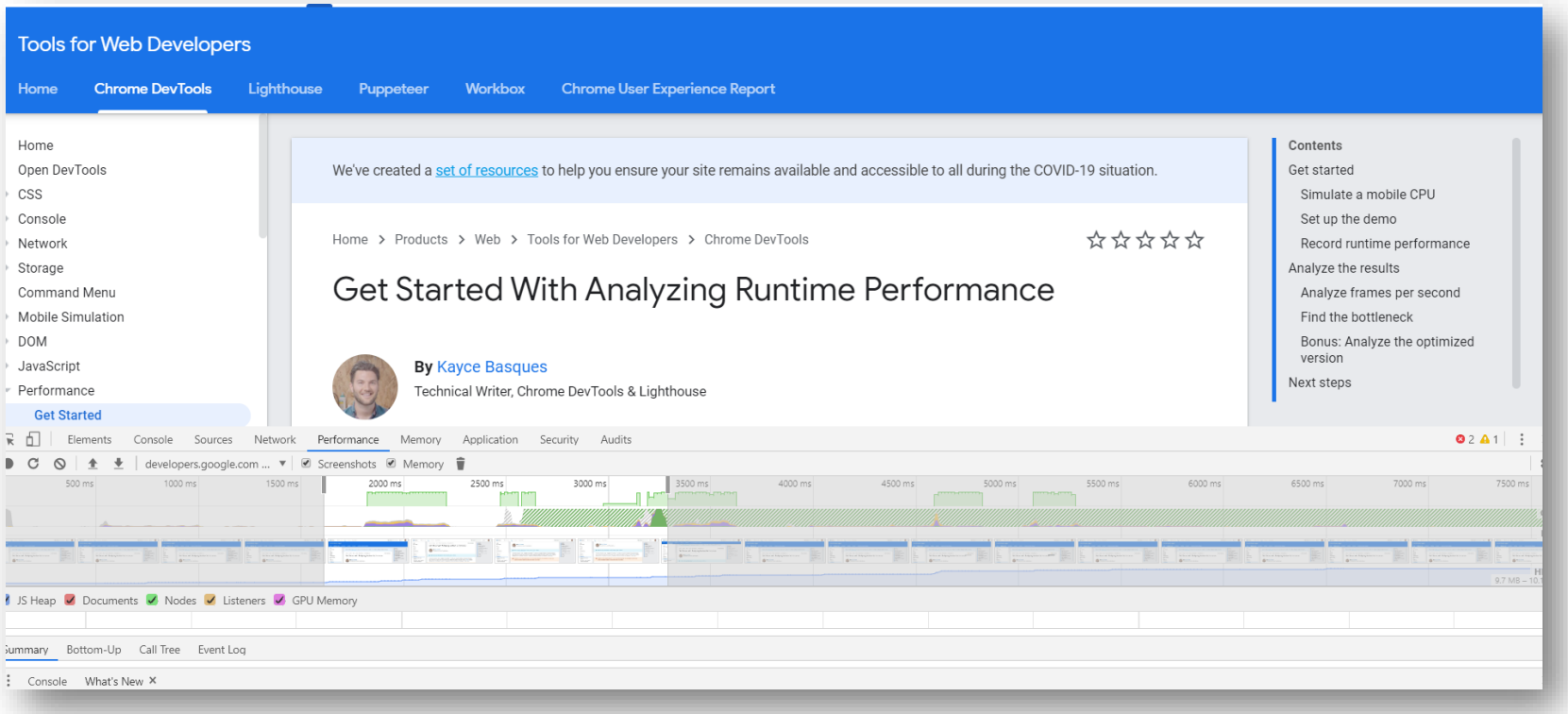
- https://visualvm.github.io/
- `jvisualvm`

Java/server JVisualVM



Browser: developer tools

Profiling/check performance



https://developers.google.com/web/tools/chrome-devtools/evaluate-performance

Example with Google Chrome

Incognito mode

At the top right, click the three dots and then New Incognito Window.

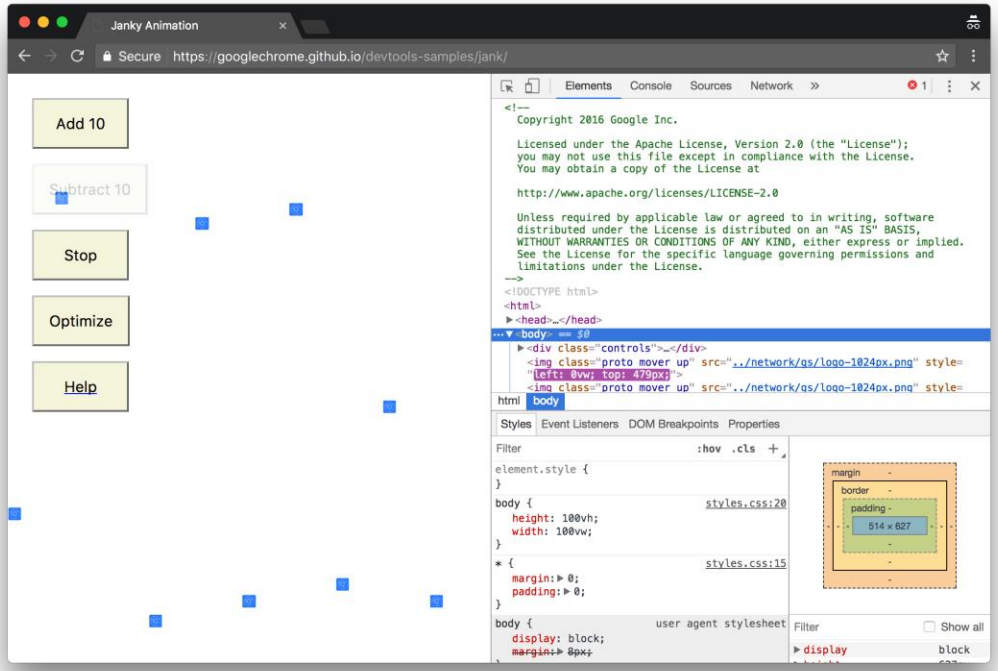
Windows, Linux, or Chrome OS: Press Ctrl + Shift + n.
Mac: Press ⌘ + Shift + n.

DevTools

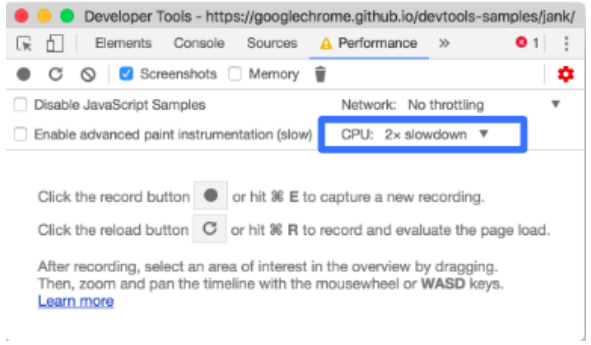
Windows, Linux: Control+Shift+I
Mac: Command+Option+I

Example with Google Chrome

https://googlechrome.github.io/devtools-samples/jank/



Performance>CPU>2 x Slowdown



Performance>Record

click Add 10 (20 times)

try Optimize / Un-optimize

Stop

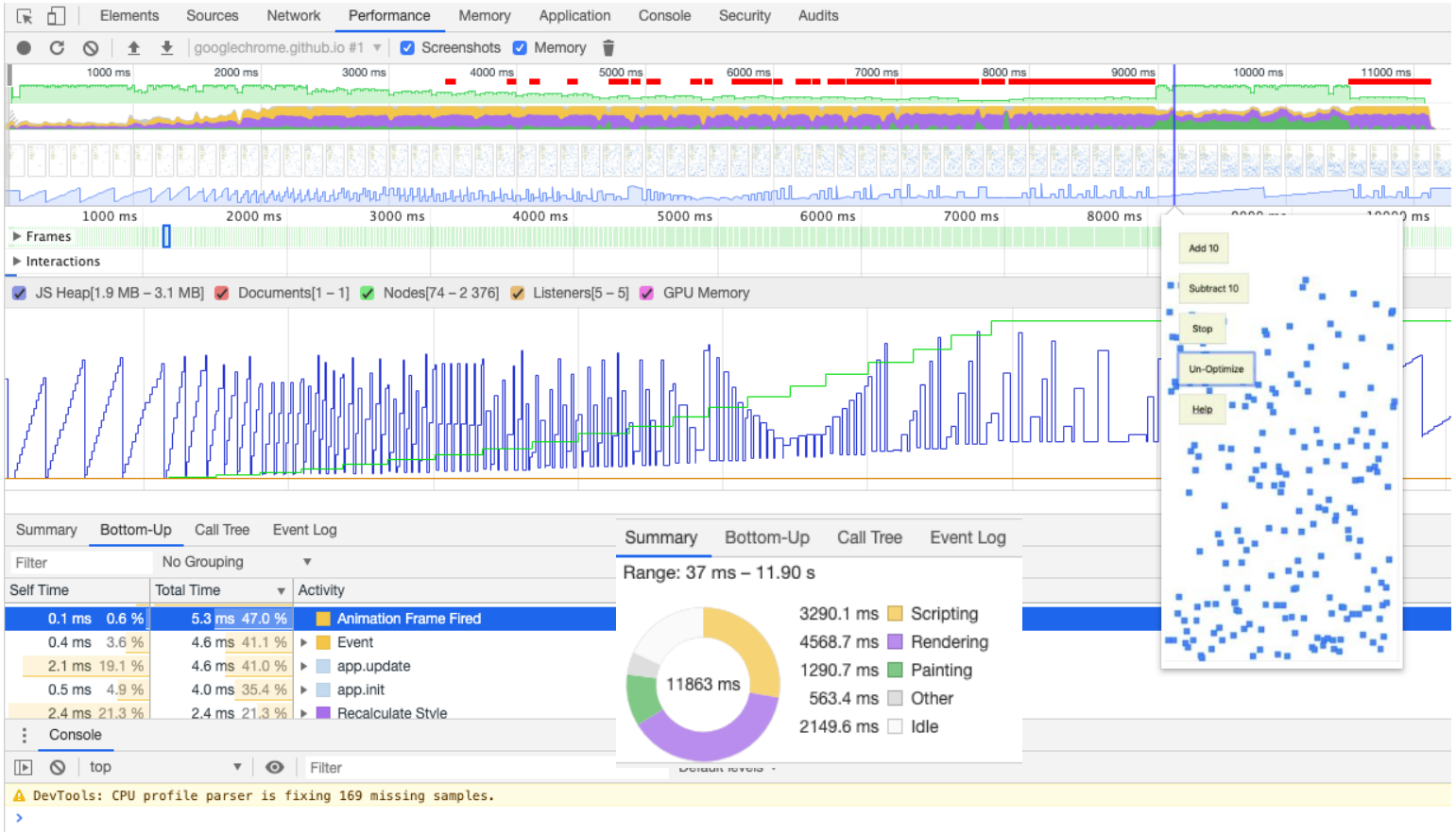
Example with Google Chrome

Profile result:

Frames per Second

CPU

Bottleneck



Other tools for browser

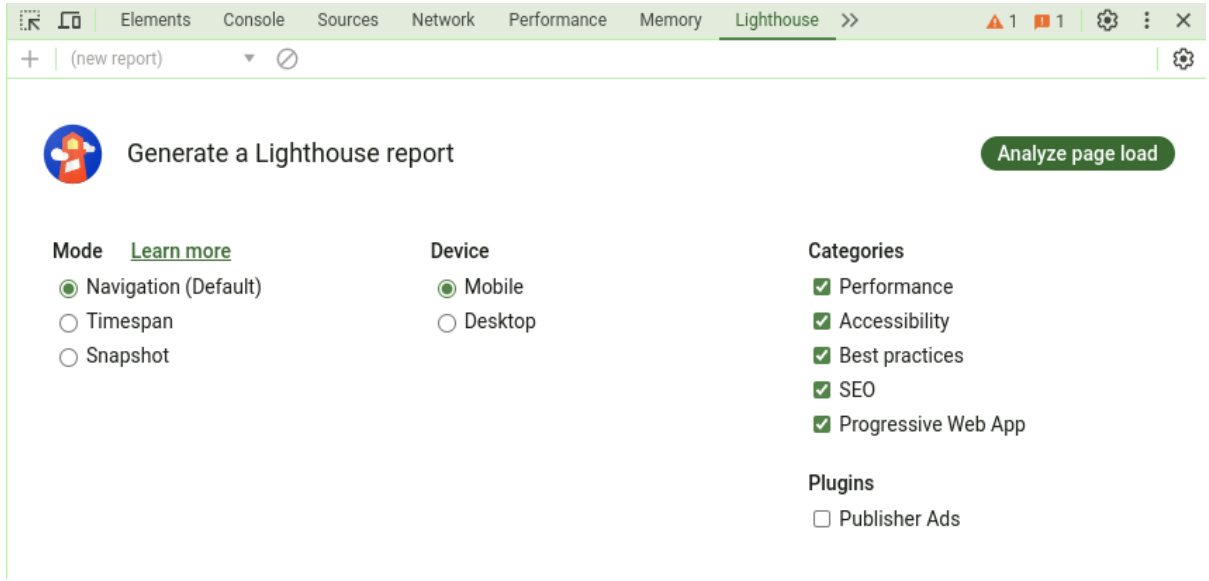
RAIL model:

Response, Animation, Idle, Load

https://developers.google.com/web/fundamentals/performance/rail

https://webpagetest.org/easy

Lighthouse (with Chrome)



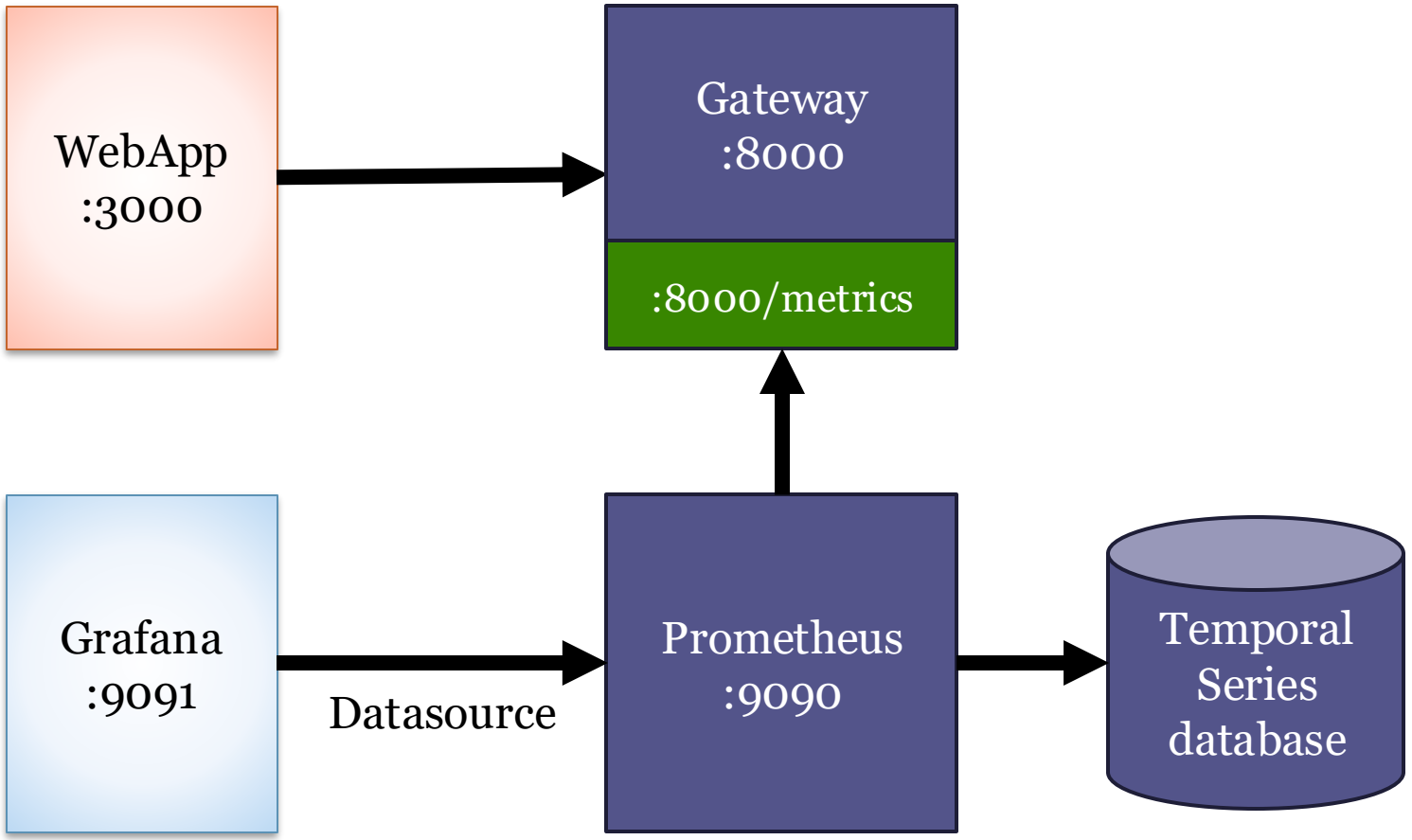
Server side monitoring

Cloud platforms like Azure provide monitoring solutions

- Also available in Google Cloud, Amazon AWS, Alibaba Cloud…
- In the case of Azure: Azure Monitor

We can also set up our own monitoring solution

Typical software: *Prometheus* and *Grafana*


WebApp
:3000
Gateway
:8000
:8000/metrics
Grafana
:9091
Datasource
Prometheus
:9090
Temporal
Series
database

Server side monitoring

We use a library to extract metrics from gatewayservice

npm install prom-client express-prom-bundle

```
const metricsMiddleware:RequestHandler = promBundle({includeMethod: true});
app.use(metricsMiddleware);
```

If we launch the userservice, in */metrics* we can see raw data can be used by Prometheus to store it and by Grafana to plot nice charts

We can choose which metrics to measure [doc]

Server side monitoring

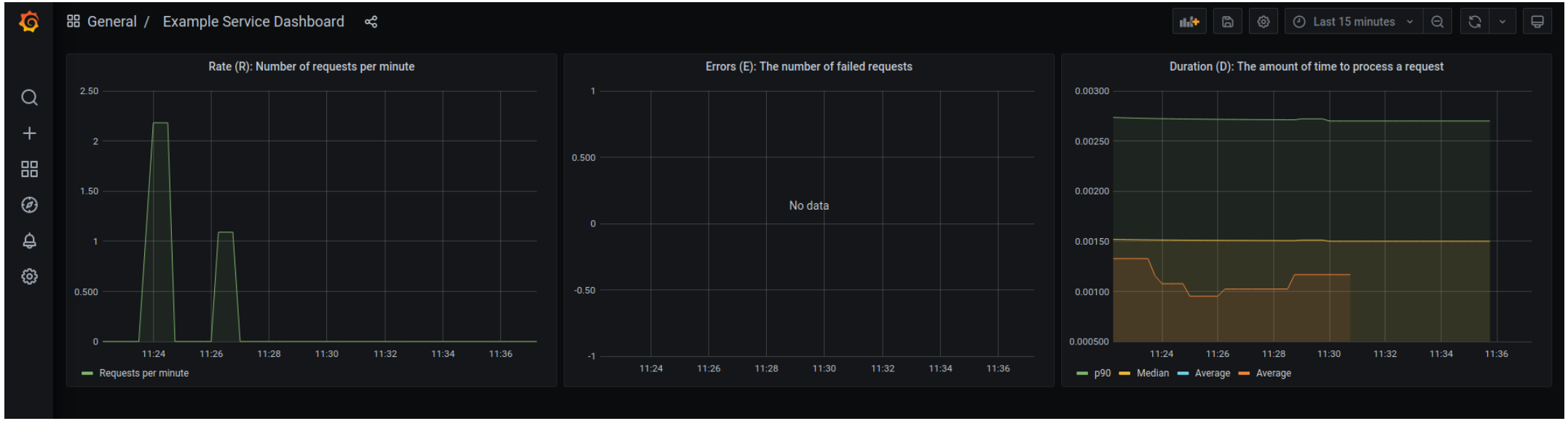
- Grafana cannot use this data directly, we need Prometheus
  - Prometheus retrieves data exposed by a service (e.g. userservice) and stores it in a time series database so it can be consumed by Grafana
  - We configured a docker image [prom/prometheus] with a single file

```
global:
  scrape_interval: 5s

scrape_configs:
  - job_name: 'users-service'
    static_configs:
      - targets: ['users:3000']
```

Server side monitoring

- How to configure Grafana
  - Grafana will use Prometheus as data source
  - We also have a docker image for running it [grafana/grafana]
  - We can configure datasource and dashboard (which charts to plot)



Example of Real Grafana Dashboards

https://grafana.wikimedia.org/

Links

Monitoring & Profiling

Get Started With Analyzing Runtime Performance

https://developers.google.com/web/tools/chrome-devtools/evaluate-performance/

How to Use the Timeline Tool

https://developers.google.com/web/tools/chrome-devtools/evaluate-performance timeline-tool#profile-js