LU13b - Monitoring

Es gibt viele Möglichkeiten, um gehostete Systeme zu überwachen, Logs auszuwerten etc.

Bei vielen Dockerumgebungen wird Kubernetes eingesetzt, welches für die meisten Punkte bereits geeignete Funktionen besitzt. Zusätzlich existiert Software wie Splunk, um immensen Datenmenge aller Systeme auszuwerten und herunterzubrechen.

Für unseren Use-Case wäre der Einsatz solcher Software Overkill, dennoch wollen wir das Monitoring nicht ignorieren.

Für Backends empfiehlt es sich einen Healthcheck-API einzubauen. Dadurch können Sie einfach testen, ob die Applikation noch läuft, ohne dass sie eine produktive (und allenfalls komplexe) API aufrufen müssen.

```
@app.route("/health")
    def health():
        return {"status": "ok"}, 200
```

```
CURL http://ec2-54-80-83-95.compute-1.amazonaws.com/api/cicd/develop/health
```

Aber auch bei statischen Frontends kann ein entsprechender Check eingebaut werden, zum Beispiel mit einer JSON-Datei.

```
{
    "message": "online",
    "color": "green"
}
```

```
CURL https://alexanderpeter.github.io/ffit-lu08-club-accounting-
website/status.json
```

Shields.io

Github, Gitlab etc. bieten oft bereits Badges an, aber nur beschränkt. Die Webseite https://shields.io/ schafft da Abhilfe und bietet zusätzliche (statische und dynamische) Badges an.

Statische Badges Statische Badges sind im Prinzip hardcoded und dienen dazu vom Build unabhängige Informationen darzustellen. Shields.io erlaubt den Inhalt via URL-Parameter zu setzen: https://img.shields.io/badge/any%20text-you%20like-blue

Dynamische Badges Eine Auswahl dynamischer Badges finden Sie unter: https://github.com/AlexanderPeter/cicd

Shields.io erlaubt, dass Farbe, Label, Nachricht Style, Logo etc. beliebig via statischen URL-Parametern und dnamischen JSON-Werten gesetzt werden können. https://shields.io/badges/dynamic-json-badge

Dadurch istes möglich, Status, Coverage, Versionen, Downloads etc. dynamisch abzufragen und eine entsprechende Bilddatei zu generieren. Diese Möglichkeit können wir ebenfalls nutzen, um einen System-Healthcheck grafisch darzustellen.

Bei folgendem Beispiel wird immer „ Frontend status“ als Label verwendet. Ein separater Style oder ein Logo wird nicht verwendet. Die Farbe und die Nachricht werden von dem JSON mit der angegebenen URL geholt.

```
![Frontend
status](https://img.shields.io/badge/dynamic/json?url=https%3A%2F%2Falexande
rpeter.github.io%2Fffit-lu08-club-accounting-
website%2Fstatus.json&query=message&label=%F0%9F%8C%90%20Frontend%20status&c
olor=color)
```

https://img.shields.io/badge/dynamic/json?url=https%3A%2F%2Falexanderpeter.github.io%2Fffit-lu08-club-accounting-website%2Fstatus.json&query=message&label=%F0%9F%8C%90%20Frontend%20status&color=color

https://img.shields.io/badge/dynamic/json?url=https%3A%2F%2Falexanderpeter.github.io%2Fffit-lu08-club-acccounting-website%2Fstatus.json&query=message&label=%F0%9F%8C%90%20Frontend%20status&color=color

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/ffit/3-jahr/cicd/learningunits/lu13/b?rev=1779787523

Last update: **2026/05/26 11:25**