Final Report - DevOps

EvilTwitter

Group E

abea, beba, gujo, luka, sena

IT University of Copenhagen
Denmark
26 - 4 - 2021

Contents

1 System's Perspective

1.1 Design of the system

1.2 Architecture of the system

1.3 Dependencies

-

1.4 Interactions of subsystems

1.5 Current state of the systems

- https://sonarcloud.io/dashboard?id=gustavjohansen98_E-vil-Corp

1.6 Software license agreement

2 Process' Perspective

Some more text

2.1 Development Strategy

TODO: fancy intro here

2.1.1 Tools used

The teams aimed at doing agile development via including practices like self organising teams and iterative delivery (TODO). This becomes more once the teams Kanban board is explained

For communication between the team members a Teams Group was used, which contained a general chat, a chat for arranging meetings and a chat that contains various useful links

To host the code a Github Repository was used, which was owned by one group member

Github Actions

Github Projects

Digital Ocean Droplet

Digital Ocean Database Cluster

2.2 Monitoring

2.2.1 Setup

The monitoring of the EvilTwitter application was done using the monitoring and alerting toolkit Prometheus[1] to gather information from the application. Two dotnet package was used to retrieve data from the application. Prometheus-net.SystemMetrics[2] was used to retreive system information from where the application was running, and prometheus-net[3] to gather information from the Controllers

Grafana[4] was used to interpret this data and gain valuable information

[1] https://prometheus.io/
[2] https://github.com/Daniel15/prometheus-net.SystemMetrics
[3] https://github.com/prometheus-net/prometheus-net
[4] https://grafana.com/

2.2.2 TODO: title

The information gathered in Grafana was setup in 3 categories: 1. Alerts, 2. Controller usage, 3. System.

Figure 1: Caption

First the alert graph can be seen at the top right, which is a value that can be either 1 or 0. This translates to what value latest returned last, with any latest value greater than 0 would resolved to a value of 1 and anything else would resolved to a value of 0. Hence if the Api is down or returns odd values this would be registered by Grafana

2.3 Logging

2.4 Security assessment

3 Lessons Learned Perspective

3.1 Evolution and refactoring

3.2 Operations

3.3 Maintenance

Even more text :O