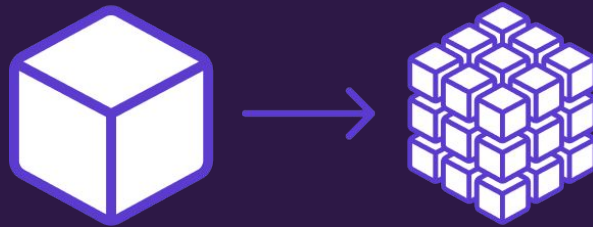


18.01.2020

CLC – Monolith to Microservices



Nicole Hölzl, Paul Hörmann, Fabian Kastner
DSE



Motivation



Motivation

- Aus Sicht eines Data Scientists einer der relevantesten Projektvorschläge
 - Für private Projekte, Transformieren von Data Science Monolith Projekten zu Microservice Anwendungen
- Bestehende Monolith Anwendung
 - Bereits bestehendes Privatprojekt konnte somit in Microservices aufgeteilt werden

Monolith

Monolith

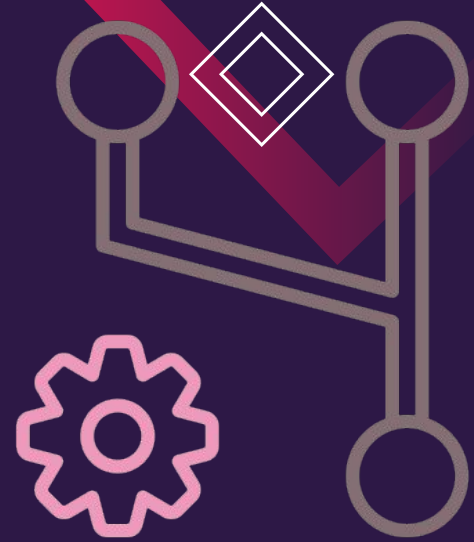
- Django (Python) Web Application
 - Visualisierung, Speicherung und Vorhersage von Aktienkursen
- Teile
 - Frontend (Web Server)
 - Datenbank (SQLite, MySQL)
 - Data Retriever (Python Script Subprocess)
 - Machine Learning (Python Script Subprocess)
 - Static, Local Config File




Versionierung/CI

Versionierung/CI

- Github
 - Main Repository
Beinhält die restlichen Repos als Submodules
- Github Actions:
 - Jeder Push auf den 'main' Branch triggert den CI Prozess
 - Die Action wird in den Submodules web, data, nn und config ausgeführt
 - Python Packages werden installiert und etwaige Tests ausgeführt, Ein Docker Image wird gebaut und auf DockerHub hochgeladen
Abschließend werden mittels kubectl die Deployments neu eingespielt




Django
By GitHub Actions



Build and Test a Django Project

Set up this workflow

 actions/starter-workflows

Python ●

Publish Docker Container
By GitHub Actions




Build, test and push Docker image to GitHub Packages.

Set up this workflow

 actions/starter-workflows


Dockerfile ●

Python application
By GitHub Actions



Create and test a Python application.

Set up this workflow

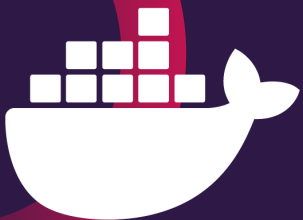
 actions/starter-workflows

Python ●



Development





Development



- Identifizieren der einzelnen Microservices
- Auskoppelung aus der Monolith Application
- Einrichten der Kommunikation (Datenbank Authentifikation, HTTP Requests, Exposing von Ports)
- Erstellen von Dockerfiles zur Ausführung der Microservices in Containern
- Docker-compose zum Erstellen und Ausführen der Einzelnen Dockerfiles, Einrichten der Netzwerke und Volumes



Microservices



Architektur

- Databases
 - 2 VM's; Services wären eine Option gewesen, aber VM's sind billiger.
- Kubernetes
 - Azure:
 - Web-deployment -> Django App
 - Config-deployment -> Config Service
 - Data-deployment -> Data Update Mechanism
 - Local:
 - Wie in in Azure plus:
 - Data-deployment -> zusätzlicher Container mit DB
 - Django DB-deployment -> DB für django



Azure



Resources



- VM mit PostgreSQL
- VM mit MySQL
- Azure Kubernetes Service
- Public static IP
- + dadurch automatisch generierte Ressourcen

Azure Dashboard



Application

