HOCHSCHULE KONSTANZ TECHNIK, WIRTSCHAFT UND GESTALTUNG
UNIVERSITY OF APPLIED SCIENCES

CAD-Project - Mom based Information Live Flow

Paul Drautzburg, Lukas Hansen, Georg Mohr, Kim De Souza, Sebastian Thuemmel, Sascha Drobig

Vorwort

Das vorliegende Dokument beschreibt die Umsetzung für das Projekt im Rahmen des MSI-Kurses *Cloud Application Development*.

Inhaltsverzeichnis

Abbildungsverzeichnis **iv**

Tabellenverzeichnis **v**

1 Motivation **1**
1.1 Zielsetzung 1
1.2 Die 12 Faktor-APP Anforderungen 2

2 Einleitung **2**

3 Verbindung der Systeme (MOM) **2**
3.1 RabbitMQ 2

4 Datenquelle (Wetter-API) **3**

5 Datenverarbeitung **4**
5.1 CEP 4
5.2 Datenbank 4

6 Anwendersicht **4**
6.1 Andriod-APP 4
6.2 Web-Client 4

7 Deployment **4**
7.1 Allgemein 4
7.2 Cloudfoundry 4
7.3 REDIS 4
7.4 AWS 4

8 Kostenmodell **4**

Abbildungsverzeichnis

Tabellenverzeichnis

1 12 Faktor App Anforderungen 1
2 Validierung nach "12 Faktor APP" 3

1 Motivation

1.1 Zielsetzung

ToDo: Verweis auf 12 Faktor APP Standard !!! Die Tabelle soll am Anfang stehen, am besten in der Zielsetzung. Die folgende Tabelle beschreibt die Kernanforderungen der 12 Faktor APP,

12 Faktor APP Anforderungen		
ID	**Anforderung**	**Beschreibung**
1.	Codebase	Eine im Versionsmanagementsystem verwaltete Codebase, viele Deployments.
2.	Abhängigkeiten	Abhängigkeiten explizit deklarieren und isolieren.
3.	Konfiguration	Die Konfiguration in Umgebungsvariablen ablegen.
4.	Unterstützende Dienste	Unterstützende Dienste als angehängte Ressourcen behandeln.
5.	Build, release, run	Build- und Run-Phase strikt trennen.
6.	Prozesse	Die App als einen oder mehrere Prozesse ausführen.
7.	Bindung an Ports	Dienste durch das Binden von Ports exportieren.
8.	Nebenläufigkeit	Mit dem Prozess-Modell skalieren.
9.	Einweggebrauch	Robuster mit schnellem Start und problemlosen Stopp.
10.	Dev-Prod-Vergleichbarkeit	Entwicklung, Staging und Produktion so ähnlich wie möglich halten.
11.	Logs	Logs als Strom von Ereignissen behandeln.
12.	Admin-Prozesse	Admin/Management-Aufgaben als einmalige Vorgänge behandeln.

Tab. 1: 12 Faktor App Anforderungen

1.2 Die 12 Faktor-APP Anforderungen

2 Einleitung

3 Verbindung der System

Die entwickelte Anwendung besteht mit einem Java-Service für die verwendete Wetter-API, der Complex Event Processing Engine und den Anwender-Clients aus drei Komponenten. Diese Komponenten müssen möglichst stark entkoppelt miteinander kommunizieren können. Durch eine starke Entkopplung wird erreicht ,dass die jeweiligen Komponenten keine Kenntnisse über vorhandene Schnittstellen oder die verwendete Programmiersprache besitzen müssen. Um dies zu realisieren, wird RabbitMQ als Message-oriented Middleware (MoM) eingesetzt.

3.1 RabbitMQ

Bei RabbitMQ handelt es sich um einen OpenSource Message Broker welcher unter anderem Bibliotheken für Java, JavaScript, Swift und C# anbietet, wodurch die Kommunikation mit Android-, iOS- und Webapplikationen möglich wird. Durch die Verwendung von Queues und Topics wird die asynchrone Verteilung der Nachrichten ermöglicht. RabbitMQ verwendet als Standard das Messaging Protokoll AMQP, bietet aber Plugins für alternative Protokolle wie MQTT und STOMP. Da auch mobile Geräte zu den eingesetzten Komponenten gehören, wird das Protokoll MQTT eingesetzt, da dieses speziell für den Einsatz in Mobilgeräten entwickelt wurde. Die Kommunikation der einzelnen Komponenten erfolgt mit MQTT über Topics. Damit der Nachrichtenaustausch stattfinden kann, müssen sich die miteinander kommunizierenden Komponenten auf ein oder mehrere gemeinsame Topics einigen. Der Aufbau eines Topics ist mit REST-Schnittstellen vergleichbar und kann aus mehreren Topic-Leveln bestehen. Zusätzlich können beim Abonnement von Topics Platzhalter wie + und # eingesetzt werden. Diese funktionieren wie reguläre Ausdrücke und ersetzen im Falle des Platzhalters + eine einzelne Topic-Ebene und beim Platzhalter # alle nachfolgenden Ebenen. Die Topics dieser Anwendung sehen wie folgt aus:

78467/today
: Abonnement des Wetters von Postleitzahl 78467 des heutigen Tages

+/today
: Abonnement des Wetters aller verfügbaren Postleitzahlen des heutigen Tages

78467/today/alert
: Abonnement der Wetterwarnungen für die Postleitzahl 78467

+/weekly
: Abonnement der Vorhersage der nächsten Woche aller verfügbaren Postleitzahlen

#
: Abonnement aller verfügbaren Topics

Damit ein Client Daten senden und empfangen kann, muss er sich beim Verbindungsaufbau authentifizieren und für den Zugriff auf das entsprechende Topic autorisiert sein. Die Authentifizierung erfolgt über eine gewöhnliche Benutzername / Passwort - Abfrage. Um den Zugriff auf MQTT-Topics zu beschränken, ermöglicht RabbitMQ die Verwendung virtueller Hosts (vHosts). Durch diese erlangen nur die Nutzer Zugriff auf ein Topic, wenn sie für den vHost des Publishers autorisiert sind. Durch dieses Verfahren kann das gleiche Topic von mehreren Nutzern mit unterschiedlichen vHosts verwendet werden, ohne das sie die Nachrichten anderer vHosts des gleichen Topics lesen können. Auf diese Weise erfüllt die Anwendung die Anforderung der Multi-Tenancy.

4 Datenquelle (Wetter-API)

Validierung nach ”12 Faktor APP”			
ID	**Anforderung**	**Validierungs Element**	**Erfüllt**
1.	Codebase	....	Nein
2.	Abhängigkeiten	....	Nein
3.	Konfiguration	....	Nein
4.	Unterstützende Dienste	....	Nein
5.	Build, release, run	....	Nein
6.	Prozesse	....	Nein
7.	Bindung an Ports	....	Nein
8.	Nebenläufigkeit	....	Nein
9.	Einweggebrauch	....	Nein
10.	Dev-Prod-Vergleichbarkeit	....	Nein
11.	Logs	....	Nein
12.	Admin-Prozesse	....	Nein

Tab. 2: Validierung nach ”12 Faktor APP”

ToDo: Verweis auf 12 Faktor APP Standard !!!

5 Datenverarbeitung

5.1 CEP

5.2 Datenbank

6 Anwendersicht

6.1 Andriod-APP

6.2 Web-Client

7 Deployment

7.1 Allgemein

7.2 Cloudfoundry

7.3 REDIS

7.4 AWS

8 Kostenmodell