Anforderungsspezifikation

Silvan Adrian
Fabian Binna

1 Änderungshistorie

Datum	Version	Änderung	Autor
02.10.15	1.00	Erstellung des Dokuments	Gruppe
02.10.15	1.01	Nicht funktionale Anforderungen	Silvan Adrian
02.10.15	1.02	Use Cases Aktoren + User Stories Aktoren	Silvan Adrian
03.10.15	1.03	Anforderungen API	Fabian Binna
03.10.15	1.04	Anforderungen Dashboards + Mockups eingefügt	Silvan Adrian

Inhaltsverzeichnis

2 Einführung

2.1 Zweck

Dieses Dokument beinhaltet die Anforderung zur Analyse.

2.2 Gültigkeitsbereich

Dieses Dokument ist während des ganzen Projekts gültig.

2.3 Referenzen

-

3 Anforderungen

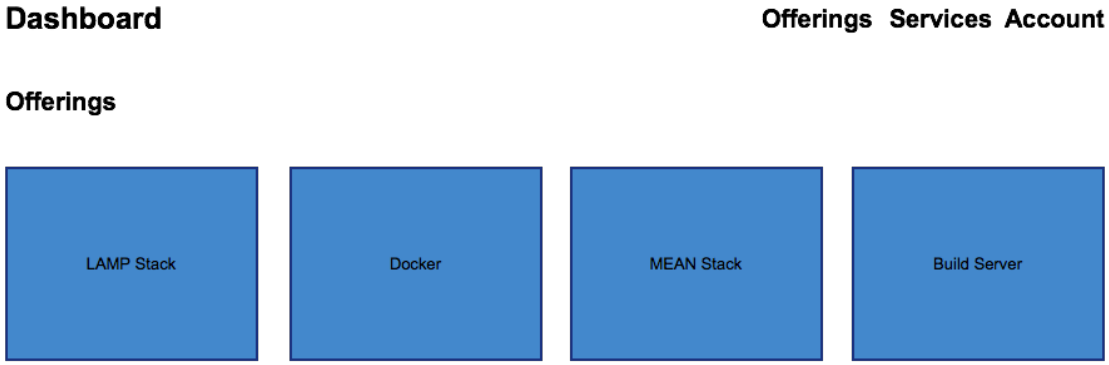
3.1 API

Die API definiert einen Workflow der einen Service auf einer Cloud erstellt. Es ist offen, ob dieser Service über mehrere Cloud Anbieter hinaus geht. Der Service wird durch ein Konfigurationsfile (z.B. json) definiert. Zusätzlich können Scriptfiles referenziert werden, die die Software auf den Instanzen installieren. Ein Service kann auch wieder gelöscht werden. Es ist nicht die Aufgabe der API existierende Services zu identifizieren. Die API muss Modular sein, das heisst es sollte möglich sein andere oder eigene Programme für die Cloud Kommunikation zu verwenden. Innerhalb der API werden Compute, Storage, Network usw. als ServiceModule bezeichnet. Diese Abstraktion ermöglicht das wiederverwerten und erweitern der API.

3.2 Customer-Dashboard

3.2.1 Homescreen

Im Homescreen (sobald der Customer auf das Dashboard zugreift) werden alle zu Verfügung stehenden Services angezeigt. Hier werden die Services Offerings genannt um eine Unterscheidung zwischen Abonnierten Services (Services) und zur Verfügung stehenden Services (Offerings) machen zu können.



3.2.2 Services Übersicht

In der Services Übersicht werden dem Customer alle abonnierten Services angezeigt und können hier auch gekündigt werden.

3.2.3 Service abonnieren

Sobald ein Service auf dem Homescreen ausgewählt wird muss noch der zuständige Provider gewählt werden (häng auch wieder davon ab von welchem Provider bisher Logindaten hinterlegt wurden). Da momentan noch kein Hybrid Betrieb vorgesehen ist muss ein Account gewählt werden unter welchem der Service verrechnet wird.

3.2.4 Cloud Credentials

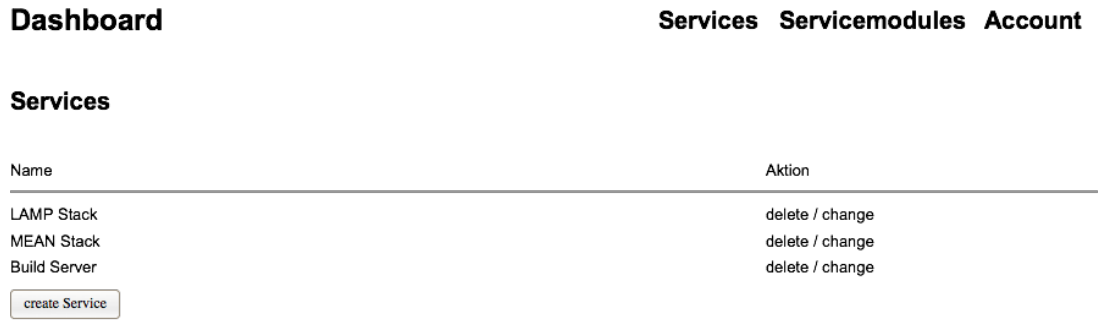
Da für jeden Provider wieder Logindaten benötigt werden müssen diese an einer zentralen Stelle gespeichert werden (anhand von denen kann sich auch das Service Angebot ändern).

3.3 Admin-Dashboard

Zusätzlich zum Customer-Dashboard soll ein Admin-Dashboard zur Verfügung stellen in welchem der Admin Services und Servicemodule erstellen kann.

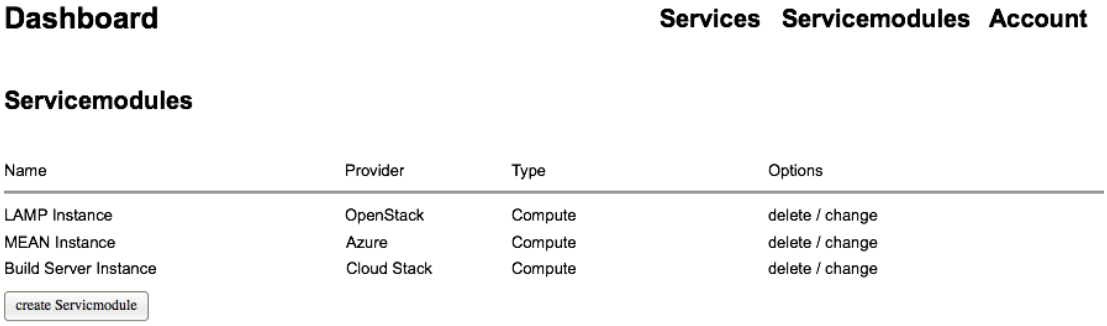
3.3.1 Service

Ein Service hat einen bestimmten Namen und jedem Service sind eine gewisse Anzahl Servicemodule zugeteilt. um den Service abbilden zu können. Hier kann der Admin den Service ändern und je nach Anforderung den Service anpassen.



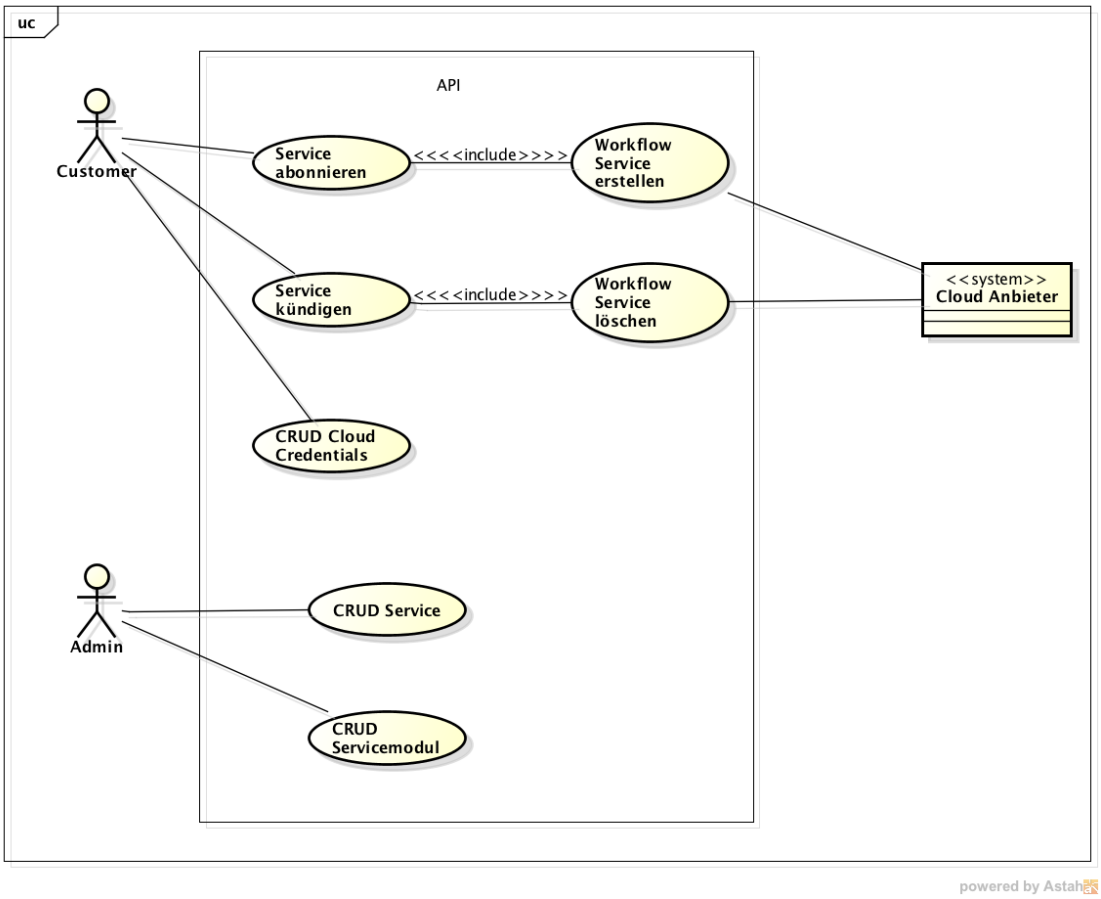
3.3.2 Servicemodul

Jedes Servicemodul besitzt einen Namen und wird einem Provider zugeschrieben, dabei kann jedes Servicemodule den Typ Compute,Network oder Storage haben.



4 Use Cases

4.1 Use Case Diagramm



4.2 Aktoren & Stakeholders

4.2.1 Customer

Als Customer möchte ich meine abonnierten Services verwalten.

Aktor	Typ	Ziele
Customer	Primary	• Service abonnieren • Service kündigen

4.2.2 Admin

Als Admin möchte ich Services und Servicemodule verwalten können.

Aktor	Typ	Ziele
Admin	Primary	• Service erstellen • Service anpassen • Service löschen • Servicemodul erstellen • Servicemodul anpassen • Servicemodul löschen

4.3 Beschreibungen fully dressed

4.3.1 Service abonnieren

Primäraktor	Customer
Steakholders und Interessen	Customer: Möchte einen Service abonnieren
Vorbedingungen	Das Customer-Dashboard wurde geöffnet
Nachbedingungen	Das Customer-Dashboard wurde geschlossen
Standartablauf	1. Der Customer gibt die Webadresse für das Dashboard ein
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

4.3.2 Service kündigen

Primäraktor	Customer

Steakholders und Interessen	Customer: Möchte einen Service kündigen
Vorbedingungen	Das Customer-Dashboard wurde geöffnet
Nachbedingungen	Das Customer-Dashboard wurde geschlossen
Standartablauf	1. Der Customer gibt die Webadresse für das Dashboard ein
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

4.3.3 Services verwalten

Primäraktor	Customer
Steakholders und Interessen	Admin: Möchte einen Service verwalten
Vorbedingungen	Das Admin-Dashboard wurde geöffnet
Nachbedingungen	Das Admin-Dashboard wurde geschlossen
Standartablauf	1. Der Admin gibt die Webadresse für das Admin-Dashboard ein
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

4.3.4 Servicemodule verwalten

Primäraktor	Customer
Steakholders und Interessen	Admin: Möchte ein Servicemodul verwalten
Vorbedingungen	Das Admin-Dashboard wurde geöffnet
Nachbedingungen	Das Admin-Dashboard wurde geschlossen
Standartablauf	1. Der Customer gibt die Webadresse für das Dashboard ein
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

5 User Stories

5.1 Rollen

5.1.1 User

Als User benutze ich das Dashboard, um mir einen Service zu abonnieren und

5.1.2 Admin

Als Admin benutze ich die API über die Kommandozeile oder nutze das Admin-Dashboard um neue Services zusammenzustellen.

6 Nichtfunktionale Anforderungen

6.1 Menge

- Die Software unterstützt mehr als 30 Cloud Anbieter (libcloud)
- Bei jedem Cloud Anbieter bestehen eine gewisse Anzahl Services (von Anbieter zu Anbieter verschieden)

6.2 Schnittstellen

- Die Software wird über HTTP/HTTPS angesprochen

- Zur Interaktion im Admin-Dashboard werden die herkömmlichen Schnittstellen gebraucht (Maus,Tastatur,Bildschirm)
- Interaktionen können auch über die Kommandozeile ausgeführt werden

6.3 Qualitätsmerkmale

6.3.1 Funktionalität

siehe Abschnitt API und Dashboard

6.3.2 Zuverlässigkeit

- Der Workflow zum erstellen eines Services soll entweder durchgeführt und abgeschlossen werden oder falls Unterbruch/Fehler rückgängig gemacht werden.
- Die Software soll verteilt betrieben werden und eine möglichst hohe Verfügbarkeit bieten

6.3.3 Benutzerbarkeit

- Die Software kann über das vorgesehene Admin-Dashboard benutzt werden
- Die API kann auch über die Kommandozeile angesprochen werden

6.3.4 Effizienz

-

6.3.5 Änderbarkeit

Die Software soll modular aufgebaut werden, damit Erweiterungen in Zukunft problemlos möglich sind.

6.3.6 Übertragbarkeit

Das Projekt wird in Python geschrieben ist somit also auf Python mindestens in der Version 2.5 angewiesen, kann allerdings durch den Einsatz eines Docker Containers einfach Übertragbar gemacht werden.