ImkerCloud

Archtekturdokument

Inhalt

[1. Einführung 3](#_Toc72354421)

[2. Architektur 3](#_Toc72354422)

[2.1 Back-End 4](#_Toc72354423)

[2.1.1 Entscheidungsfindung basierend auf DMN 5](#_Toc72354424)

[2.2 Front-End 6](#_Toc72354425)

[3. Models at Runtime 7](#_Toc72354426)

# Einführung

Wir leben in einer Welt, in der die Informatik das Ökosystem verschiedener Bereiche bereichert hat, sei es Marketing, menschliche Beziehungen und sogar Landwirtschaft und Imkerei. Es wurde eine Frage gestellt: Wie können wir den Umgang mit einem Bienenstock verbessern?

Die Antwort darauf war offensichtlich Informatik. Die Anwendung "Imker Cloud" zielt darauf ab, Smart-Bienenstöcke mit einer Anwendung zu verknüpfen, die Sensoren verwalten, Daten sammeln und Entscheidungen auf der Grundlage verschiedener Faktoren treffen kann. Wir können die Zeit vergessen, in der Imker nacheinander Hunderte von Bienenstöcken überprüfen mussten. Mit dieser Anwendung sind alle Informationen nur einen Klick entfernt.

# Architektur

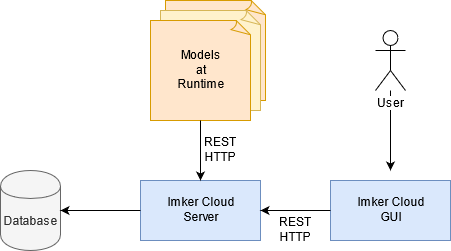


Abbildung 1 - ImkerCloud Komponentendiagramm

Der erste Schritt bei der Gestaltung der Architektur war die Entscheidung, welche der Komponenten die "Hauptfigur" sein wird. Zwischen den "Modellen zur Laufzeit" und dem Server haben wir beschlossen, dass die Modelle bestimmen, was der Server tut. Wir werden in einem anderen Kapitel über die Modelle sprechen. Im Moment konzentrieren wir uns auf die Backend-Ebene.

Die Anwendung "ImkerCloud" ist in drei Hauptebenen unterteilt:

1. Models At Runtime
2. Back-End
3. Client

In den folgenden Unterkapiteln werden die einzelnen Ebenen ausführlicher beschrieben.

## 2.1 Back-End

Die Backend-Komponente kapselt den Server und eine Datenbank. Die Datenbank speichert Informationen über Bienenstöcke, Benutzer, die die Anwendung verwenden, welcher Benutzer für welchen Bienenstock verantwortlich ist, sowie Benachrichtigungen, die Benutzer möglicherweise erhalten können.

Der Server ist als SpringBoot-Anwendung in der Programmiersprache Java implementiert. Unsere Entscheidung für Java und SpringBoot wurde durch die Tatsache getroffen, dass SpringBoot ein großartiger Bootstrap für kleine bis mittlere Projekte ist, der auch viele Funktionen und Abhängigkeiten enthält, die die Entwicklung der Anwendung schneller und einfacher machen.

Der Server übernimmt erneut seine Rolle als "Gehirn" der Anwendung. Es bietet Methoden zum Bearbeiten von Daten und ermöglicht es Clients, Informationen aus der Datenbank anzufordern. Der Server soll Funktionen für die folgenden Komponenten bereitstellen:

1. **Bienenstöcke**. Sensoren lösen REST-APIs aus, die der Server anbietet. Der Server bestätigt dann die Anforderung und beginnt mit der Verarbeitung. Wenn der Sensor eine Temperaturhöhe signalisiert, kühlt er den Bienenstock ab und aktualisiert den Server über den Temperaturzustand. Der Server aktualisiert die Informationen entsprechend. Ein weiterer Anwendungsfall wäre der Bienenstockverkehr. Für einen bestimmten Bienenstock sendet der Sensor Anforderungen an den Server, ob eine Biene den Bienenstock verlassen oder ihn betreten hat. Der Server aktualisiert die aktuelle Bevölkerungszahl entsprechend. Ein Sensor kann auch eine Anfrage an den Server senden, die Informationen über die aktuelle Aktivität des Bienenstocks enthält. Ist der Bienenstock hyperaktiv, speichert der Server diese Informationen und benachrichtigt den Imker, der für den Bienenstock verantwortlich ist.
2. **Benachrichtigungen**. Der Server sendet verschiedene Arten von Benachrichtigungen an Benutzer. Diese Benachrichtigungen werden in der Datenbank gespeichert. Der Benutzer kann dann anfordern, seine Benachrichtigungen zu sehen, wann immer er möchte. Im Falle einer Gefahr, z. B. wenn der Bienenstock kritische Temperatur- oder Aktivitätsniveaus erreicht, sendet der Server eine Notfall-E-Mail.
3. **Benutzer** können in der ImkerCloud-Datenbank gespeichert werden. Sie können sich dann mit ihren Anmeldeinformationen anmelden, um den Status der Bienenstöcke und ihre Benachrichtigungen zu überprüfen.

### 2.1.1 Entscheidungsfindung basierend auf DMN

Da sich die Bedingungen häufig ändern können, haben wir beschlossen, DMN zum Server hinzuzufügen. Der Server lädt die DMN-Tabelle in den Speicher und trifft darauf basierend Entscheidungen.

Nach einigen Recherchen haben wir herausgefunden, dass Camunda eine Maven-Abhängigkeit bietet, die eine Entscheidungstabelle in den Speicher laden und dann darauf hinweisen kann, was zu tun ist.

n/A

## Front-End

Die Front-End-Komponente ist hauptsächlich eine grafische Benutzeroberfläche, über die sich der Benutzer anmelden und verschiedene Daten visualisieren kann. Um diese Komponente zu entwickeln, haben wir Angular verwendet, das auf NodeJs ausgeführt wird. Unten finden Sie einige Screenshots, wie die GUI aussieht.

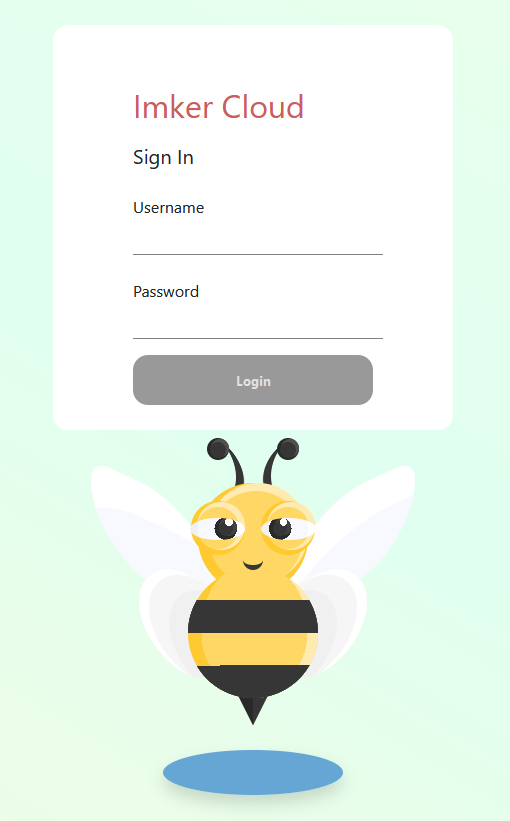


Abbildung 2 - Imker Cloud - GUI – Einloggenseite

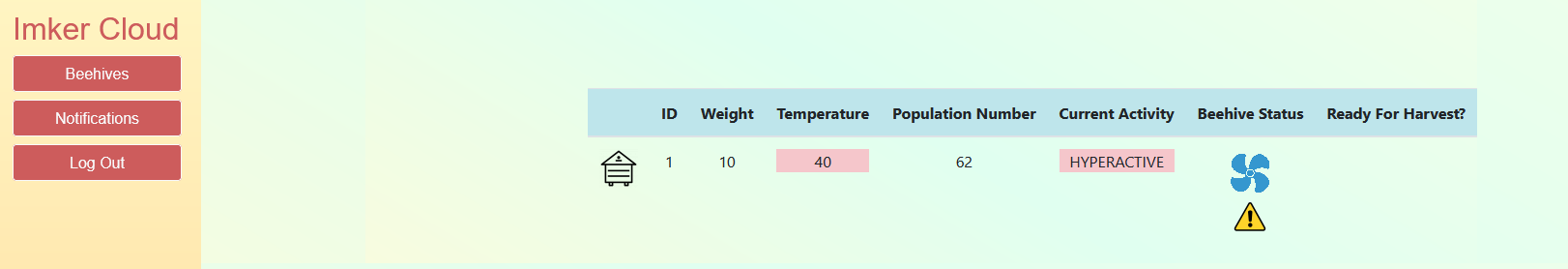


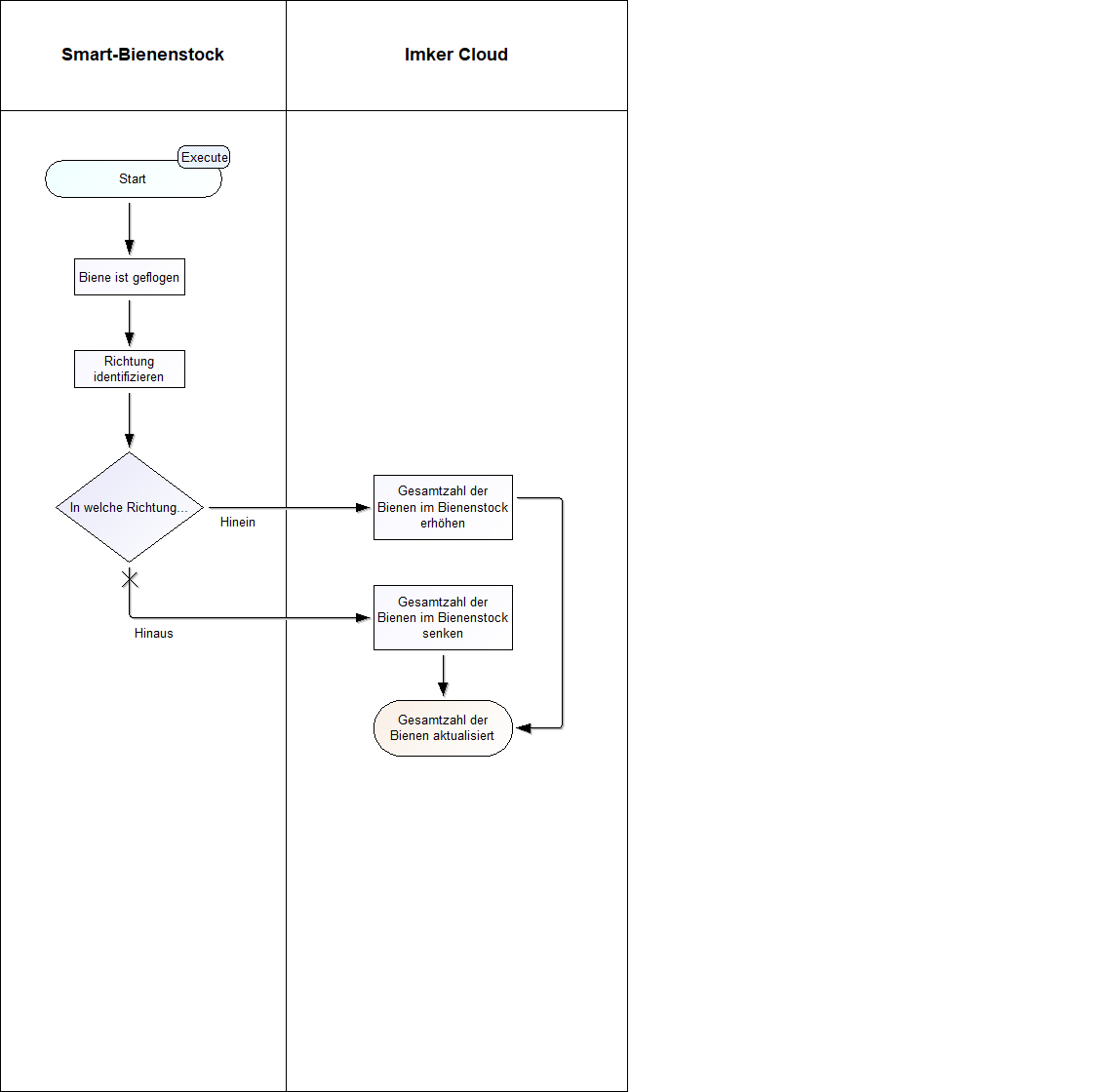
Abbildung 3 - ImkerCloud - GUI – Bienenstöckeübersichtsseite

n/A Abbildung 4 – ImkerCloud – GUI Benachrichtigungsübersichtsseite

# Models at Runtime

Flussdiagramme und DMN-Tabellen spielen in dieser Anwendung die Hauptrolle. Die Flussdiagramme fungieren als Bienenstocksensoren, während die DMN-Tabelle unterschiedliche Bedingungen speichert und angibt, was zu tun ist, wenn sie erfüllt sind.

Wir beginnen dieses Kapitel mit einer Erklärung, was genau die Flussdiagramme tun. Unten finden Sie ein Beispiel für das Flussdiagramm "Aktivitätskontrollsensor ausgelöst".



Wir können zwei Lanes sehen: den Smart-Bienenstock und die Imker Cloud. Das Flussdiagramm beginnt mit der Operation "Biene ist geflogen". Dies "löst" den Sensor aus. Wenn Sie das Flussdiagramm ausführen, wird eine Bearbeitungsbox angezeigt, in der Sie gefragt werden, in welchem Bienenstock dies geschehen ist.

Die zweite Operation lautet "Richtung identifizieren". Hier verarbeitet der Sensor die Richtung. Hier erscheint auch ein Bearbeitungsfeld, in dem der Benutzer nach seiner Eingabe gefragt wird. Es kann "Hinein" oder "Hinaus" sein.

Basierend auf den Eingaben des Benutzers sendet der "Sensor" dann zwei verschiedene Anforderungen an den Server. Wenn die Biene außerhalb des Bienenstocks geflogen ist, teilt der Sensor dem Server mit, dass eine Biene abgereist ist und dass sie von der allgemeinen Populationszahl abgezogen werden sollte. Wenn die Biene hineingeflogen ist, weist der Sensor den Server an, die allgemeine Bevölkerungszahl zu erhöhen.