

Hochschule Karlsruhe  
Technik und Wirtschaft  
UNIVERSITY OF APPLIED SCIENCES

DOKUMENTATION

# Verteilte Systeme Master Labor

*Pol Zeimet (65834)*  
*zepo1012@hs-karlsruhe.de*

*Yannick Stephan (65934)*  
*stya1012@hs-karlsruhe.de*

Aufgabe 2

20. November 2019

# 1 Architekturentwurf

Auf Basis der Analyse aus Aufgabe 1 wurde eine Microservice Architektur entworfen, welche in Abbildung 1 zu sehen ist. Diese Architektur beinhaltet eine Menge von Core-, Composite- und API-Services. Eine genauere Beschreibung der einzelnen Services sind in den folgenden Unterkapiteln zu entnehmen. In Kapitel 2 sind die konkreten REST-API Schnittstellen der einzelnen Services aufgezeigt.

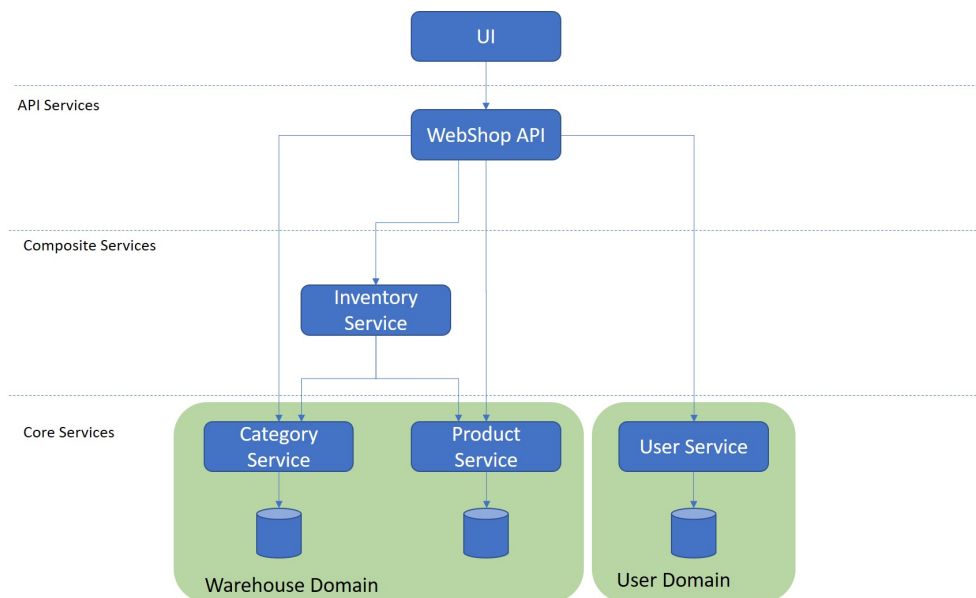


Abbildung 1: Entwurf einer Microservice-basierten Zielarchitektur

## 1.1 API Services

### 1.1.1 WebShop API

Dieser API-Service dient als Schnittstelle zwischen der Frontend-Anwendung und den restlichen Microservices, die von außen nicht sichtbar sein sollen.

## 1.2 Composite Service

### 1.2.1 Inventory Service

Der Inventory Service ist für alle Funktionen zuständig, bei denen es Abhängigkeiten zwischen Produkten und Kategorien gibt.

## **1.3 Core Services**

### **1.3.1 Category Service**

Der Category Service erstellt, löscht und listet Kategorien der WebShop-Anwendung.

### **1.3.2 Product Service**

Der Product Service ist für das Listen, Erstellen und Suchen von Produkten der Anwendung zuständig.

### **1.3.3 User Service**

Der User Service ist zuständig für die Benutzerverwaltung der Anwendung. Hier können neue Benutzer angelegt und ausgegeben werden. Zudem können sich Rollen anhand vom Benutzerlevel zurückgegeben werden lassen.

## 2 REST-API

Alle yaml-Dateien für die folgenden REST-Schnittstellen befinden sich im Anhang.

### 2.1 WebShop API

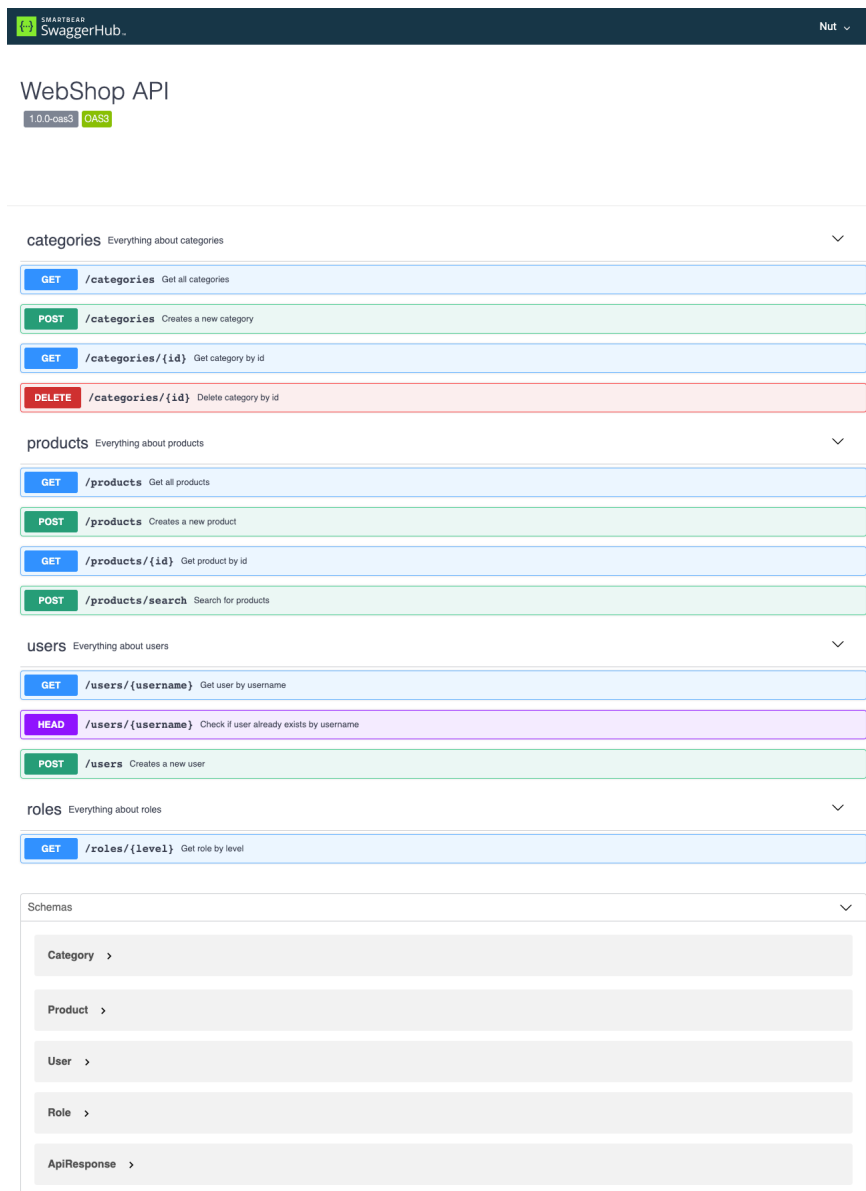


Abbildung 2: WebShop REST-API

## 2.2 Inventory Service API

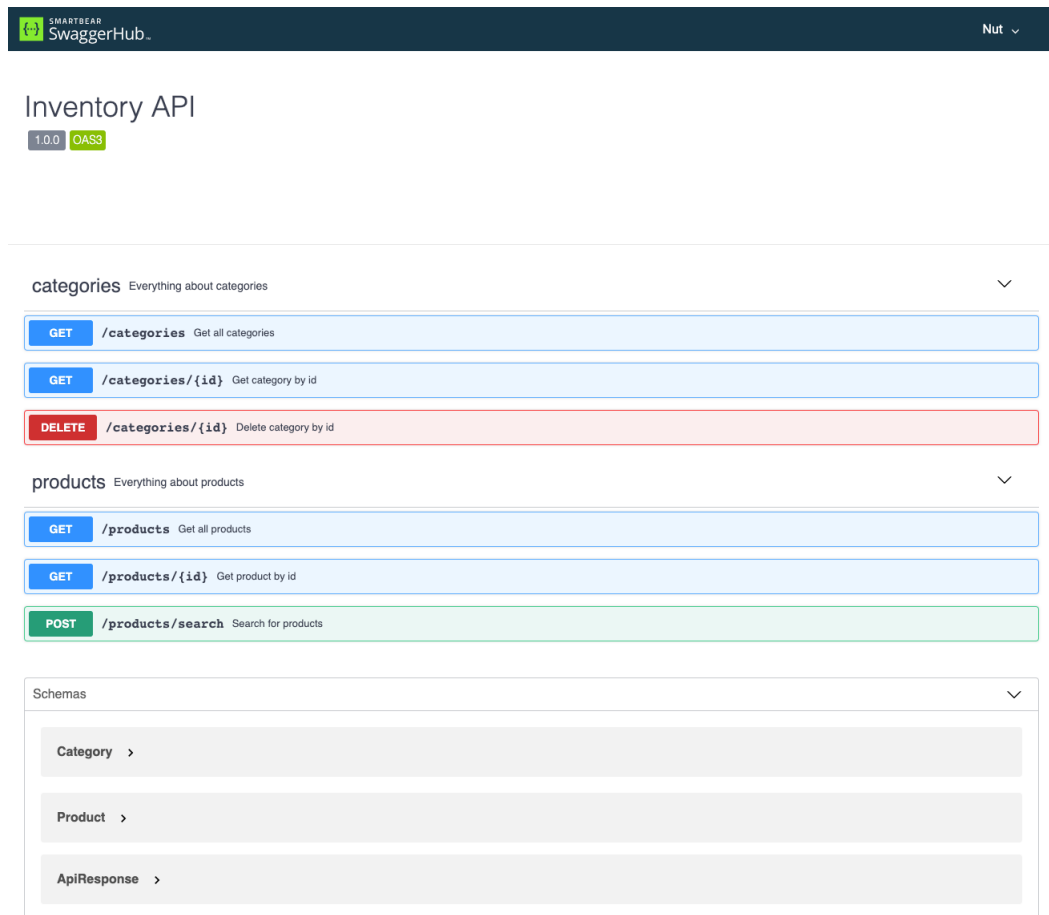


Abbildung 3: Inventory Service REST-API

## 2.3 Category Service API

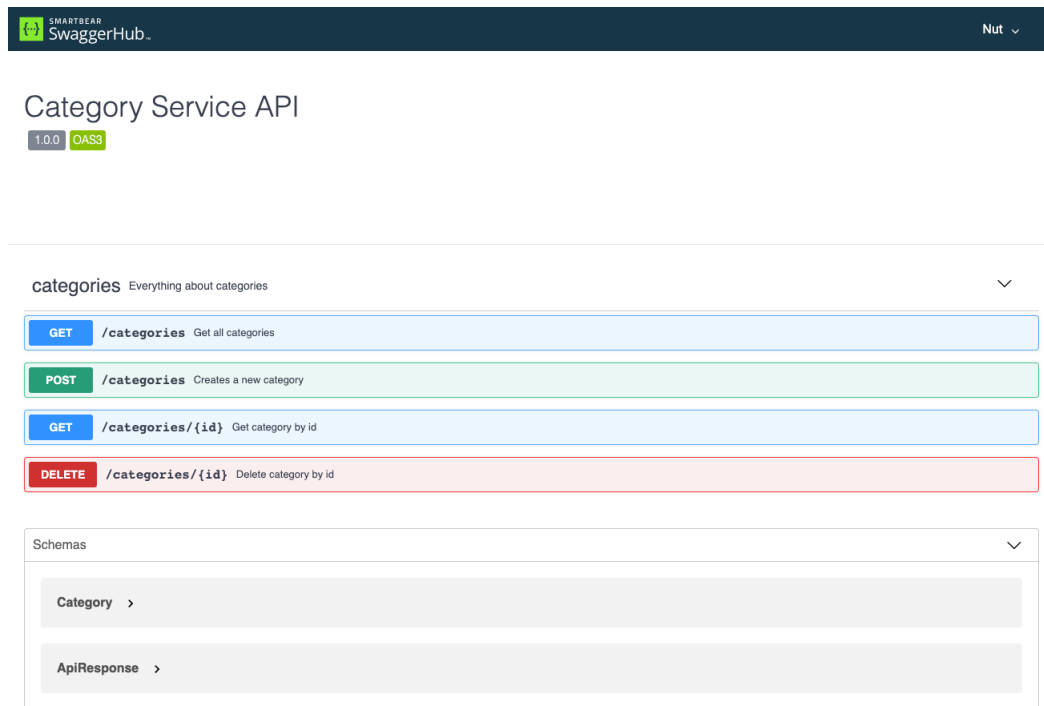


Abbildung 4: Category Service REST-API

## 2.4 Product Service API

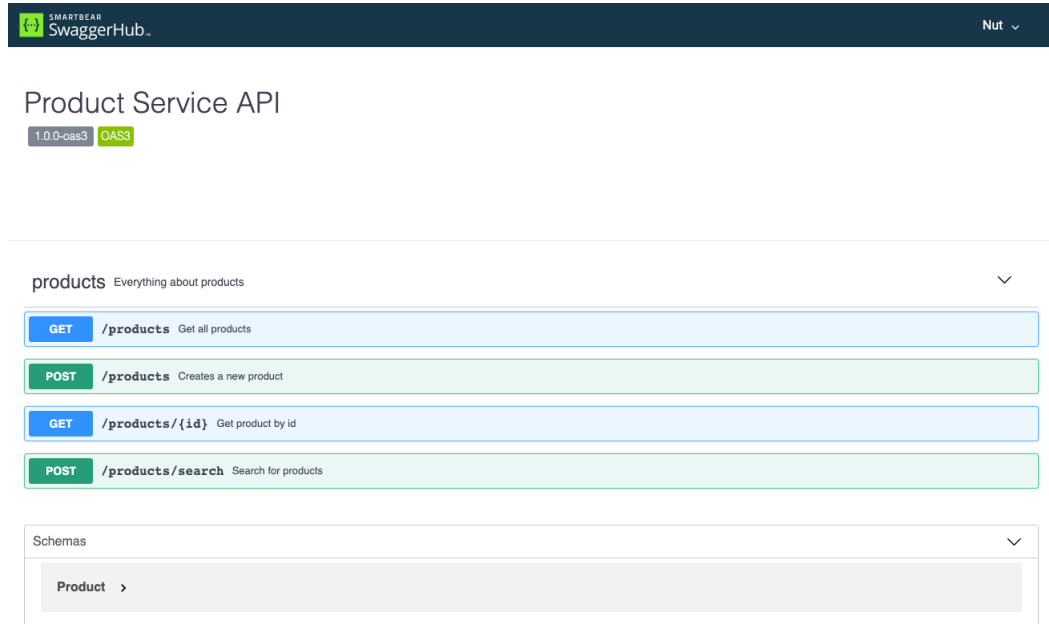
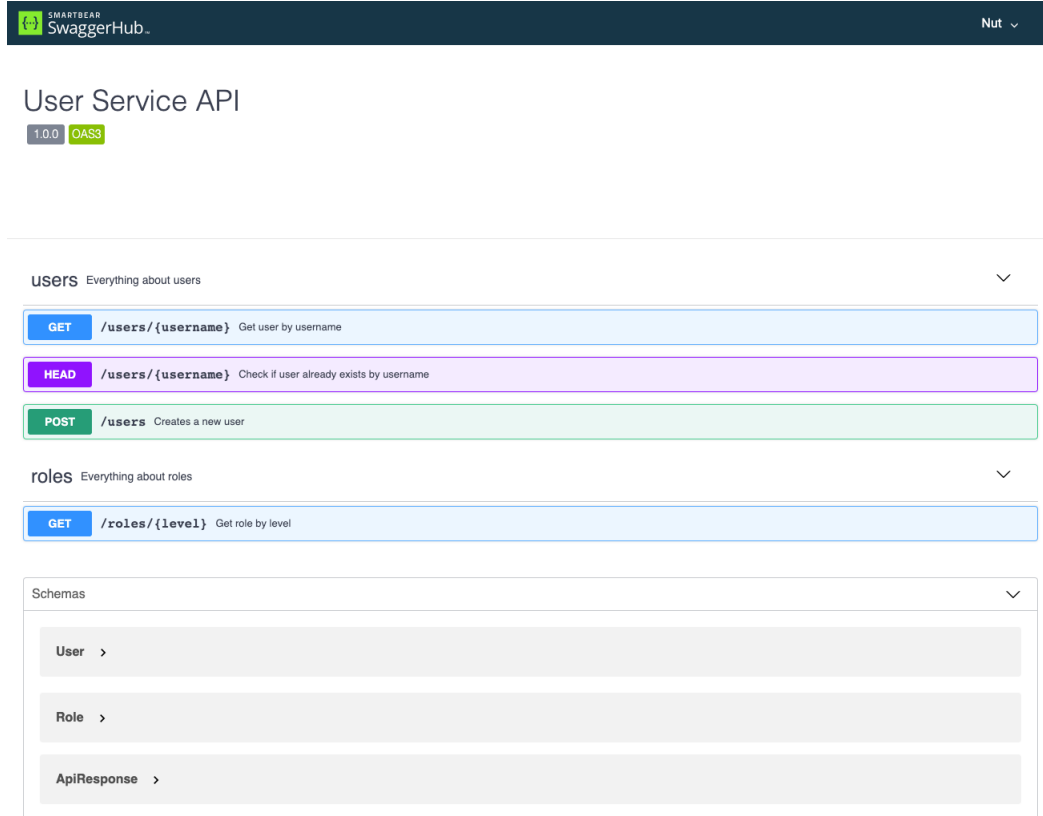


Abbildung 5: Product Service REST-API

## 2.5 User Service API



The image shows the SwaggerHub interface for the 'User Service API'. At the top, there is a dark blue header with the 'SMARTBEAR SwaggerHub' logo on the left and a 'Nut' dropdown menu on the right. Below the header, the title 'User Service API' is displayed, followed by a version indicator '1.0.0' and the 'OAS3' specification version. The main content area is divided into three sections: 'users', 'roles', and 'Schemas'. The 'users' section is titled 'users Everything about users' and contains three endpoints: a GET endpoint for '/users/{username}' (Get user by username), a HEAD endpoint for '/users/{username}' (Check if user already exists by username), and a POST endpoint for '/users' (Creates a new user). The 'roles' section is titled 'roles Everything about roles' and contains one GET endpoint for '/roles/{level}' (Get role by level). The 'Schemas' section is titled 'Schemas' and contains three expandable schema entries: 'User', 'Role', and 'ApiResponse', each with a right-pointing chevron icon.

SMARTBEAR  
SwaggerHub

Nut

### User Service API

1.0.0 OAS3

**users** Everything about users

**GET** /users/{username} Get user by username

**HEAD** /users/{username} Check if user already exists by username

**POST** /users Creates a new user

**roles** Everything about roles

**GET** /roles/{level} Get role by level

**Schemas**

**User** >

**Role** >

**ApiResponse** >

Abbildung 6: User Service REST-API