

**Philosophische** Fakultät III

Sprach- , Literatur- und Kulturwissenschaften

Institut für Information und Medien, Sprache und Kultur (I:IMSK)  
Lehrstuhl für Medieninformatik

Seminar KI für Serious Games

Modul: INF-M07.2

Leitung: Professor Bernd Ludwig

Abgabedatum: 31.03.2018

**„How To“ – Handbuch**

David Halbhuber 1744590

Jakob Fehle 1770881

[1 Motivation 3](#_Toc510232152)

[2 Verwendung Schnittstelle zur Nährstoffberechnung (API) 3](#_Toc510232153)

[2.1 Installation Flask 3](#_Toc510232154)

[3 Verwendung des Frontend/Datenbanksuche 4](#_Toc510232155)

# Motivation

Das vorliegende Dokument dient als Handbuch für die im Rahmen des Seminars „KI for serious games“ entstanden Software. Um die Strukturiertheit der Seminararbeit möglichst hoch zu halten, und um ein Art Quickstart in die von uns entwickelten Anwendungen zu ermöglichen, haben wir uns entschlossen diese zusätzliche Anleitung anzufertig. Außerdem soll dieses Dokument erläutern wie die entwickelten Dienste im Falle eines Ausfalls wieder gestartet werden können.

# Verwendung Schnittstelle zur Nährstoffberechnung (API)

Die entwickelte Schnittstelle, die es anderen Benutzern und Entwickler gestatten soll selbst die Nährwerte von neuen Gerichten und Rezepten auszuwerten, welche nicht im ursprünglich von Hr.Ullmann vorgesehenen. raw Format vorliegen. Dabei dient die Python Bibliothek Flask als Webframework und build-in Webserver. Da zum Zeitpunkt der Abgabe der vorliegenden Arbeit Flask nicht auf dem Universitätsserver bereitgestellt werden konnte wird im folgenden kurz dessen Installation erläutert und anschließen geschildert wie die Webapplikation schlussendlich gestartet werden kann.

## Installation Flask

Flask kann mit Hilfe von Python Paketmanagers „pip“ installiert werden, dazu genüg folgender Linux Befehl:

* sudo python pip install flask

Nach der Installation von Flask kann der Webserver durch folgende Linux Befehl gestartet werden:

* Wechsel in korrektes Verzeichnis
  + Cd /home/kochbar/server/server/src
    - python api.py

Die gestartete Anwendung horcht nun auf Serverport 5000, konkret erreichbar also unter <http://132.199.138.70:9009> . Für den Fall das der Dienst ausfällt muss Flask nicht neuinstalliert werden, sonder kann dann direkt mit dem zweiten Schritt begonnen und der Webserver gestartet werden. Sobald der Dienst unter oben genannter Adresse erreichbar ist erwartet dieser als Übergabe Paramater eine Json-Datei die einen Rezept-Titel String sowie einen Inhaltsstoff String. Eine beispielhafte Json-Datei kann im verwendete Github-Repository[[1]](#footnote-1) gefunden werden.

# Verwendung des Frontend/Datenbanksuche

Die entworfene Datenbanksuche, das Frontend, ist am Universitätsserver auf Port 9009 erreichbar, die Adresse ist folglich: <http://132.199.138.79:9009> . Zur Bereitstellung verwenden wir hierfür **node.js.** Zum aktuellen Zeitpunkt läuft der Webserver und ist aus dem Uni-Netz erreichbar. Für den Fall das der Dienst nicht mehr erreichbar ist, zum Beispiel als Folge eines Serverneustarts, kann die Webanwendung mit folgendem Linux Befehl wieder gestartet werden.

* Wechsel in das korrekte Verzeichnis
  + CD /home/kochbar/frontend\_final/dbAPI
* Starten des node.js Servers
  + screen
    - nodejs server.js
      * STRG+A
        + D
* Der Webserver läuft nun im Hintergrund und sollte über die obene genannte Adresse erreichbar sein.

Für die Darstellung der Rezepte, die Speicherung der Nutzereingabe (also die Auswahl der Gerichte), sowie für die Berechnungen des Nährwertbedarfs und des Nährwertverbrauchs, verwendet die Anwendung drei verschiedene SQL-Tabellen.

* kochbar.referencenuts
  + Speichert die empfohlen Tagesmengen der Nährwerte in Abhängigkeit der Usergruppe (Alter,Geschlecht) ab
* kochbar.users
  + Ordnet User zu gegessenen Gerichten zu
* kochbar.recipenuts
  + Diese Tabelle speichert die von uns berechneten Nährwerte zu den ca 300000 Rezepten aus dem ursprünglichen kochbar-Datensatz. Diese Tabelle wird auch für die Darstellung der Nährwerte in der Suchergebnisanzeige verwendet, sowie zu schlussendlichen Nährwertberechnung für den individuellen Benutzer.

1. Beispiel Json-Datei: <https://github.com/JakobFehle/KI4SG/blob/master/json/testparse.json> [↑](#footnote-ref-1)