Web-Services 2015

Einzelbericht

Team: JunkfoodAnalyzer

Name: Andreas Konsek

*Diese Vorlage dient zur Dokumentation des individuellen Beitrags zum Blockseminar in Ihrem Team. Bitte dokumentieren Sie hier in den jeweiligen Abschnitten diejenigen Themen, Klassen und weitere Bausteine, die maßgeblich Sie erstellt haben. Da die Teams zu vertikalen Arbeit aufgefordert waren, werden Sie in jedem der Bereiche vorraussichtlich entsprechende Eintragungen vornehmen können.*

*Bitte bennen Sie die von Ihnen erzeugten Ergebnisse so, dass die entsprechenden Inhalte im Repository Ihnen zugeordnet werden können. Dies kann durch Benennung des Dateinamens, aber auch von Klassen- und Methodennamen geschehen.*

# Use Cases

Benennen Sie, für welche Use-Cases Sie verantwortlich zeichnen.

* **Der Admin soll User anlegen können**
* **Der Admin soll User ändern können**
* **Der Admin soll User löschen können**

# Der User soll in der Hauptansicht eine Liste von konfigurierten Speisen zur Auswahl gestellt bekommen

* **Der Admin soll neue Zutaten und vorkonfigurierte Speisen anlegen können**
* **Der Admin soll die angelegten Speisen und Zutaten löschen können**

# Schnittstelle

Bennen Sie Ihren Anteil an der Erstellung der Service Schnittstelle. Benennen Sie Gründe, nach welchen Gesichtspunkten Sie die Schnittstelle entworfen haben

Den Web-Service Ingredient mit Modals

Teile des Web-Service Junkfood

Teile des Web-Service User

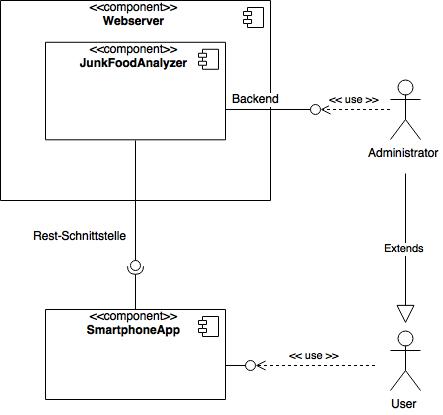
Modals auf aktuellem Stand halten (z.B.: auf Grund von Datenbankänderungen)

# Technische Architektur – Blockschaubild

Welche Bausteine der Architektur haben Sie angeregt bzw. entworfen? Beziehen Sie sich dazu bitte auf das Architekturbild und färben Sie ggfs. die Bereiche ein. Welche Gründe gab es für Ihre Entscheidung? Welche Bibiotheken haben Sie eingesetzt bzw. angeregt?

Ich war am Entwurf der Komponente Webserver, sowie der Komponente JunkFoodAnalyzer beteiligt.

Es wurde entschieden, dass der Administrator über ein Backend direkt auf den Server zugreifen kann, um eine klare Trennung von User und Admin zu errreichen.



# Technische Architektur – Datenhaltung

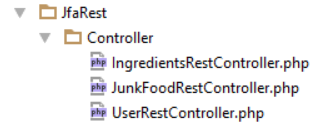
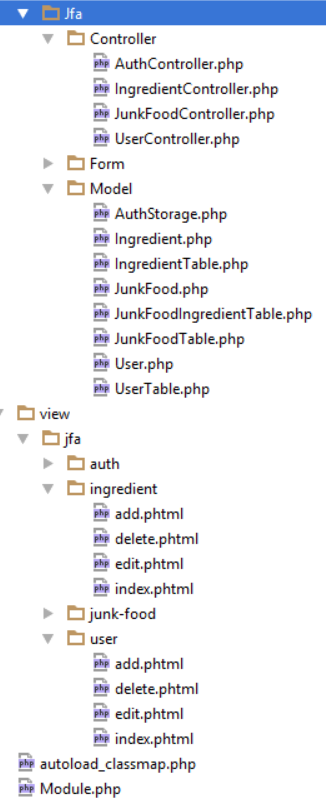
Beim Entwurf der Datenbank kam die Frage auf welche Daten des Users als Login-Daten verwendet werden sollten. Zur Auswahl standen E-Mail und Passwort, sowie Name und Passwort. Entschieden wurde, dass als Daten die Kombination Name und Passwort verwendet wird.

Ein weiterer Diskussionspunkt war die Realisierung der Speicherung der Bilder des jeweiligen Junkfoods. Zur Diskussion für die Speicherung standen die Möglichkeiten das Bild als ByteArray direkt in der Tabelle Junkfood zu speichern oder die Bilder auf dem Server zu speichern und den Dateinamen zu speichern. Hierbei wurde sich darauf geeinigt, dass die Bilder in einem festen Verzeichnis auf dem Server gespeichert werden und in der Spalte „ImagePath“ in der Tabelle „Junkfood“ der jeweilige Dateiname gespeichert werden soll.

Die sich ergebende m:n – Beziehung zwischen den Tabellen „Junkfood“ und „Ingredients“ wurde durch die Relationstabelle aufgelöst.

# Technische Architektur – Klassendiagramm

Welchen Teil der Klassen haben Sie entworfen und ggfs. durch Design Patterns mit gestaltet? Warum haben Sie dieses Design Pattern angewandt?



# Implementierung der App

Welche Klassen und Methoden der App haben Sie implementiert?

**IngredientController.php**

* indexAction()
* addAction()
* editAction()
* deleteAction()
* getIngredientTable()

**IngredientRestController.php**

* getList()
* get($id)
* create($data)
* update($id, $data)
* delete($id)
* getIngredientsTable()

**Ingredient.php**

* exchangeArray($data)
* getArrayCopy()
* setInputFilter(InputFilterInterface $inputFilter)

**IngredientForm.php**

**IngredientTable.php**

* fetchAll()
* getIngredient($id)
* saveIngredient(Ingredient $ingr)
* getIngredientsByJunkfood($junk)

**JunkfoodController.php**

* getJunkfoodTable()
* getIngredientTable()
* getRelationTable()

**UserController.php**

* addAction()
* editAction()
* deleteAction()