Systementwurf

Version 1.0.1

Informatikprojekt WS 2010/2011

Freigabe am: 18.11.2010

Dokumenthistorie

Version	Datum	Status	Autor	Bemerkung
1.0.0	11.10.10	angelegt	Lokalhorst (alle)	
1.0.1	17.11.10	erweitert	Zöllner	Ucs, Diag, Pattern
1.0.2	18.11.10	geändert	Pätzold	Ucs, ERM

Inhaltsverzeichnis

1 Grundlegende Architektur

Es handelt sich um eine Client-Server-Architektur. Die Anwendung kann eine Vielzahl von Nutzeranfragen über das Internet gleichzeitig abwickeln.

1.1 Architekturmuster

1.1.1 MVC

Die Kochrezepteplattform wird nach dem MVC-Muster erstellt. Die grundsätzliche Idee von Model-View-Controller besteht in einer Trennung der Objekte, welche die Logik einer Applikation und die Daten repräsentieren, von der konkreten Darstellung und Manipulation der Daten.

Das MVC-Architekturmuster umfasst drei wesentliche Elemente:

1. Model (Datenmodell und Geschäftslogik)
2. View (Präsentation)
3. Controller (Ablaufsteuerung)

Das **Model** repräsentiert das Datenmodell einer Anwendung und deren aktuellen Zustand. Es ist unabhängig von einer bestimmten Darstellung der Ausgabe oder vom Verhalten der Eingabe.

Die **View** entspricht der Präsentationsschicht einer Model-Komponente. Es können alle Informationen oder nur Teile des Models dargestellt werden. Sie sendet das Benutzerverhalten an den Controller oder es wird der aktuelle Zustand des Models abgefragt.

Der **Controller**, die Ablaufsteuerung der Anwendung, bestimmt das Verhalten, mit welchem auf Benutzereingaben mittels einer Benutzerschnittstelle reagiert wird. Es werden dabei Dienste der entsprechenden View oder des Models aufgerufen. Mit dem Controller wird der Zustand des Models verändert, weiter können Eigenschaften der Ausgabe manipuliert werden.

Die Vorteile des Model-View-Controller Musters liegen in seiner hohen Flexibilität, es sind Änderungen an Teilobjekten und verschiedene Views möglich. Des weiteren ist die Aufteilung in kleinere Objekte übersichtlicher und die Anwendung wird dadurch einfacher wartbar,auch wird eine Wiederverwertbarkeit gefördert. Durch die klare Trennung in drei Schichten wird die Ablaufsteuerung einer Anwendung merklich vereinfacht.

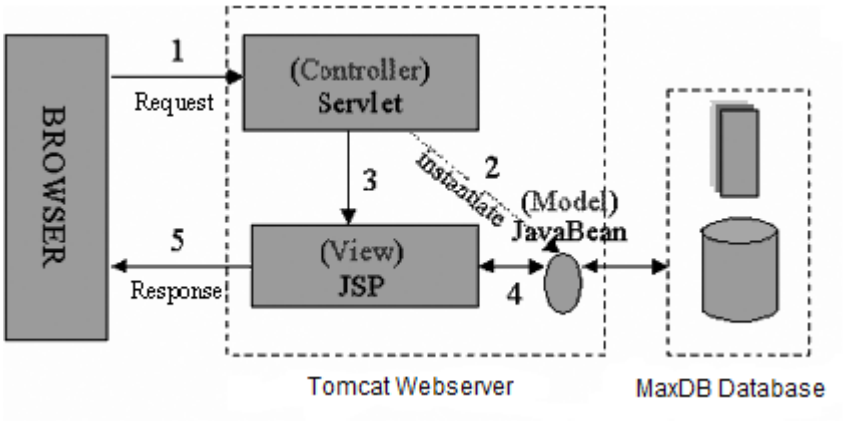
1.1.2 Konkrete Verwendung

Die Anwendung arbeitet grundsätzlich nach dem „Model-View-Controller-Prinzip“, d.h. es gibt eine logische Trennung zwischen Daten-, Geschäfts- und Darstellungslogik.

Die View-Schicht wird mit JSP (Java Server Pages) realisiert und generiert dynamisch die Anwendungsoberfläche in HTML, die im Webbrowser (Client) angezeigt wird.

Der Controller ist ein Servlet. Gemäß der Anwengunslogik instanziiert er die notwendigen Daten und leitet sie an die passende JSP weiter.

Das Modell bildet alle Objekte ab, deren Daten persistent gespeichert werden müssen. Die Struktur der Daten liegt sowohl als Java-Klassen (JavaBeans), als auch als Beschreibung in einer Datenbank vor.



1.2 Entwurfsmuster

Die Kochrezepteplattform wird mit Hilfe des Command-Patterns erstellt.

Grundlegend handelt es sich hierbei um ein Entwurfsmuster, bei dem jeder Befehl, der von einem Benutzer am System aufgerufen werden kann in einem eigenen Objekt gekapselt wird. Das Befehlsobjekt enthält dabei alle Informationen, die für die Ausführung des von ihm repräsentierten Befehls notwendig sind.

Im konkreten Fall der Kochrezepteplattform ,heißt das nun, dass das Controllerservlet im wesentlichen aus einer Instanz eines sogenannten CommandBroker besteht.

Am CommandBroker sind alle Commandobjekte und somit auch alle auf der Kochrezepteplattform vom Nutzer aufrufbaren Befehle registriert.

Das Controllerservlet nimmt nun den Befehlsnamen entgegen, veranlasst über den CommandBroker dessen Ausführung und reagiert dann entsprechend auf das Ergebnis der Ausführung.

2 Systemkomponenten

2.1 Use Cases

VORBEMERKUNG: *Unter „Nutzer“ sind alle Benutzer der Plattform unabhängig vom Status (Gast, registrierter Nutzer, Administrator) zu verstehen. Welchen Status der Nutzer haben muss, ergibt sich aus der Kategorie „Akteure“ u/o wird in „Ablauf“ bzw. „Alternativen“ erklärt.*

2.1.1. Use Case „Suchen“

Name:	Suchen
Vorbedingung:	Der Nutzer hat einen frei formulierten Begriff in das Textfeld eingegeben und ein Filterkriterium (Bsp.: Kategorie, Rezept, User etc.) ausgewählt.
Nachbedingung:	Suchbegriff wurde ein- oder mehrmals gefunden. Suchergebnis(se) wird/werden angezeigt.
Nachbedingung Misserfolg	Der Suchbegriff wurde nicht gefunden. Ein entsprechender Hinweis wird angezeigt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer drückt den Knopf „Suchen“.
Ablauf:	1. Abhängig vom Filterkriterium wird in der entsprechenden Datenbanktabelle nach dem eingegebenen Begriff gesucht. 2. Wird ein Eintrag zum gesuchten Begriff gefunden, so wird dieser zusammen mit dem Rezeptnamen angezeigt. Dieser kann dann vom Nutzer ausgewählt werden. 3. Wurde kein passender Eintrag gefunden, wird dies dem Nutzer mitgeteilt.
Alternativen:	a) zu 1.: Keine Benutzereingabe im Textfeld: Es wird keine Suche durchgeführt. Der Nutzer wird informiert, dass eine Eingabe zur Suche benötigt wird und hat die Möglichkeit, einen Suchbegriff einzugeben und ein Filterkriterium zu wählen. b) zu 2.: Hat ein Nutzer mit Admin-Rechten nach einem registrierten Nutzer gesucht, kann er statt des Rezepts den Nutzer auswählen.

2.1.2 Use Case „Nutzer auswählen“

Name:	Nutzer auswählen
Vorbedingung:	Nutzer ist als Administrator eingeloggt. Ihm wurde in Use Case „Suchen“ ein registrierter Nutzer angezeigt.
Nachbedingung:	Nutzerdaten werden angezeigt.
Nachbedingung Misserfolg	Nutzerdaten werden nicht angezeigt.
Akteure:	Admin
Ausl. Ereignis:	Administrator wählt in Ergebnisliste der Suche den gesuchten registrierten Nutzer aus.
Ablauf:	1. Nutzerdaten (Bsp.: registrierter Nutzer hat Admin-Status j/n) werden angezeigt und können vom Administrator geändert oder der registrierte Nutzer gelöscht werden.
Alternativen:	-

2.1.3 Use Case „Nutzer löschen“

Name:	Nutzer löschen
Vorbedingung:	Use Case „Nutzer anzeigen“, Nutzer ist als Administrator eingeloggt.
Nachbedingung:	Der Nutzer wird in der Datenbank gelöscht.
Nachbedingung Misserfolg	Der registrierte Nutzer wird nicht gelöscht.
Akteure:	Admin
Ausl. Ereignis:	Admin wählt in Use Case „Nutzer auswählen“ „Nutzer löschen“.
Ablauf:	1. Nachfrage, ob der Nutzer wirklich gelöscht werden soll. 2. Der Admin bestätigt dies. 3. Alle von dem zu löschenden reg. Nutzer abhängigen Daten bzgl. „Rezepte“, „Zutaten“ etc. werden mit dem Superuser „Chefkoch“ verknüpft. 4. Die restlichen Nutzerdaten werden in der DB gelöscht. 5. Mitteilung, dass der Nutzer gelöscht wurde.
Alternativen:	a) zu 2.: Der Admin bestätigt nicht und beendet so den Löschvorgang. b) zu 3.: Nicht alle relevanten Daten konnten mit „Chefkoch“ verknüpft werden: Nutzer wird nicht gelöscht. Mitteilung, dass nicht alle Daten gelöscht werden konnten. c) zu 4.: Nicht alle Nutzerdaten konnten gelöscht

	werden: Nutzer wird nicht gelöscht. Mitteilung, dass nicht alle Daten gelöscht werden konnten.

2.1.4 Use Case „Rezept auswählen“

Name:	Rezept auswählen
Vorbedingung:	Nutzer hat in UC „Suche“ eine nicht leere Liste an Suchergebnissen erhalten.
Nachbedingung:	Das gewählte Rezept wird angezeigt.
Nachbedingung Misserfolg	Das gewählte Rezept wird nicht angezeigt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Akteur klickt den passenden Listeneintrag an.
Ablauf:	1. Es wird geprüft, ob der Nutzer eingeloggt ist. 2. Die zugehörigen Daten (Zutaten, Mengen, Zubereitungsschritte, Personenzahl etc.) werden aus der Datenbank geladen und zusammen mit den Wahlmöglichkeiten „Zutaten auf Einkaufsliste“ und „Rezept drucken“ angezeigt. 3. Die Personenzahl wird initial mit 4 angezeigt. Der Nutzer kann diese jedoch durch Eingabe ändern.
Alternativen:	a) zu 1./2.: Ist der Nutzer eingeloggt, wird geprüft, ob er Autor des gewählten Rezepts ist und/oder Admin-Status hat. In diesem Fall werden zusätzlich noch die Auswahlmöglichkeiten „Rezept löschen“ und „Rezept bearbeiten“ angezeigt.

2.1.5 Use Case „Rezept drucken“

Name:	Rezept drucken
Vorbedingung:	UC „Rezept anzeigen“
Nachbedingung:	Rezept wurde gedruckt.
Nachbedingung Misserfolg	Das Rezept wurde nicht gedruckt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt in UC „Rezept auswählen“ die Schaltfläche „Rezept drucken“
Ablauf:	1. Die eingegebene Personenzahl wird eingelesen. 2. Dialog mit Anzeige der eingelesenen Personenzahl und Frage, ob der Nutzer diese verwenden möchte. 3. Nutzer bestätigt dies. 4. Die Mengenangaben werden entsprechend

	berechnet. 5. Eine druckbare Version des Rezepts wird angezeigt. 6. Nutzer druckt das Rezept über den Druckdialog des Browsers aus.
Alternativen:	a) zu 1.: Das Feld der Personenzahl ist leer, da z.B. der Nutzer die vorgegebene 4 gelöscht hat: Nutzer erhält entsprechende Mitteilung, der UC wird beendet und das Rezept angezeigt, wo der Nutzer die Personenzahl ändern kann. b) zu 2.: Nutzer möchte Personenzahl ändern: UC wird beendet und das Rezept angezeigt, wo der Nutzer die Personenzahl ändern kann.

2.1.6 Use Case „Rezeptzutaten in Einkaufsliste aufnehmen“

Name:	Rezeptzutaten in Einkaufsliste aufnehmen
Vorbedingung:	UC „Rezept anzeigen“
Nachbedingung:	Die Zutaten wurden auf die Einkaufsliste gesetzt.
Nachbedingung Misserfolg	Die Zutaten wurden nicht auf die Einkaufsliste gesetzt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „zu Einkaufsliste hinzufügen“.
Ablauf:	1. Die eingegebene Personenzahl wird eingelesen. 2. Dialog mit Anzeige der eingelesenen Personenzahl und Frage, ob der Nutzer diese verwenden möchte. 3. Nutzer bestätigt dies. 4. Die Mengenangaben werden entsprechend berechnet. 5. Bei leerer Einkaufsliste werden die Zutaten mit Mengenangabe und Einheit auf die Einkaufsliste übernommen. 6. Der Name des Rezepts wird ebenfalls auf der Einkaufsliste vermerkt.
Alternativen:	a) zu 1.: Das Feld der Personenzahl ist leer, da z.B. der Nutzer die vorgegebene 4 gelöscht hat: Nutzer erhält entsprechende Mitteilung, der UC wird beendet und das Rezept angezeigt, wo der Nutzer die Personenzahl ändern kann. b) zu 2.: Nutzer möchte Personenzahl ändern: UC wird beendet und das Rezept angezeigt, wo der Nutzer die Personenzahl ändern kann. c) zu 5.: Ist die Einkaufsliste nicht leer, wird für jede Rezeptzutat geprüft, ob sie bereits auf der Einkaufsliste vorkommt. In diesem Fall werden beide Mengenwerte addiert, sofern die Einheit übereinstimmt. d) zu c): Stimmen die Einheiten zweier gleicher Zutaten

	nicht überein, wird die hinzuzufügende Zutat mit ihrer Mengenangabe und Einheit der Einkaufsliste zugefügt und die bereits auf der Liste vorhandene gleiche Zutat unverändert belassen.

2.1.7 Use Case „Rezept löschen“

Name:	Rezept löschen
Vorbedingung:	UC „Rezept auswählen“; Login als Admin oder Login als reg. Nutzer, der Autor des Rezepts ist.
Nachbedingung:	Das Rezept wurde gelöscht.
Nachbedingung Misserfolg	Das Rezept wurde nicht gelöscht.
Akteure:	reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „Rezept löschen“
Ablauf:	1. Nutzer wird gefragt, ob er das Rezept wirklich löschen will. 2. Nutzer bestätigt dies. 3. Alle Rezeptdaten werden auf der Datenbank gelöscht. 4. Rückmeldung, dass das Rezepte erfolgreich gelöscht wurden.
Alternativen:	a) zu 2.: Nutzer bestätigt nicht: UC wird beendet und das Rezept angezeigt. b) zu 3./4.: Das Rezept kann nicht vollständig gelöscht werden: Mitteilung, dass das Rezept nicht gelöscht werden konnte.

2.1.8 Use Case „Rezept bearbeiten“

Name:	Rezept bearbeiten
Vorbedingung:	UC „Rezept auswählen“; Nutzer ist Admin Autor des Rezepts ist.
Nachbedingung:	Das Rezept wurde bearbeitet.
Nachbedingung Misserfolg	Das Rezept wurde geändert aber nicht frei gegeben.
Akteure:	Registrierter Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „bearbeiten“
Ablauf:	1. Das System Zeigt die Rezeptbestandteile sowohl in editierbarer Form (Beschreibung, Zubereitungsdauer, Zubereitungsschritte) als auch mit Auswahlfeldern an (Zutatennamen, Kategorie, Schwierigkeitsgrad) die vom Nutzer bearbeitet können 2. Speichern mit „Speichern“-Schaltfläche

Alternativen:	a) zu 1.: Hinzufügen („hinzufügen“) bzw. Entfernen („entfernen“) von Rezeptzutaten b) zu 1.: Anlegen („anlegen“) neuer Zutaten c) zu 1.: Hinzufügen („neu“) bzw. Entfernen („löschen“) von Zubereitungsschritten d) zu 2.: Die Checkbox “veröffentlichen“ wird vom Nutzer gesetzt

2.1.9 Use Case „Rezept speichern“

Name:	Rezept speichern
Vorbedingung:	UC „Rezept bearbeiten“ od. UC „neues Rezept anlegen“
Nachbedingung:	Das Rezept wurde gespeichert.
Nachbedingung Misserfolg	Das Rezept wurde nicht gespeichert.
Akteure:	reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „Rezept speichern“.
Ablauf:	1. Ist der Rezeptname nicht leer und noch nicht vorhanden/vergeben, sowie eine Kategorie gewählt, wird das Rezept gespeichert.
Alternativen:	a) zu 1.: Der Nutzer erhält eine Fehlermitteilung und hat die Möglichkeit, die entsprechenden Angaben zu machen.

2.1.10 Use Case „neue Zutat anlegen“

Name:	neue Zutat anlegen
Vorbedingung:	UC „Rezept bearbeiten“ od. UC „neues Rezept anlegen“
Nachbedingung:	neue Zutat wurde angelegt
Nachbedingung Misserfolg	Es wurde keine neue Zutat angelegt.
Akteure:	reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „Neue Zutat anlegen“
Ablauf:	1. Der Nutzer gibt in freier Form den Namen der Zutat ein, wählt eine der angegebenen Einheiten (ml, g, Stck., <leer>) aus und bestätigt seine Angaben. 2. Es wird geprüft, ob eine Zutat mit diesem Namen bereits angelegt wurde. 3. Die Zutat wird der Liste der Zutaten hinzugefügt. 4. Rückkehr zum auslösenden UC.
Alternativen:	a) zu 2.: Zutat bereits vorhanden: Der Nutzer wird informiert, dass diese Zutat bereits existiert. Rückkehr zum auslösenden UC.

2.1.11. Use Case „neues Rezept anlegen“

Name:	neues Rezept anlegen
Vorbedingung:	Login als registrierter Nutzer
Nachbedingung:	Ein neues Rezept wurde angelegt.
Nachbedingung Misserfolg	Es wurde kein neues Rezept angelegt. Das System gibt einen Hinweis aus und verbleibt im UC
Akteure:	reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer wählt Schaltfläche „Rezept anlegen“
Ablauf:	1. Der Nutzer gibt einen Rezeptnamen ein. 2. Der Nutzer trägt in freier Form eine Beschreibung des Rezepts ein. 3. Der Nutzer drückt die Schaltfläche „anlegen“. 4. UC „Rezept bearbeiten“.
Alternativen:	Keine

2.1.12 Use Case „Einkaufsliste leeren“

Name:	Einkaufsliste leeren
Vorbedingung:	Login als registrierter Nutzer, Einkaufsliste wird

	angezeigt.
Nachbedingung:	Die Einkaufsliste wurde geleert.
Nachbedingung Misserfolg	Die Einkaufsliste wurde nicht geleert.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	In UC „Einkaufsliste anzeigen“ wird die Taste „Einkaufsliste leeren“ betätigt.
Ablauf:	1. Der Nutzer erhält eine Meldung „Die Einkaufsliste wurde geleert“ 2. Die leere Einkaufsliste wird angezeigt.
Alternativen:	Keine

2.1.13 Use Case „Einkaufsliste anzeigen“

Name:	Einkaufsliste anzeigen
Vorbedingung:	-
Nachbedingung:	Die Einkaufsliste wird angezeigt.
Nachbedingung Misserfolg	Die Einkaufsliste wird nicht angezeigt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer klickt Schaltfläche „Einkaufsliste anzeigen““.
Ablauf:	1. Die Einkaufsliste wird angezeigt
Alternativen:	Keine

2.1.14 Use Case „Einkaufsliste drucken“

Name:	Einkaufsliste drucken
Vorbedingung:	Die Einkaufsliste wird angezeigt.
Nachbedingung:	Die Einkaufsliste wurde gedruckt.
Nachbedingung Misserfolg	Die Einkaufsliste wurde nicht gedruckt.
Akteure:	Gast, reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer klickt auf die Schaltfläche „Einkaufsliste Drucken“
Ablauf:	1. Die Einkaufsliste wird in einer druckerfreundlichen Version im Browser angezeigt. 2. Der Nutzer verwendet den Druckerdialog des Browsers um die Liste auszudrucken.
Alternativen:	Keine

2.1.15 Use Case „Einkaufsliste speichern“

Name:	Einkaufsliste speichern
Vorbedingung:	Die Einkaufsliste wird angezeigt, Nutzer eingeloggt.
Nachbedingung:	Die Einkaufsliste wurde in der Datenbank gespeichert.
Nachbedingung Misserfolg	Die Einkaufsliste wurde nicht in der Datenbank gespeichert.
Akteure:	reg. Nutzer, Admin
Ausl. Ereignis:	Nutzer klickt auf Schaltfläche „Einkaufsliste speichern“
Ablauf:	1. Die Einkaufsliste wird in der Datenbank gespeichert. 2. Mitteilung an den Nutzer, dass die Einkaufsliste gespeichert wurde.
Alternativen:	Keine

2.1.16 Use Case „Login“

Name:	Login
Vorbedingung:	Benutzer ist registriert
Nachbedingung:	Hinweistext über erfolgreiche Anmeldung, Nutzer ist nun angemeldet.
Nachbedingung Misserfolg	Der Nutzer ist nicht eingeloggt.
Akteure:	Registrierte Benutzer, Admin
Ausl. Ereignis:	Benutzer wählt „login“ im Login-Bereich der Seite
Ablauf:	1. Username eingeben 2. Password eingeben
Alternativen:	Keine

2.1.17 Use Case „Logout“

Name:	Logout
Vorbedingung:	Nutzer ist angemeldet
Nachbedingung:	Nutzer ist abgemeldet
Nachbedingung Misserfolg	Nutzer ist nicht abgemeldet
Akteure:	Registrierte Nutzer, Admin
Ausl. Ereignis:	Benutzer wählt Schaltfläche „logout“ im Account-Bereich
Ablauf:	1. Nutzer wird abgemeldet.
Alternativen:	Keine

2.1.18 Use Case „Registrierung“

Name:	Registrieren
Vorbedingung:	Nutzer ist nicht angemeldet
Nachbedingung:	Nutzer ist registriert und eingeloggt.
Nachbedingung Misserfolg	Das System gibt einen Hinweistext und verbleibt im UC
Akteure:	Gast
Ausl. Ereignis:	Gast wählt „Registrieren“
Ablauf:	1. Username eingeben 2. Password eingeben 3. Password wiederholen 4. Geheimfrage eingeben 5. Geheimantwort eingeben 6. Mit „Registrieren“-Button“ bestätigen
Alternativen:	Keine

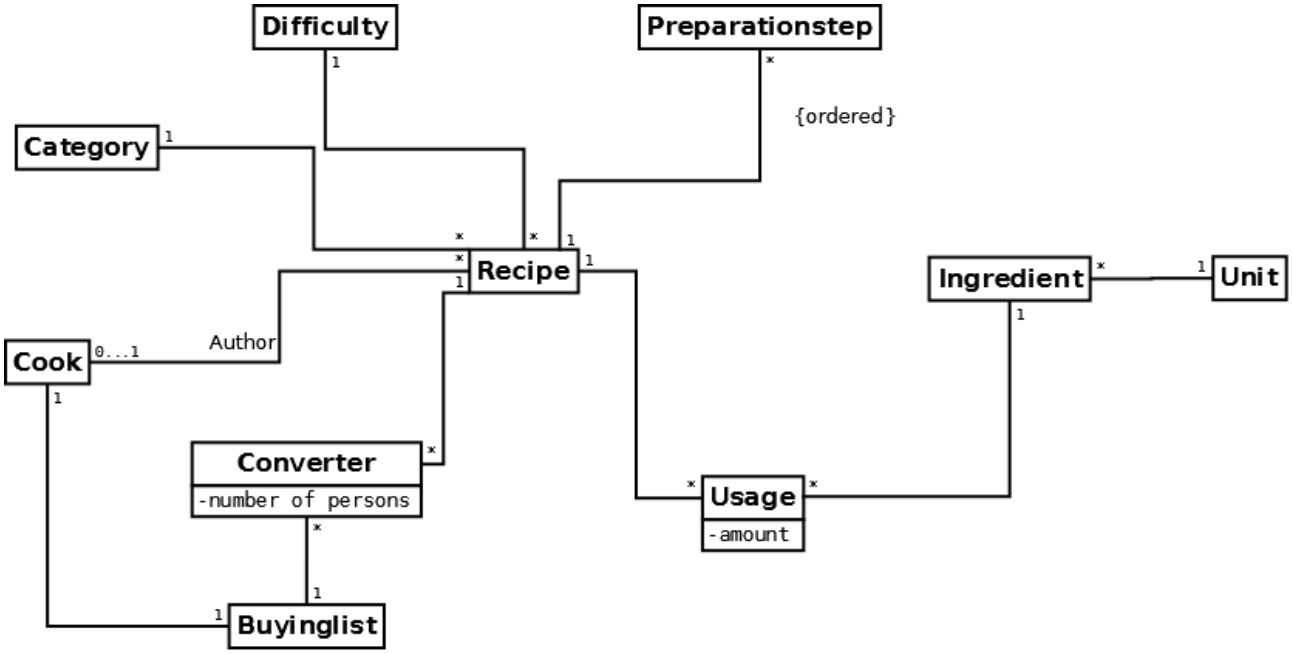
2.1.19 Use Case „Kategorie verwalten“

Name:	Kategorie verwalten
Vorbedingung:	Nutzer als Admin eingeloggt
Nachbedingung:	Kategorie wurde verwaltet
Nachbedingung Misserfolg	Kategorie wurde nicht verwaltet
Akteure:	Admin
Ausl. Ereignis:	Admin wählt „Kategorie anlegen“ oder „Kategorie löschen“ oder „Kategorie bearbeiten“
Ablauf:	1. Kategorienamen eingeben und mit „Anlegen“ bestätigen. 2. Eingabe wird überprüft 3. Admin wird informiert, dass Kategorie angelegt wurde.
Alternativen:	a) zu 1.: Admin bearbeitet vorhandene Kategorie: Kategoriename wird geprüft und entsprechend geändert. b) zu a): Admin löscht vorhandene Kategorie: Kategorie wird gelöscht und Admin entsprechend informiert. c) zu 2./a): Kategoriename bereits vorhanden: Kategorie wird nicht angelegt bzw. geändert. Admin wird entsprechend informiert und kann es erneut versuchen. d) zu b): Kategorie kann nicht gelöscht werden: Admin wird entsprechend informiert und kann es erneut versuchen.

2.1.20 Use Case „Passwort vergessen“

Name:	Passwort vergessen
Vorbedingung:	Nutzer ist nicht angemeldet
Nachbedingung:	Nutzer ist nicht angemeldet
Nachbedingung Misserfolg	Das System gibt einen Hinweistext und verbleibt im UC
Akteure:	Nicht angemeldeter Nutzer
Ausl. Ereignis:	Nutzer wählt „Passwort vergessen“
Ablauf:	1. Eingabe des registrierten Namens. 2. Bestätigt mit „Suchen“-Schaltfeld 3. Das System stellt die gegeime Frage zum gesuchte Nutzernamen. 4. Eingabe der Antwort zur geheimen Frage 5. Eingabe neues Passwort 6. Wiederholung des neuen Pssworts 7. „Passwort ändern“-Schaltfläche drücken
Alternativen:	Keine

2.2 Fachliches Klassendiagramm

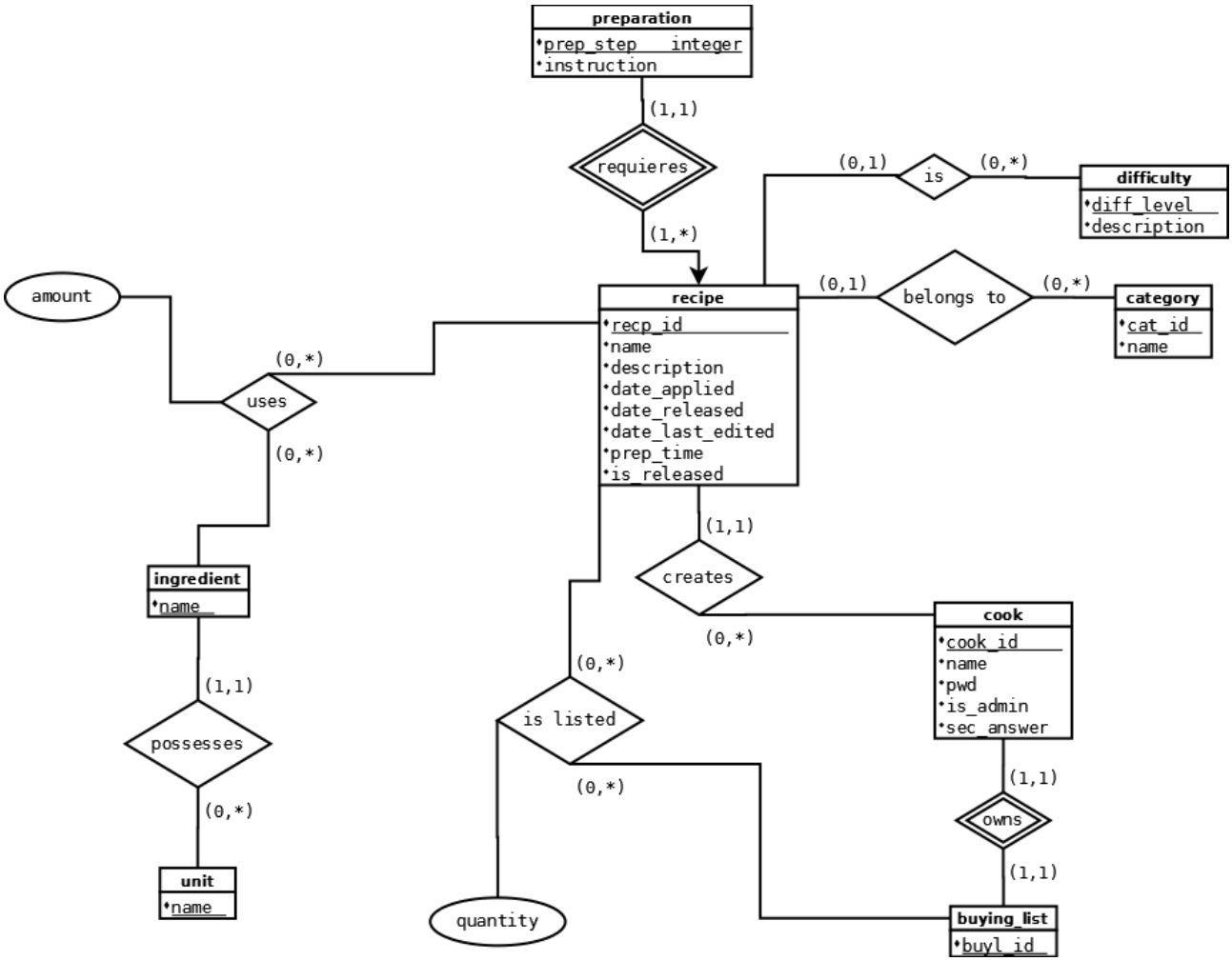


Die detaillierten Klassendiagramme mit vollständiger API Beschreibung, befinden sich in den Dokumenten:

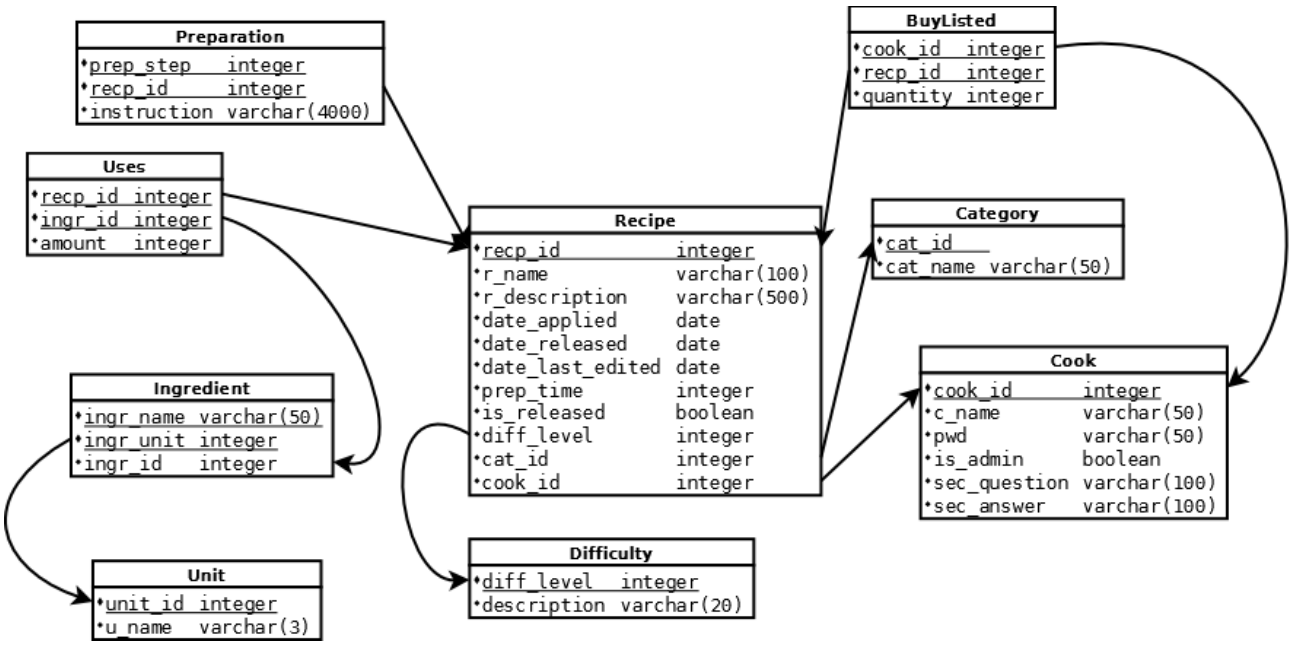
- de_lokalhorst_command.PNG
- de_lokalhorst_DB.PNG
- de_lokalhorst_DB_DTO.PNG
- de_lokalhorst_helper.PNG
- de_lokalhorst_servlet.PNG
- de_lokalhorst_all.PNG

3 Datenbank

3.1 ERM



3.2 RDM


Preparation
prep_step integer
recp_id integer
instruction varchar(4000)
Uses
recp_id integer
ingr_id integer
amount integer
Ingredient
ingr_name varchar(50)
ingr_unit integer
ingr_id integer
Unit
unit_id integer
u_name varchar(3)
Recipe
recp_id integer
r_name varchar(100)
r_description varchar(500)
date_applied date
date_released date
date_last_edited date
prep_time integer
is_released boolean
diff_level integer
cat_id integer
cook_id integer
Difficulty
diff_level integer
description varchar(20)
BuyListed
cook_id integer
recp_id integer
quantity integer
Category
cat_id
cat_name varchar(50)
Cook
cook_id integer
c_name varchar(50)
pwd varchar(50)
is_admin boolean
sec_question varchar(100)
sec_answer varchar(100)

4 Weitere technische Maßnahmen

- Verbindung zur Datenbank mit Hilfe der Connectionpooling Möglichkeit von Tomcat

5. Definition von Commands und Eingabe-/Ausgabe-Daten

5.1 Bestandteile eines Commands

Für die Realisierung jedes Commands sind folgende Bestandteile notwendig:

- Ein Formular mit den Eingabe-Parametern des Commands
- Eine **Command-Klasse** zur Ausführung des Commands und zur Generierung der Ausgabe-Daten
- Eine *oder mehrere* **Data Beans** mit den Ausgabe-Daten des Commands oder Daten-Objekte im Kontext der Session (eckigen Klammern, z.B. [cook])
- Eine *oder mehrere* **JSP**, die sich um die Darstellung des Ausgabe-Daten kümmern

Anmerkung: Ein Command muss nicht immer die selbe Art von Daten liefern und zur Selben JSP-Seite führen. Im folgenden werden nur die „positiven“ Fälle bei der Ausführung eines Commands berücksichtigt.

5.2 Übersicht der Commands

Command	Eingabedaten	Ausgabedaten	Zielseite
AddRecipeToBuyList	[buylist] [recp_name_bl] [recp_id_bl] [quantity] buylistentries	buylist summation	ShowBuyList
CalculateAmounts	recipeattributes ingredients prepsteps persons	recipe	ShowRecipe
ClearBuyList	[buylist] [cook]		ShowBuyList
CreateRecipe	[cook] name desc	[recipe] [ingredients] [difficulties] [categories] [units] recp_id	EditRecipe
DeleteCook	cook_id cook_name [cook]		AdminFunctions
EditCook	adminstatus adminstatus_old cook_id		AdminFunctions

Command	Eingabedaten	Ausgabedaten	Zielseite
	cook_name [cook]		
EditRecipe	recp_id [recipe] [ingredients] [difficulties] [categories] [units] edit release desc prep_time difficulty category prepstep_#	[recipe] [ingredients] [difficulties] [categories] [units] recp_id [release]	EditRecipe
LoginCook	username password	[cook]	Default
LogoutCook	[cook]		Default
PrintSite	param recipeattributes ingredients prepsteps *alternativ* buylist	Recipe *alternativ* buylist summation	PrintRecipe *alternativ* PrintBuyList
Redirect	targetview		*„Eingabedatum targetview"*
RegisterCook	newName newPassword newPassword_2 newQuestion newAnswer	newName newPassword newPassword_2 newQuestion newAnswer	Default
SaveBuyList	[cook] [buylist]	Buylist summation	ShowBuyList
SearchCook	search_term	cooks recipes	SearchresultCook
SearchOwnRecipe	[cook]	recipes	SearchresultOwnRecipe
SearchRecipe	search_term	recipes	SearchresultByName
SecretName	name	[sec_cook]	SecretQuestion
SecretQuestion	answer password password_2 [sec_cook]	[Cook]	Default
ShowBuyList	[buylist]	buylist summation	ShowBuyList
ShowRecipe	param	recipe	ShowRecipe