Hochschule RheinMain
Medieninformatik

Architekturdokumentation - Arc42

Lagersystemsteuerung für Pizzabetriebe

im Zeitraum
29.04.2024 - 08.07.2024

vorgelegt von

Artur Konkel, Anna-Livia Martin,
Sarah Schwarzer, Vivien Weber

Projektfertigstellung: 08.07.2024

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Revisionshistorie

Version	Datum	Autor(en)	Änderungen
1.00	31.05.2024	Anna-Livia Martin	Initiales Aufsetzen des Projekts
1.01	02.06.2024	Artur Konkel	Hinzufügen der Bausteinsicht-Methoden und dessen Beschreibungen
1.02	07.06.2024	Vivien Weber	Hinzufügen der Themen ’Einführung und Ziele’, ’Randbedingungen’ und ’Kontextabgrenzung’
1.03	07.06.2024	Artur Konkel	Entfernen von nicht korrekt dokumentierten Bausteinsichten
1.04	10.06.2024	Vivien Weber	Einfügen des Sequenzdiagrammes ’Lageraufbau’ und ’Zutatenpaket-Erstellung’ und deren Beschreibungen
1.05	10.06.2024	Vivien Weber	Einfügen des Sequenzdiagrammes ’Platzierung eines Zutatenpakets’ und dessen Beschreibung
1.06	11.06.2024	Sarah Schwarzer	Einfügen des Sequenzdiagrammes ’Entnahme eines Zutatenpakets’ und dessen Beschreibung
1.07	11.06.2024	Vivien Weber	Einfügen des Funktionsablaufs ’Entnahme’
1.08	12.06.2024	Artur Konkel	Anpassung der Titelseite
1.09	12.06.2024	Artur Konkel	Hinzufügen der Gesamtübersicht des Systems
1.10	12.06.2024	Artur Konkel	Einfügen der Bausteine ’Editor’ und ’Entnahme’ inkl. Beschreibung
1.11	13.06.2024	Artur Konkel	Einfügen des Bausteins ’Platzierung’ inkl. Beschreibung
1.12	14.06.2024	Anna-Livia Martin	Anpassung des Thementitel für bessere Verständlichkeit
1.13	14.06.2024	Anna-Livia Martin	Aktualisierung der Sequenzdiagramme ’Entnahme’, ’Lagereditor’, ’Platzierung’ und ’Zutatenpaketverwaltung’
1.14	15.06.2024	Vivien Weber	Korrektur grammatikalischer Fehler
1.15	16.06.2024	Artur Konkel	Aktualisierung der Bausteine ’Entnahme’ und ’Platzierung’

2 Einführung und Ziele

In diesem Abschnitt erfolgt eine Einführung in das Lagersystem, seine Architektur und erläutert die Ziele, die das Lagersystem verfolgt.

2.1 Einführung

Ziel des Lagersystems ist es, die Verwaltung von Zutatenpaketen in einem Lager zu ermöglichen. Dies umfasst die Platzierung, Entnahme, Verwaltung und Speicherung von Zutatenpaketen sowie die Bearbeitung des Lagers selbst.

2.2 Ziele

Die Hauptziele des Lagersystems sind:

- **Effiziente Lagerverwaltung:** Das System soll eine effiziente Verwaltung der Lagerbestände ermöglichen, einschließlich der Platzierung und Entnahme von Zutatenpaketen.
- **Flexibilität**: Das System soll durch die Editier-Funktion des Zutatenlagers auf individuelle Lager-Bedürfnisse des Benutzers eingehen können.
- **Sicherheit:** Das System soll bei der Lagerung dafür sorgen, dass keine Zutatenpakete in einem gemeinsamen Ablagebereich gelagert werden, dessen Inhalte sich nicht vertragen. Außerdem soll die Größe überprüft werden, damit es kein Paket gibt, dessen Proportionen die des Ablagebereiches übertreffen und beim Stapeln auf einem weiteren Paket darf das untere Paket nicht schmaler sein.
- **Benutzerfreundlichkeit:** Das System soll eine intuitive Benutzeroberfläche bieten, die eine einfache und schnelle Bedienung ermöglicht.
- **Erweiterbarkeit:** Die Architektur soll flexibel und erweiterbar sein, um zukünftige Anpassungen und Erweiterungen zu unterstützen.
- **Datenintegrität:** Sicherstellung der Datenintegrität bei allen Operationen, insbesondere bei der Speicherung und Verwaltung von Lagerdaten.
- **Zuverlässigkeit:** Das System soll zuverlässig und robust sein, um einen kontinuierlichen Betrieb zu gewährleisten.

3 Randbedingungen

Der folgende Abschnitt stellt die Randbedingungen dar, welche bei der Entwicklung des Lagersystems berücksichtigt wurden:

3.1 Technische Randbedingungen

- **Technologie:** Das System wird in Java implementiert und nutzt bestehende Java-Bibliotheken für die Verwaltung der Lagerdaten.
- **Plattform:** Das System soll auf dem Betriebssystem Linux laufen.
- **Leistung:** Das System muss in der Lage sein, ein großes Zutatenlager effizient zu verwalten, ohne dass die Performance beeinträchtigt wird.
- **Sicherheit:** Die Daten müssen gegen unbefugten Zugriff geschützt sein. Dazu gehören Maßnahmen zur Zugriffskontrolle und Datenverschlüsselung.
- **Skalierbarkeit:** Das System muss skalierbar sein, um mit wachsenden Anforderungen und Datenmengen umgehen zu können.

3.2 Organisatorische Randbedingungen

- **Das Team:** Die Entwicklung erfolgt durch ein festes Team von Entwicklern und wird durch regelmäßige Meetings koordiniert. Die Entwickler sind Artur Konkel, Anna-Livia Martin, Sarah Schwarzer und Vivien Weber.
- **Zeitplan:** Die Entwicklung beginnt im Juni und soll bis zum 08. Juli desselben Jahres abgeschlossen sein, mit regelmäßigen Meilensteinen zur Überprüfung des Fortschritts.
- **Vorgehensmodell:** Es wird ein agiles Vorgehensmodell genutzt, um flexibel auf Änderungen reagieren zu können.
- **Entwicklungswerkzeuge:** Entwicklung mit IntelliJ IDEA als IDE, Versionsverwaltung mit Git und Organisation mit Miro.

3.3 Konventionen

- **Architekturdokumentation:** Nutzung des arc42-Templates in der Version 1.0 für die Dokumentation.
- **Sprache:** Die Dokumentation und der Quellcode werden in Deutsch verfasst, um die Zielgruppe, die hauptsächlich aus deutschsprachigen Benutzern besteht, bestmöglich zu erreichen.

4 Kontextabgrenzung

Das Lagersystem interagiert mit verschiedenen externen Systemen und Komponenten. Wie das Umfeld von dem Lagersystem aussieht, beschreibt der folgende Abschnitt:

4.1 Fachlicher Kontext

Lagerist (Benutzer) Der Lagerist überwacht und steuert die gesamten Lagerprozesse. Er interagiert direkt mit dem System, um Zutatenpakete zu platzieren, zu entnehmen und um das Lager zu editieren durchzuführen. Er verwendet die Benutzeroberfläche des Systems, um die Lagerbestände zu verwalten und um den Lager-Editor zu bedienen.
Pizzabäcker (Benutzer) Der Pizzabäcker übernimmt die Organisation der benötigten Zutatenpakete. Er interagiert direkt mit dem System, besitzt jedoch eingeschränkte Zugriffsrechte. Er verwendet die Benutzeroberfläche des Systems, um die Lagerbestände zu verwalten.

5 Lösungsstrategie

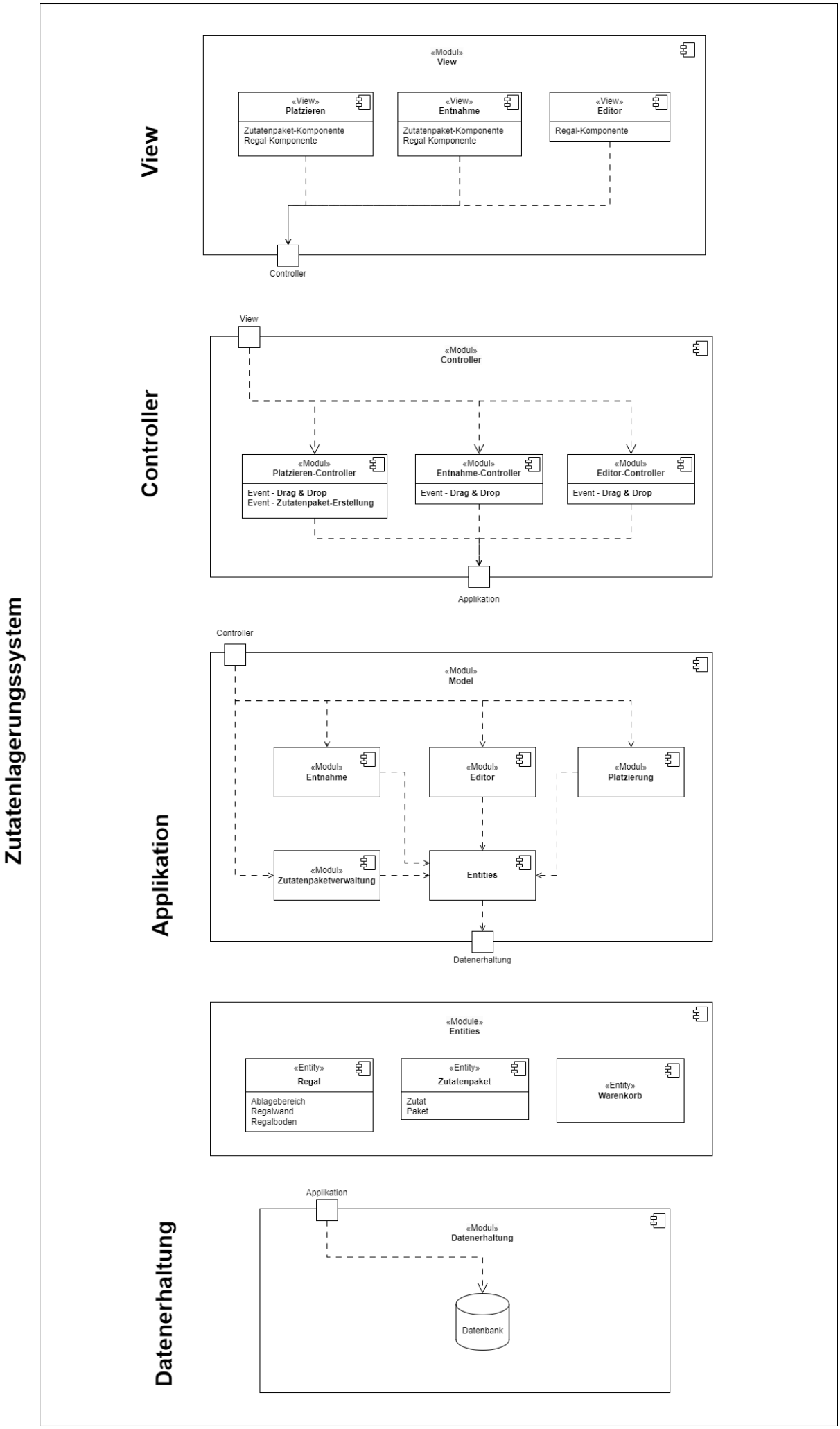
6 Systemübersicht

Um einführend eine Übersicht zu erhalten, welche Komponenten im System die Hauptrollen spielen und in welcher der drei Bestandteile des MVC-Pattern sich diese befindet, erläutert die Systemübersicht.

Das System besteht auf der Interaktionsoberfläche aus den drei verschiedenen Views Platzieren, Entnahme und dem Editor. Die Platzieren,- und die Entnahme-View beinhalten zusätzlich die Zutatenpaket-Komponente. Die Regalkomponente ist Bestandteil bei allen Views. Zusammen definieren sie die View im MVC-Pattern , welche die Benutzerinteraktionen entgegennimmt und diese an ihre Controller weiterleitet. Als Observer am Model erkennen die Views Änderungen und aktualisieren sich.

Die Views haben parallel zu sich ihre Controller, die für die Steuerung der eingehenden Benutzerinteraktionen zuständig sind. Sie dienen als Vermittler zwischen der View und dem Model. Sie verarbeiten die Benutzereingaben und aktualisieren daraufhin das Model und die View, wenn nötig. So ist der Platzieren-Controller, Entnahme-Controller und der Editor-Controller für die weiterverarbeitung der Events, wie Drag &Drop und der Zutatenpaket-Erstellung zuständig.

Im Model ist die tatsächliche Geschäftslogik implementiert. Die Entnahme beinhaltet die Logik für das Entfernen von Zutatenpaketen aus dem Regal. Die Platzierung ist hingegen für das korrekte Platzieren der Zutatenpakete zuständig. Die Zutatenpaketverwaltung besitzt die Aufgabe aus Zutaten und Paketen Zutatenpakete zu erstellen und diese zu verwalten. Der Editorhat die Zuständigkeit, das Regal korrekt aufzubauen und abzubilden. Das Model stellt der View und dem Controller Möglichkeiten zur Verfügung, um Änderungen bekannt zu geben. Alle drei verwenden zur Umsetzung ihrer Logik die benötigten Entitäten Regal, Zutatenpaket und den Warenkorb. Diese werden mit ihren Eigenschaften und Beziehungen in der Datenerhalten in der Datenbank persistiert.


Zutatenlagerungssystem
View
«Modul»
View
«View»
Platzieren
Zutatenpaket-Komponente
Regal-Komponente
«View»
Entnahme
Zutatenpaket-Komponente
Regal-Komponente
«View»
Editor
Regal-Komponente
Controller
Controller
View
«Modul»
Controller
«Modul»
Platzieren-Controller
Event - Drag & Drop
Event - Zutatenpaket-Erstellung
«Modul»
Entnahme-Controller
Event - Drag & Drop
«Modul»
Editor-Controller
Event - Drag & Drop
Applikation
Applikation
Controller
«Modul»
Model
«Modul»
Entnahme
«Modul»
Editor
«Modul»
Platzierung
«Modul»
Zutatenpaketverwaltung
Entities
Datenerhaltung
«Module»
Entities
«Entity»
Regal
Ablagebereich
Regalwand
Regalboden
«Entity»
Zutatenpaket
Zutat
Paket
«Entity»
Warenkorb
Datenerhaltung
Applikation
«Modul»
Datenerhaltung
Datenbank

7 Bausteinsicht

Dieser Abschnitt beschreibt die Zerlegung des Lagersystems in Module, wie sie sich auch in der Paketstruktur des Java-Quellcodes widerspiegelt. Die Beschreibung konzentriert sich nur auf die Bestandteile und dessen Funktionen. Die Module selbst verwenden Entitäten in ihren Funktionen.

7.1 Baustein - Lager

Die View integriert die Regal-View, welche für die Anzeige aller Regale zuständig ist. Die View wiederum ist mit dem Editor-Controller verbunden. Der Controller hört auf Events der View. Wird ein Event durch einen User getriggert, bekommt dies der Controller mit. Er delegiert daraufhin eventabhängig zu unterschiedlichen Methoden des Editor-Services. Dieser kümmert sich um die Verarbeitung und Aktualisierung der Businessdaten. Veränderungen dieser Businessdaten wiederum bekommt die View über lose Kopplung mit und aktualisiert sich daraufhin.

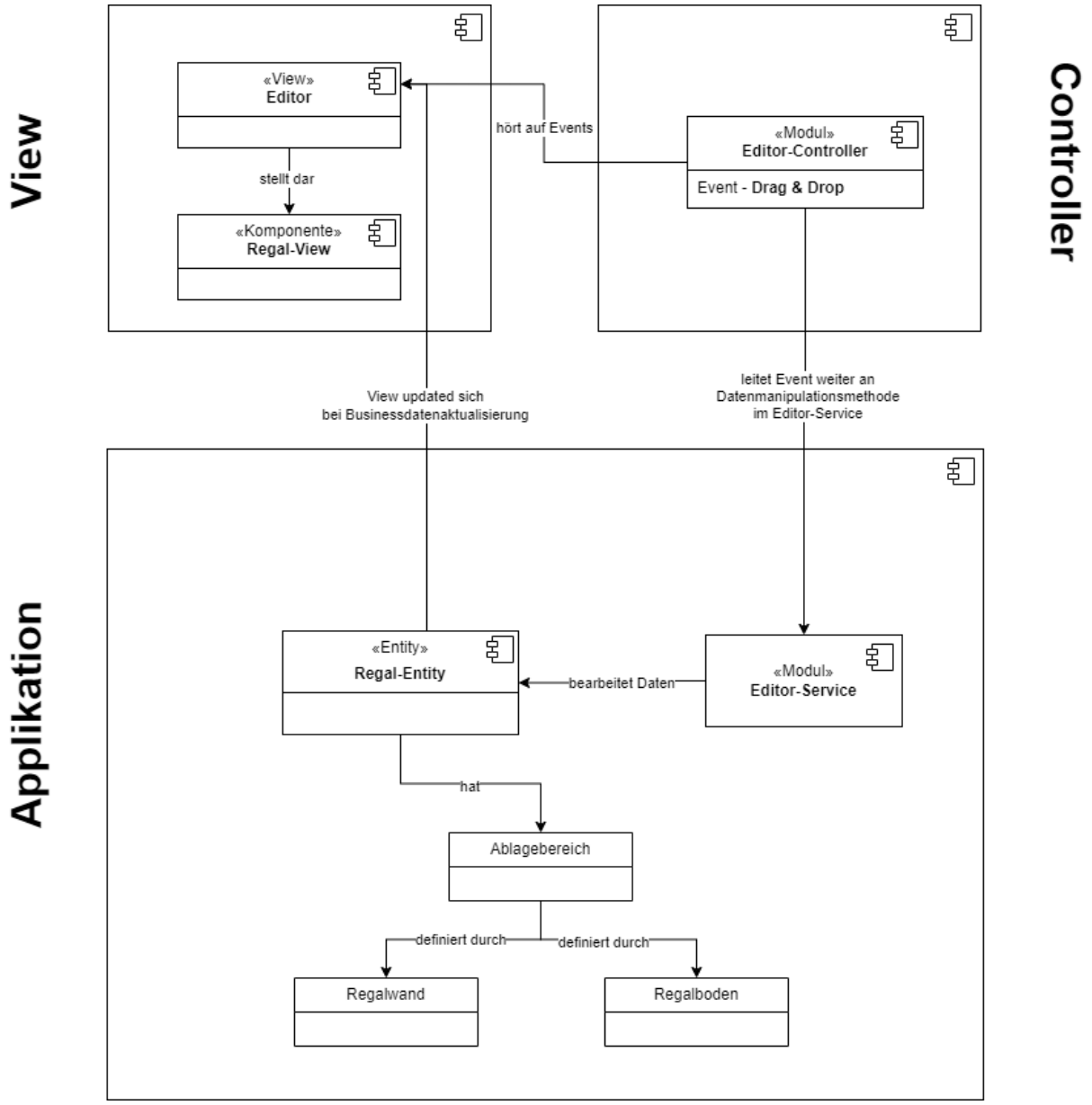


Abbildung 2: Baustein Editor

7.2 Baustein - Platzierung

Für die Anzeige der Regale, der Zutatenpakete und der Such-Komponente, ist die View zuständig, welche die Platzierungs-View integriert. Die View ist mit dem Platzierungs-Controller verbunden und dieser hört auf die Events (Drag & Drop) der View. Zudem bekommt er mit, wenn der User ein Event anstößt und delegiert dies eventabhängig zu den verschiedenen Methoden des Platzierungs-Service und Zutatenpaketverwaltungs-Service. Diese kümmern sich um die Verarbeitung und Aktualisierung der Businessdaten. Veränderungen dieser Businessdaten bekommt wiederum die View über lose Kopplung mit und aktualisiert sich daraufhin.

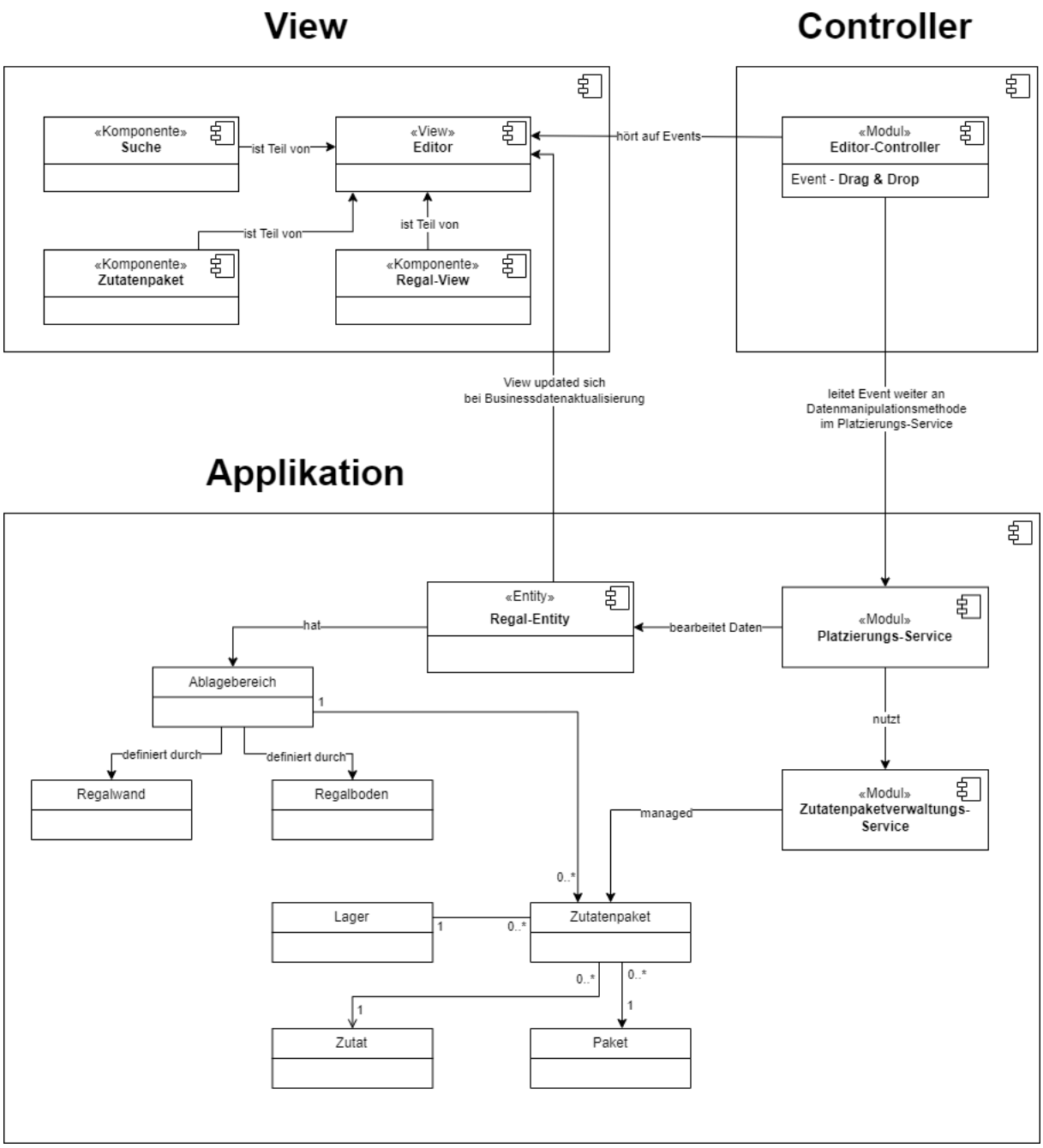


Abbildung 3: Baustein Platzierung

7.3 Baustein - Entnahme

Die Entnahme-View wird in der View integriert, welche für die Anzeige aller Regale, aller Zutatenpakete und der Such-Komponente zuständig ist. Die View ist mit dem Entnahme-Controller verbunden, der auf die Events (Drag & Drop) der View hört. Der Entnahme-Controller erfährt von den Events, die der User anstößt und delegiert daraufhin, abhängig von dem angestoßenen Event, zu unterschiedlichen Methoden des Entnahme- und Zutatenpaketverwaltung-Service. Diese kümmern sich um die Verarbeitung und Aktualisierung der Businessdaten. Der Zutatenpaketverwaltungs-Service verwaltet den Warenkorb, der Entnahme-Service managed die Zutatenpakete. Veränderungen der Businessdaten bekommt die View über lose Kopplung mit und aktualisiert

sich daraufhin.

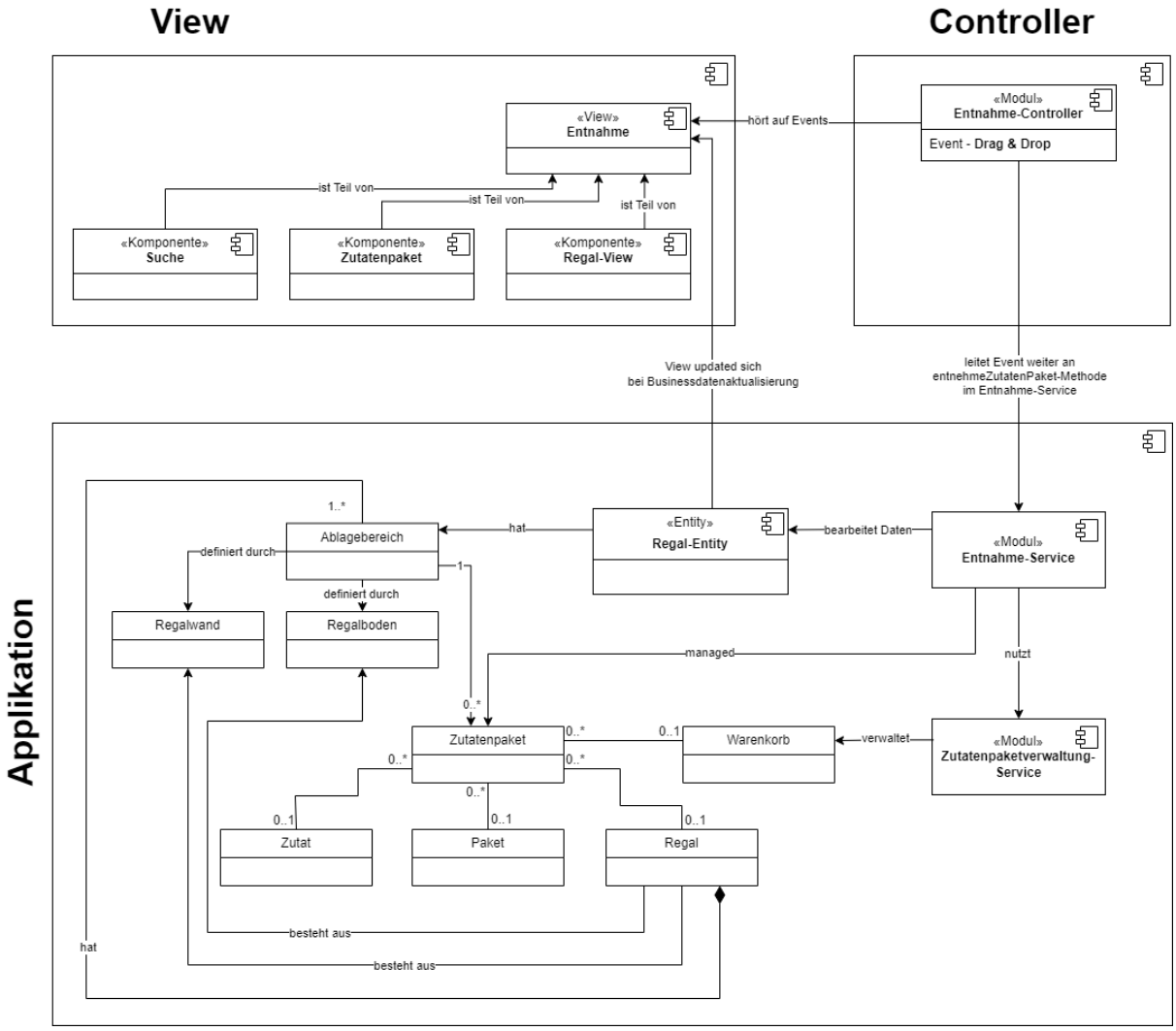


Abbildung 4: Baustein Entnahme

8 Servicemethoden

8.1 Entnahme

8.1.1 Zweck/Verantwortlichkeit

Die Entnahme ist für das Entnehmen und Entfernen von Zutatenpaketen im Lager zuständig.

8.1.2 Schnittstellen

Die Funktionalität wird mithilfe einer Java Klasse mit dem Namen Entnahme zur Verfügung gestellt.

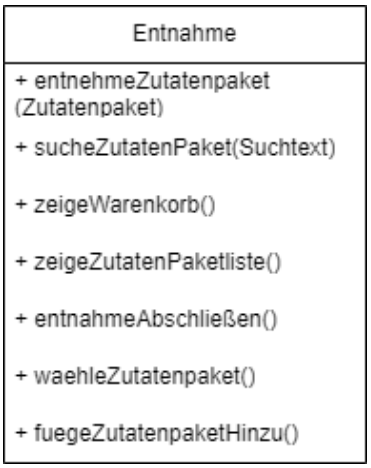


Abbildung 5: Entnahme

Methode	**Kurzbeschreibung**
entnehmeZutatenPaket(ZutatenPaket)	Entfernt das Zutatenpaket aus dem Regal und fügt es dem Warenkorb hinzu
sucheZutatenpaket(Suchtext)	Möglichkeit zur Suche des gewünschten Zutatenpakets im Lager
zeigeWarenkorb()	Anzeige der bereits entnommenen Zutatenpakete
zeigeZutatenpaketliste()	Anzeige der im Lager befindlichen Zutatenpakete
entnahmeAbschließen()	Abschließen des Entnahmeprozesses und dessen Speicherung
waehleZutatenpaket()	Wählt ein Zutatenpaket aus der Zutatenpaketverwaltung aus
fuegeZutatenpaketHinzu	Fügt ein Zutatenpaket in den Warenkorb hinzu

Tabelle 2: Entnahme Methoden

8.2 Platzierung

8.2.1 Zweck/Verantwortlichkeit

Das Modul Platzierung ist für die Positionsfindung der Zutatenpakete im Lager zuständig. Hierbei muss diese beachten, dass Zutatenpakete nur dort platziert werden könne, wo Platz ist. Zutatenpakete dürfen nur auf anderen Zutatenpakete platziert werden, wenn diese die nötige Tragfähigkeit besitzen. Außerdem dürfen Zutatenpakete nicht mit anderen Zutatenpaketen in einem Lagerbereich mit Unverträglichkeiten zusammen sein.

8.2.2 Schnittstellen

Die Funktionalität wird mithilfe einer Java Klasse mit dem Namen Platzierung zur Verfügung gestellt.

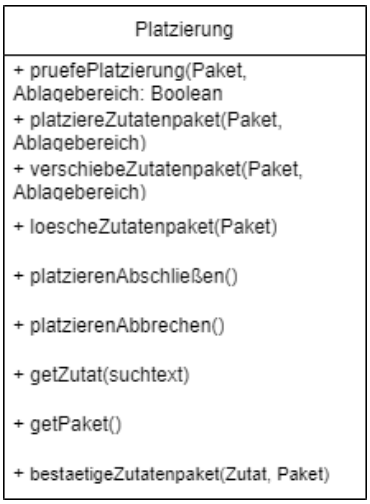


Abbildung 6: Platzierung

Methode	**Kurzbeschreibung**
pruefePlatzierung(Paket, Ablagebereich): Boolean	Prüfung der Plazierung des Zutatenpakets, ob Platz vorhanden ist, Tragfähigkeit gegeben ist und keine Unverträgichkeiten im Lagerbereich vorhanden sind.
platziereZutatenpaket(Paket, Ablagebereich)	Entgültige Platzierung des Zutatenpakets und hinzufügen zum Lager
verschiebeZutatenpaket(Paket, Ablagebereich)	Verschieben des Zutatenpakets in seiner Position mit Bebachtung der Pruefkriterien.
loescheZutatenpaket(Paket)	Entfernen des Zutatenpakets aus dem Lager
platzierenAbschließen()	Abschließen und Speichern des Platzierprozesses
platzierenAbbrechen()	Abbrechen des Platzierprozesses und herstellen des vorherigen Zustands
getZutat(suchtext)	Holt die Zutate mit dem Titel suchtext aus der Zutatenpaketverwaltung
getPaket()	Holt das Paket aus der Zutatenpaketverwaltung
bestaetigeZutatenpaket(Zutat, Paket)	Bestätigt der Zutatenpaketverwaltung die Zutat und das Paket. Daraufhin wird ein neues Zutatenpaket von ihr erstellt.

Tabelle 3: Platzierung Methoden

8.3 Zutatenpaketverwaltung

8.3.1 Zweck/Verantwortlichkeit

Die Zutatenpaketverwaltung dient zur Erstellung und Verwaltung von Zutaten, Paketen und Zutatenpaketen, die aus diesen zwei Bestandteilen bestehen.

8.3.2 Schnittstellen

Die Funktionalität wird mithilfe einer Java Klasse mit dem Namen Zutatenpaketverwaltung zur Verfügung gestellt.

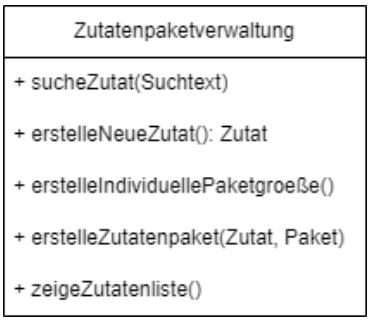


Abbildung 7: Zutatenpaketverwaltung

Methode	Kurzbeschreibung
sucheZutat(Suchtext)	Ermöglicht die Suche nach einer vorhandenen Zutat
erstelleNeueZutat()	Erstellt eine neue Zutat und fügt sie der Zutatenliste hinzu
erstelleIndividuellePaketgroeße()	Erstellt zur den Standardmaßen von Paketen, neue individuelle Pakete.
erstelleZutatenpaket(Zutat,Paket)	Zusammen mit einem Paket und einer Zutat wird ein Zutatenpaket erstellt.
zeigeZutatenliste()	Zeigt die vorhanden Zutaten.

Tabelle 4: Zutatenpaketverwaltung Methoden

8.4 Editor

8.4.1 Zweck/Verantwortlichkeit

Im Editor kann man das Regal im Lager mithilfe von Regalböden und Regalwänden aufbauen und diese bei Bedarf ändern. Anhand des Aufbaus definieren sich die einzelnen Lagerbereiche. Bestehende Lager können nur editiert werden, wenn sich keine Zutatenpakete in ihnen befinden.

8.4.2 Schnittstellen

Die Funktionalität wird mithilfe einer Java Klasse mit dem Namen Editor zur Verfügung gestellt.

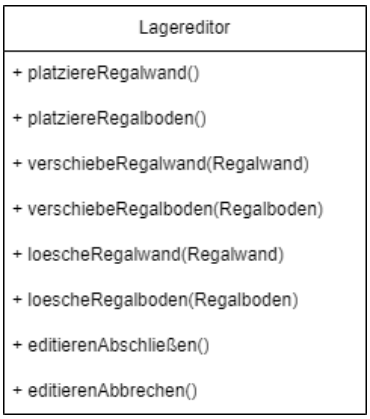


Abbildung 8: Lagereditor

Methode	Kurzbeschreibung
platziereRegalwand()	Platzierung der Regalwand an geeigneten Koordinaten. Regalwände können nur vertikal platziert werden.
platziereRegalboden()	Platzierung des Regalbodens. Regalboden kann nur zwischen zwei Regalwänden platziert werden.
verschiebeRegalwand(Regalwand)	Änderung der Koordinaten der Regalwand. Anpassung der Regalböden an diese Änderung.
verschiebeRegalboden(Regalboden)	Änderung der Koordinaten des Regalbodens. Diese wird Vertikal zwischen zwei Regalwänden verschoben.
loescheRegalwand(Regalwand)	Löschen einer Regalwand. Gelöschte Regalwände löschen auch die an sich abstützenden Regalböden.
loescheRegalboden(Regalboden)	Löschen des Regalbodens.
editierenAbschließen	Speicherung der Änderungen des Lagers.
editierenAbbrechen	Verwerfen der Änderungen und Herstellen des ursprünglichen Zustands.

Tabelle 5: Lagereditor Methoden

9 Laufzeitsicht

9.1 Sequenzdiagramm – Platzierung einer Regalwand

Das Sequenzdiagramm im unteren Abschnitt zeigt eine exemplarische Interaktion des Systems bei der Platzierung einer Regalwand.

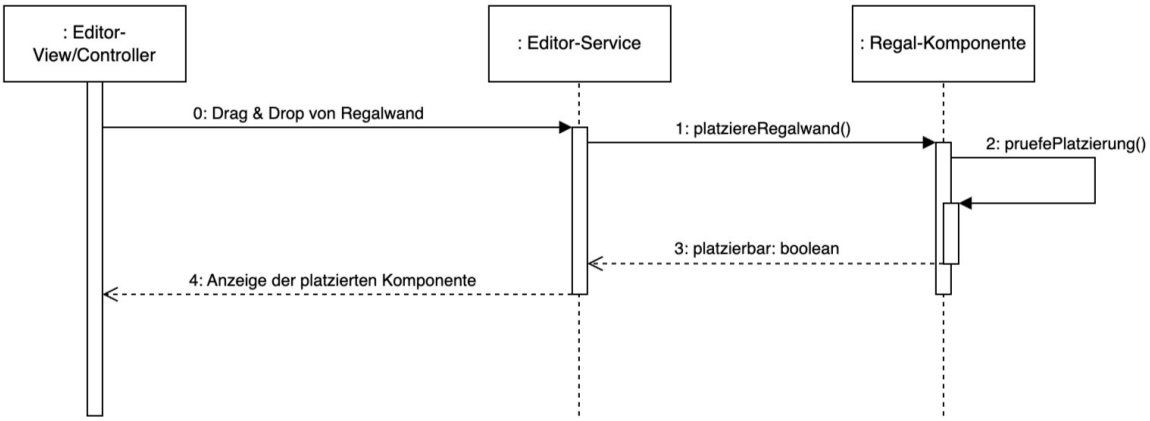


Abbildung 9: Sequenzdiagramm Platzierung Regalwand

Der User platziert eine Regalwand auf seine gewünschte Position in der Regalansicht, das ein Drag & Drop-Event im Controller auslöst. Dieses Event löst die Methode platziereRegalwand() im Editor-Service aus. Die dadurch angestoßene Regal-Komponente prüft nun, ob die Platzierung möglich ist, mit der Methode pruefePlatzierung(). Die Regal-Komponente gibt ein Boolean zurück. Wenn die Regalwand platzierbar ist, wird sie daraufhin platziert und in der Regalansicht erfolgreich angezeigt.

9.2 Sequenzdiagramm – Erstellung eines Zutatenpakets

Das Sequenzdiagramm im folgenden Abschnitt zeigt eine exemplarische Interaktion des Systems bei der Erstellung eines Zutatenpakets.

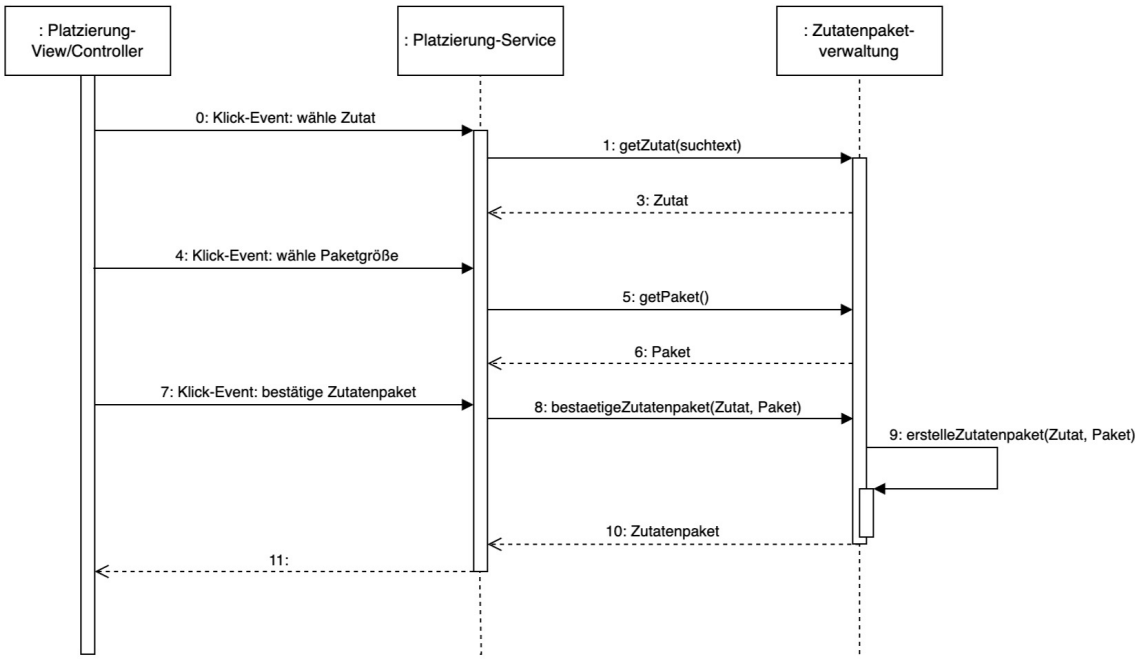


Abbildung 10: Sequenzdiagramm Erstellung eines Zutatenpakets

Durch ein Klick-Event wird eine Zutat ausgewählt. Dadurch wird in Platzierung-Service die Methode getZutat(suchtext) ausgelöst, durch die Zutatenpaket-Verwaltung wird die gewünschte Zutat als Return zurückgegeben. Durch ein weiteres Klick-Event wird das Paket ausgewählt, dadurch wird die Methode getPaket() im Platzierung-Service aufgerufen und die Zutatenpaket-Verwaltung gibt das gewählte Paket als Objekt zurück. Durch das letzte Klick-Event wird das konfigurierte Zutatenpaket bestätigt, was im Platzierung-Service die Methode bestaetigeZutatenpaket(Zutat, Paket) auslöst. In der Zutatenpaket-Verwaltung wird das Zutatenpaket erstellt mit der Methode erstelleZutatenpaket(Zutat, Paket), welches dann der Platzierung-Service erhällt und durch die View angezeigt wird.

9.3 Sequenzdiagramm – Platzierung eines Zutatenpakets

Das Sequenzdiagramm im Bild unten zeigt eine exemplarische Interaktion des Systems bei Platzierung eines Zutatenpaketes.

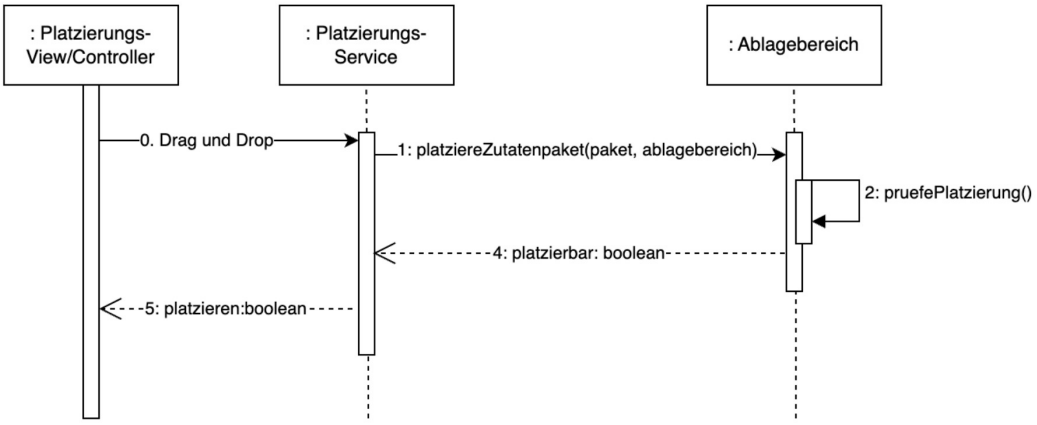


Abbildung 11: Sequenzdiagramm Platzierung eines Zutatenpakets

Zunächst zieht der User das Zutatenpaket auf einen neuen Spot, dadurch wird ein Event im Controller ausgelöst. So wird die Methode platziereZutatenpaket(paket, ablagebereich) im Platzierung-Service ausgelöst. Diese testet nun, ob es möglich ist, das Paket an der gewünschten Stelle zu platzieren. Dafür fragt der Ablagebereich über die Methode pruefePlatzierung() das jeweilige Paket, welches neu platziert werden soll, ob die Größe und die Unverträglichkeiten passen. Das Paket gibt einen Boolean zurück. Ist das Paket laut diesem platzierbar, wird im letzten Schritt das Paket platziert. Ist das Paket nicht platzierbar, wird dies im Ablagebereich angezeigt.

9.4 Sequenzdiagramm – Entnahme eines Zutatenpakets

Das Sequenzdiagramm im darunter liegenden Abschnitt zeigt eine exemplarische Interaktion des Systems bei der Entnahme eines Zutatenpakets.

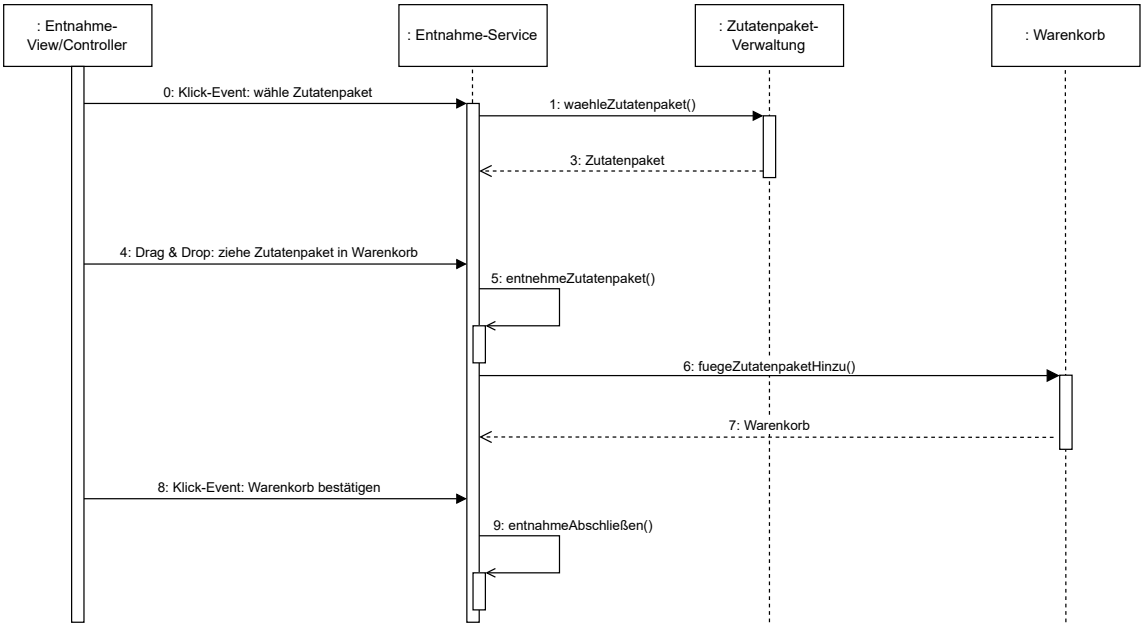


Abbildung 12: Sequenzdiagramm Entnahme eines Zutatenpakets

Als erstes wählt der User, durch ein Klick-Event, das gewünschte Zutatenpaket aus, welches er entnehmen möchte. Daraufhin wird die Methode waehleZutatenpaket() vom Entnahme-Service angestoßen. Die Zutatenpaket-Verwaltung gibt anschließend ein Zutatenpaket als Return zurück. Per Drag & Drop kann der User dann das Zutatenpaket in den Warenkorb ziehen. Dadurch prüft der der Entnahme-Service, ob die Entnahme möglich ist mit der Methode entnehmeZutatenpaket(). Im Anschluss stößt der Entnahme-Service die Methode fuegeZutatenpaketHinzu() an, worauf der Warenkorb einen Warenkorb als Return liefert. Durch ein weiteres Klick-Event bestätigt der User den Warenkorb. Der Entnahme-Service schließt zum Schluss mit der Methode entnahmeAbschließen() das Event ab.

10 Ergänzende Bausteine

10.1 Glossar

Java	**plattformunabhängige, objektorientierte Programmiersprache, die weit verbreitet für Web- , Mobil- und Unternehmensanwendungen verwendet wird**
CSS3	**Cascading Style Sheets-Standards, der für die Gestaltung und das Layout von Webseiten verwendet wird.**
Paket	**Darstellung eines Paket mit Maßen und Gewicht**
Zutat	**Darstellung einer Zutat**
Zutatpaket	**Kombination aus Zutat und Paket**
Regal	**Selbst definierter und zusammengesetzer Bereich mit der Möglichkeit Zutaten abzustellen .**
Drag & Drop	**Interaktionsmöglichkeit mit Objekten, diese per Mausklick zu z iehen und woanders zu platzieren**
Inhaltsunverträglichkeit	**Zutaten können mit anderen Zutaten Unverträglichkeiten bilden und können somit nicht zusammen eingelagert werden.**
Login	**Authentifizierung des Benutzers am System**
Warenkorb	**Sammlung der Zutaten in einem Bereich zur Entnahme.**

Tabelle 6: Glossar

11 Quellenverzeichnis

11.1 Interne Quellen

- hier eine Beispielquelle
- noch eine Quelle.com