Version	Bearbeitungsdatum	Autor
1.0	09.05.2022	Thomas Berthold
1.1	16.05.2022	Robert Peter

1. Entwicklerdokumentation

2. Einführung und Ziele

Die Möbel-Hunger-Kette möchte das frisch erworbene Unternehmen Möbel-Hier mit einer Software für Bestellverwaltung, Lagerhaltung und Auslieferung ausstatten. Diese trägt den Arbeitstitel MHMPS.

Kunden (*customer*) können aus einem Katalog (*catalogue*) einzelne Möbel (*piece of furniture*) oder Möbel-Kombinationen aus Einzelmöbeln oder Möbelteilen (*bundles*) wählen. Ihre Bestellungen werden am Telefon von einem Mitarbeiter (*employee*) entgegengenommen und in der Bestellverwaltung (*orders*) abgelegt. Für die zügige Belieferung der Kunden unterhält Möbel-Hier ein Lager (*warehouse*), das ebenfalls durch MHMPS verwaltet werden soll. Für die Auslieferung (*delivery*) kann der Kunde zwischen Selbstabholung mit eigenem oder von Möbel-Hier geliehenem LKW (*truck*) oder Anlieferung durch Möbel-Hier wählen. Für die Geschäftsführung (*manager*) soll ein Überblick (*overview*) über die Möbelverkäufe einzelner Lieferanten (*vendor*) im monatlichen Vergleich erstellt werden.

MHMPS soll somit den Kunden eine gut strukturierte und angenehm präsentierte Auswahl an Möbeln anbieten. Für die Mitarbeiter ist eine übersichtliche, schnell zu bedienende Arbeitsumgebung wichtig, die Geschäftsführung hingegen benötigt im Wesentlichen einige übersichtliche Zusammenfassungen.

2.1. Qualitätsziele

Die folgenden Qualitätsanforderungen sollen von MHMPS erfüllt werden:

Anpassbarkeit

MHMPS soll leicht an geänderte Bedingungen angepasst werden können.

Bedienkomfort

Dem Kunden soll ein angenehmes, umsatzsteigerndes Einkaufserlebnis, den Mitarbeitern und der Geschäftsführung ein leicht zu bedienendes, übersichtliches Set an Tools geboten werden.

Sicherheit

Sämtliche Daten müssen vor Diebstahl und missbräuchlicher Verwendung geschützt werden.

Stabilität

MHMPS sollte rund um die Uhr stabil laufen.

In der folgenden Tabelle wird die Wichtigkeit der Ziele angegeben. 1 = unwichtig .. 5 = sehr wichtig

Quality Demand	1	2	3	4	5
Anpassbarkeit			x		
Bedienkomfort				x	
Sicherheit					x
Stabilität					x

3. Randbedingungen

3.1. Hardware-Vorgaben

- Server
- internetfähiges Endgerät

3.2. Software-Vorgaben

- Java 11 (oder höher)
- moderner Browser

3.3. Vorgaben zum Betrieb der Software

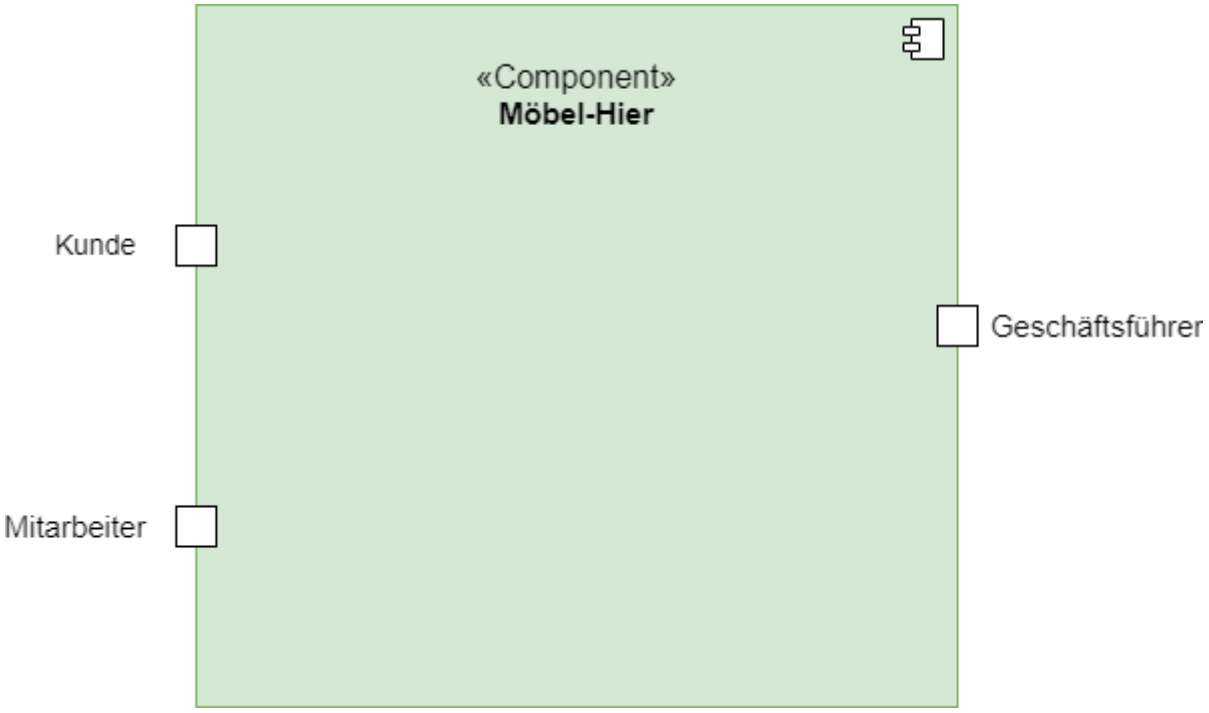
MHMPS soll interessierten Kunden rund um die Uhr einen Katalog mit den Angeboten von Möbel-Hier bieten. Diese benötigen einen Browser, um auf den Server zugreifen zu können, und minimale Grundkenntnisse in Web-Navigation.

Für die Mitarbeiter von Möbel-Hier ist MHMPS das wichtigste Arbeitswerkzeug. Während Kunden anrufen, um Bestellungen aufzugeben, zu ändern oder sich über den aktuellen Status zu informieren, nutzen die Mitarbeiter die angebotenen Tools, um die Kundenwünsche im System zu speichern, Verfügbarkeiten zu prüfen und zu ändern, ggfs. Produkte nachzuordern.

Die Geschäftsführung ist mit MHMPS jederzeit und ohne technische Kenntnisse in der Lage, sich einen Überblick über die aktuelle Geschäftsentwicklung zu verschaffen und diese mit den Vormonaten zu vergleichen. Dabei gibt es auch die Möglichkeit, dem Programm von Möbel-Hier neue Lieferanten hinzuzufügen oder bisherige Lieferanten zu entfernen.

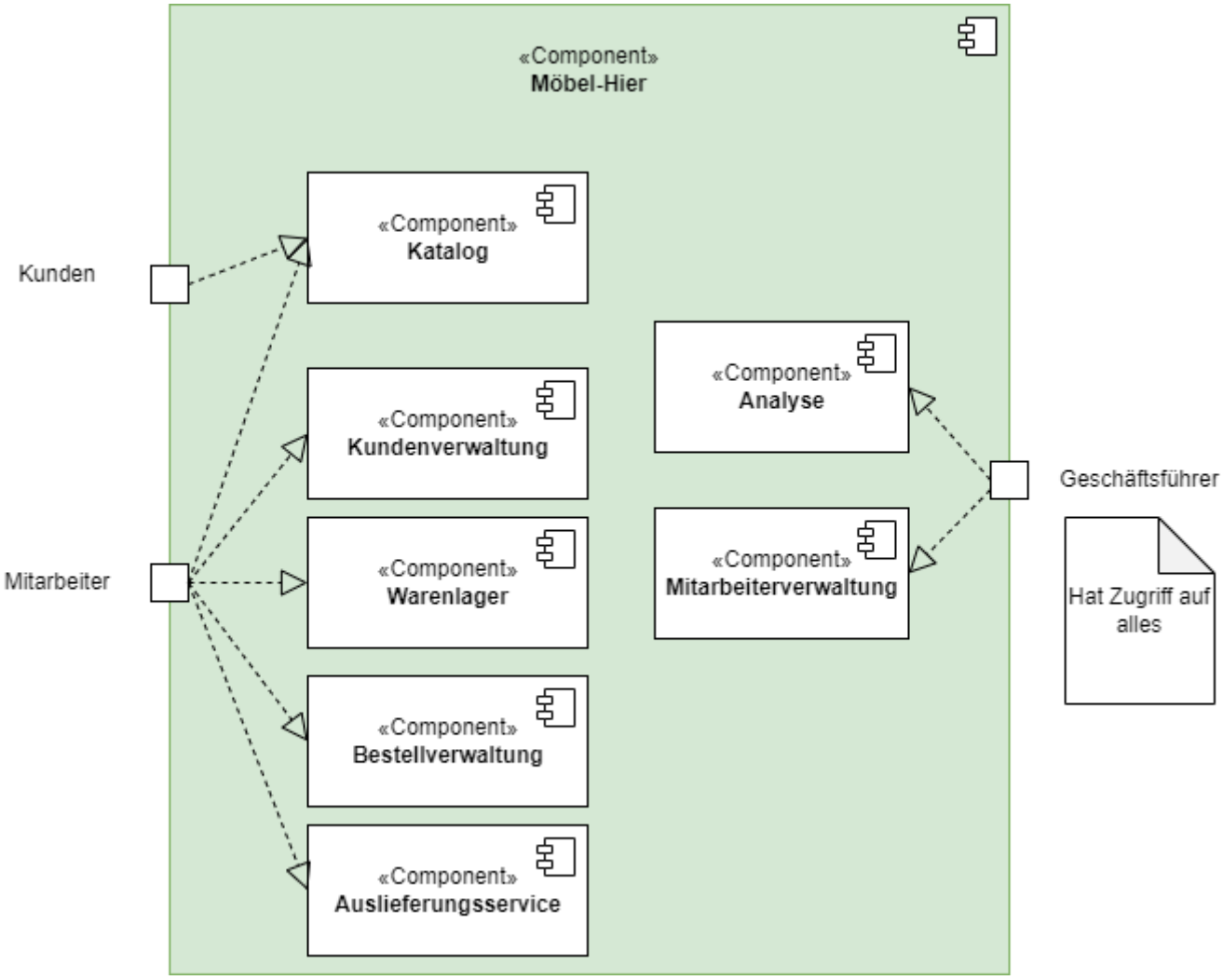
4. Kontextabgrenzung

Das Kontextdiagramm zeigt das geplante Software-System in seiner Umgebung. Zur Umgebung gehören alle Nutzergruppen des Systems und Nachbarsysteme.



4.1. Top-Level-Architektur

Top-Level-Architektur mit Hilfe eines Komponentendiagramms.

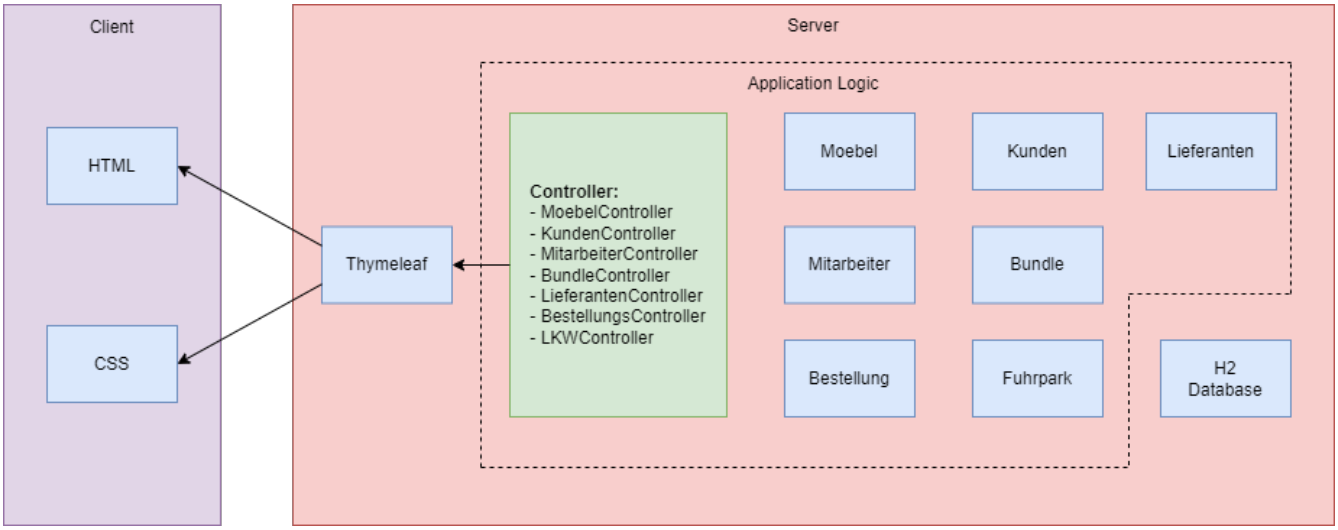


5. Lösungsstrategie

5.1. Erfüllung der Qualitätsziele

Qualitätsziel	Lösungsansatz
Anpassungsfähigkeit	• **modularer Aufbau** - Änderungen eines Moduls haben möglichst wenige Effekt auf andere Module. • **gute Dokumentation** - eine nachvollziehbare Dokumentation ermöglicht das Verständnis der Anwendung und ihrer Zusammenhänge. • **Wiederverwendbarkeit** - die Komponenten der Software können auch für andere Zwecke wieder verwendet werden.
Bedienkomfort	• **intuitive Bedienbarkeit** - Tooltips, klare Struktur, eindeutige Beschriftungen. • **Ästhetik** - angenehmes Design. • **Zugänglichkeit** - Nutzer sollen von verschiedensten Endgeräten auf den Katalog zugreifen können. • **Fehlerbehandlung** - Fehlbedienungen sollen nicht zu Systemabstürzen führen und dem Nutzer ein Feedback über die Ursache des Fehlers geben.
Sicherheit	• **Passwortschutz** - Zugriff auf die Mitarbeiter- und Geschäftsführungsfunktionen nur durch Authentifizierung. • **Zurechenbarkeit** - Nachverfolgung von Zugriffen, damit Fehler oder Missbrauch zurück verfolgt werden können. • **Integrität** - keine unauthorisierte Veränderung von Daten.
Stabilität	• **Testen** - umfangreiches Testen vor und während des Betriebs

5.2. Software-Architektur



Client-Server-Modell der Anwendung. Der Client erhält nur HTML- und CSS-Dateien. Die Logik findet auf der Serverseite statt.

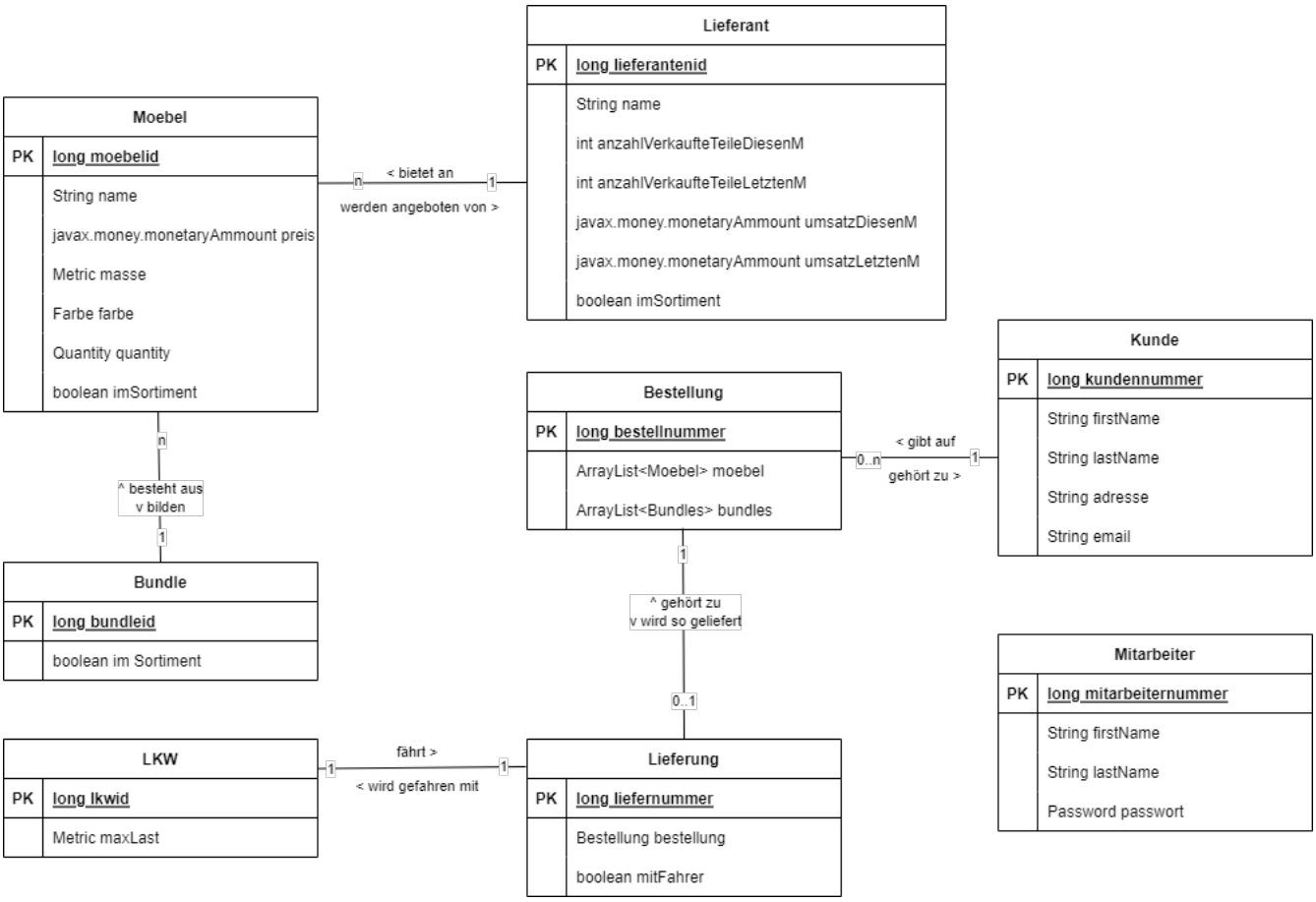
6. Entwurfsentscheidungen

6.1. Verwendete Muster

- **Spring MVC**

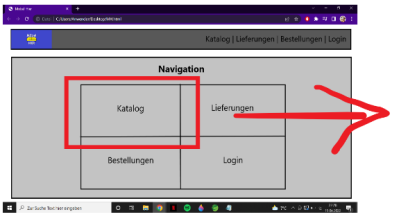
6.2. Persistenz

- **Hibernate**

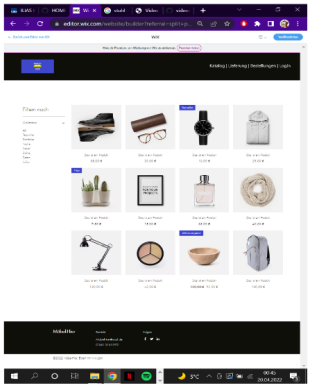


Das Persistenzmodell zeigt welche Tabellen in der Datenbank gespeichert werden sollen, welche Daten diese tragen und auch wie sie miteinander in Verbindung stehen.

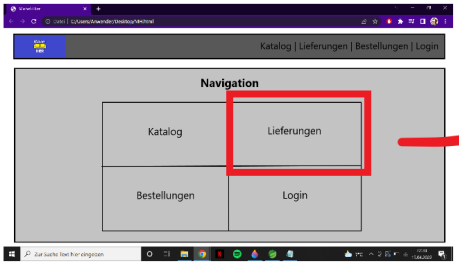
6.3. Benutzeroberfläche



Start

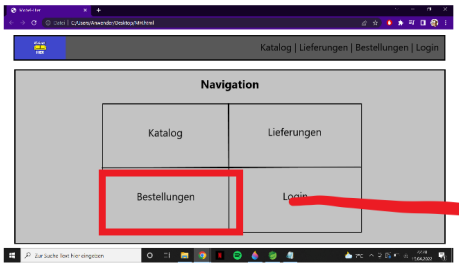


Katalog

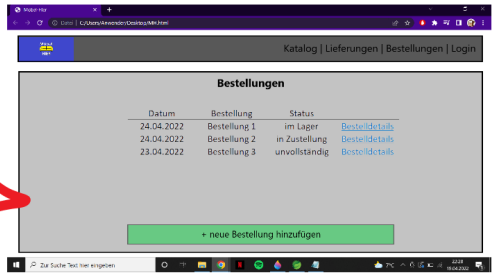


Start

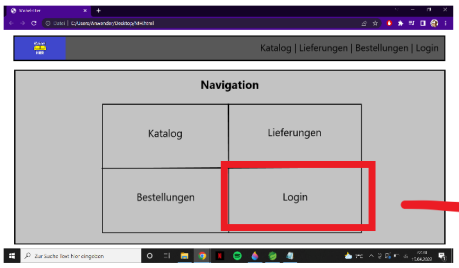
Lieferungen



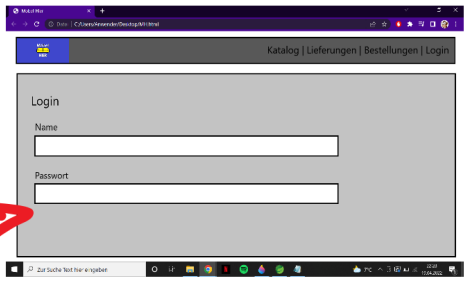
Start



Bestellungen



Start



Login

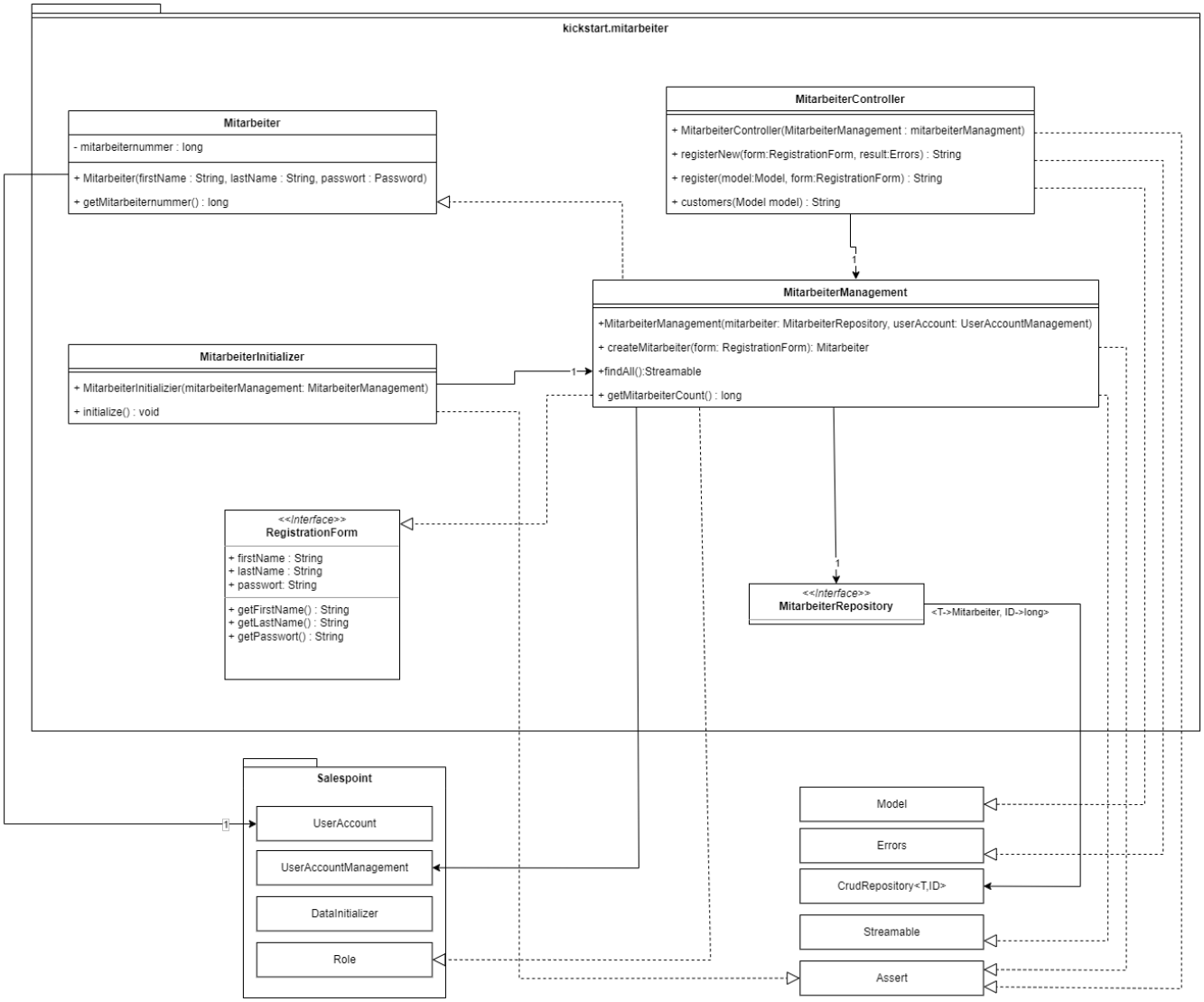
6.4. Verwendung externer Frameworks

Externes Package	Verwendet von (Klasse der eigenen Anwendung)
salespointframework.catalog	Bestellung.bestellungsController Moebel.bundle Moebel.moebel Moebel.Initializer
salespointframework.core	Bestellung.bestellungsController Moebel.bundle Kunde.kunde Kunde.kundenInitializer Moebel.Initializer
salespointframework.inventory	Moebel.Controller
salespointframework.order	Bestellung.bestellungsController
salespointframework.payment	Bestellung.bestellungsController
salespointframework.quantity	Bestellung.bestellungsController
salespointframework.userAccount	Bestellung.bestelungsController Kunde.kunde Kunde.kundenManagement Kunde.kundenRepository
springframework.data	Kunde.kundenManagement Kunde.kundenRepository Moebel.warenkatalog
springframework.stereotype	Bestellung.bestellungsController Kunde.kundenController Kunde.kundenInitializer Kunde.kundenManagement Moebel.moebelController
springframework.transaction	Kunde.kundenManagement
springframework.ui	Bestellung.bestellungsController Kunde.kundenController Moebel.moebelController
springframework.util	Bestellung.bestellungsController Kunde.kundenController Kunde.kundenInitializer Kunde.kundenManagement Moebel.moebelInitializer
springframework.validation	Kunde.kundenController Kunde.kundenForm
springframework.web	Bestellung.bestellungsController Kunde.kundenController Moebel.moebelController

7. Bausteinsicht

- Entwurfsklassendiagramme der einzelnen Packages

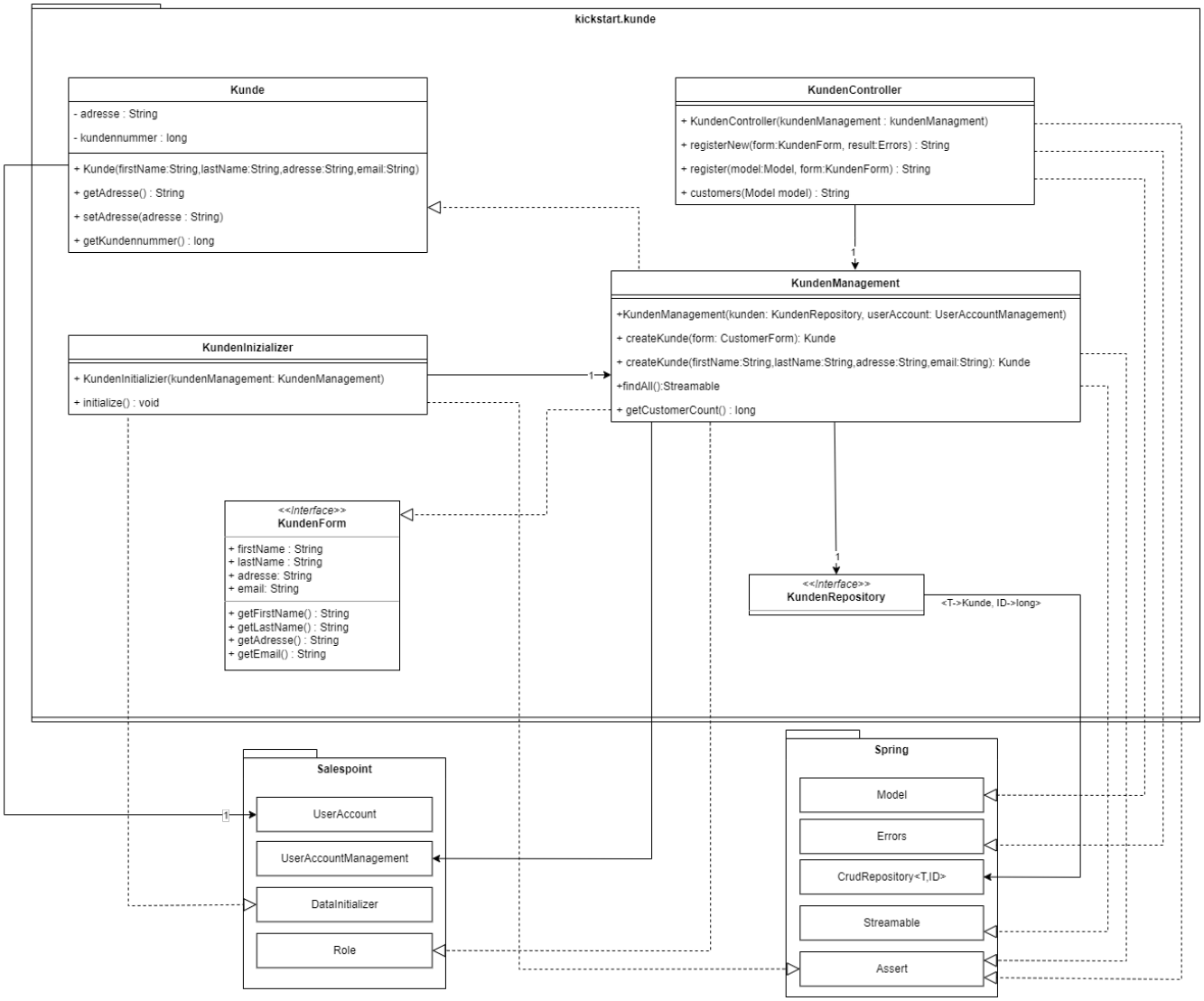
7.1. Mitarbeiter



Klasse/Enumeration	Beschreibung
Mitarbeiter	Eine Klasse zur Definition von Mitarbeitern, die Klasse UserAccount wird um einen Wert erweitert.
MitarbeiterInitializer	Eine Initializer-Klasse zum Anlegen ersten Mitarbeiterdaten
RegistrationForm	Ein Interface zur Übermittelung und Verarbeitung von Kundendaten
MitarbeiterRepository	Ein Interface zur Bereitstellung von Funktionen zur Datenbankmanipulation

Klasse/Enumeration	Beschreibung
MitarbeiterController	Eine Klasse zum Beantworten von HTML-Anfragen zu den Mitarbeitern
MitarbeiterManagement	Eine Klasse zum Anlegen und Anzeigen von Mitarbeiterdaten

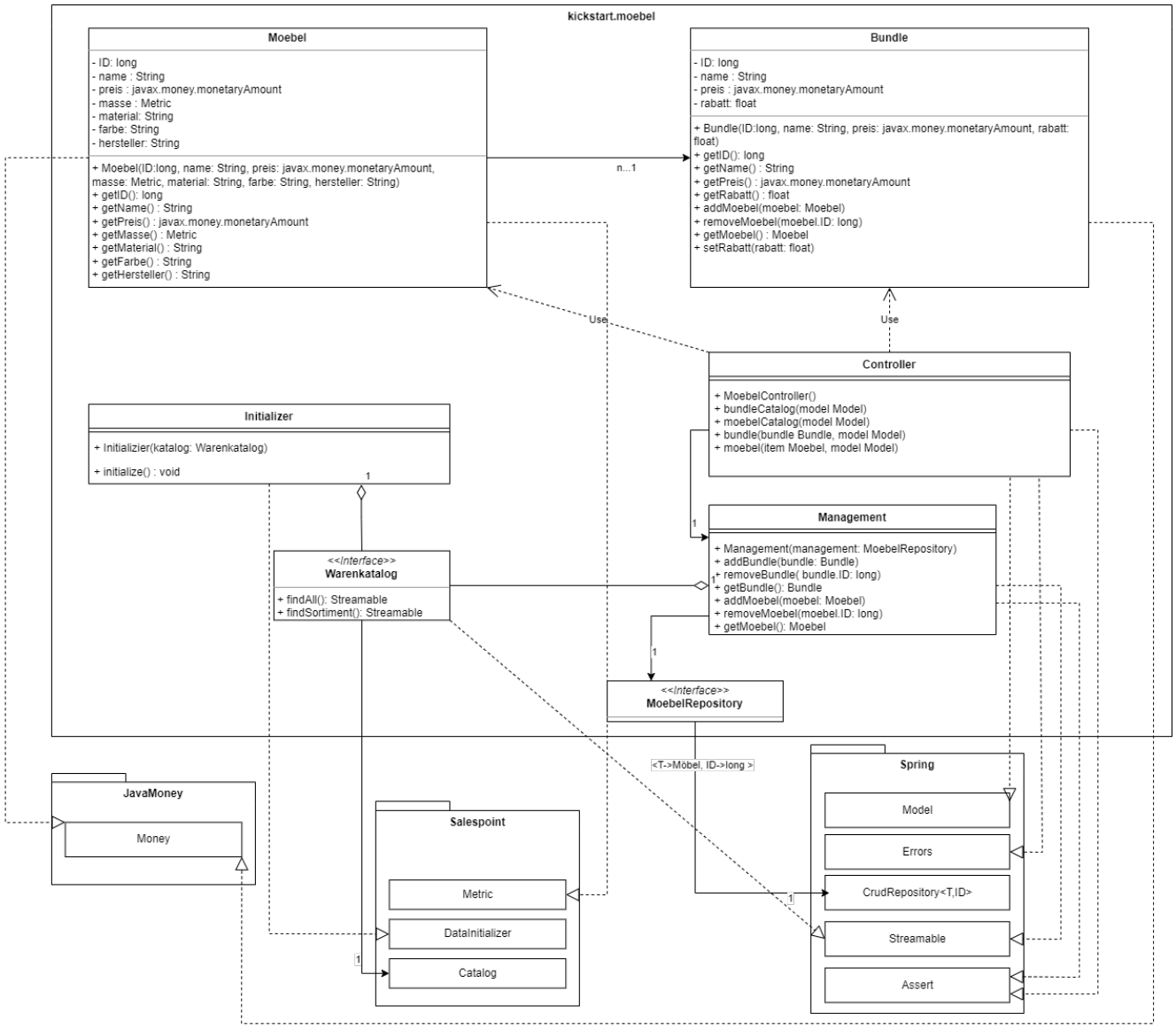
7.2. Kunden



Klasse/Enumeration	Beschreibungg
Kunde	Eine Klasse zur Definition von Kunden, die Klasse UserAccount wird mit Parametern und Funktionen erweitert
KundenInitializer	Eine Klasse um zum Start der Anwendung erste Kundendaten zu speichern
KundenController	Eine Controller-Klasse zum Anlegen und Anzeigen von Kundendaten

Klasse/Enumeration	Beschreibungg
KundenManagement	Eine Klasse zur Datenbankmanipulation und zum suchen in dieser
KundenRepository	Ein Interface zur Bereitstellung von Funktionen für die Datenbankmanipulation
KundenForm	Ein Interface zur Übermittlung und Verarbeitung von Kundendaten

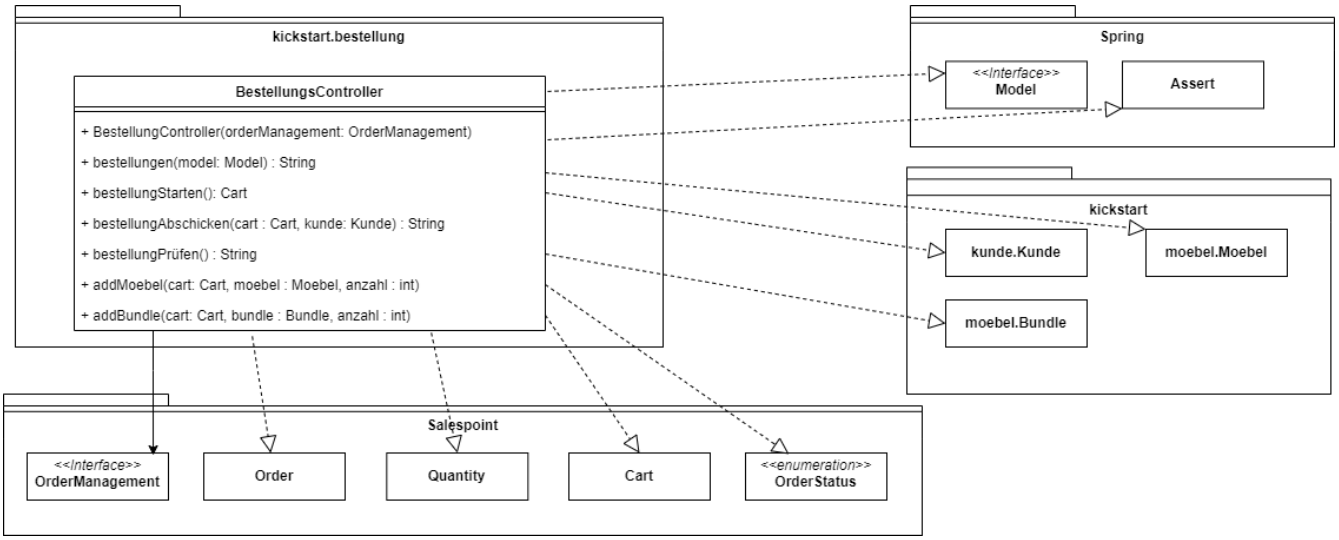
7.3. Möbel



Klasse/Enumeration	Beschreibung
Moebel	Eine Klasse zur Definition der Möbeldaten. Es ist eine Erweiterung der Product-Klasse
Bundle	Eine Klasse zur Definition der Bundledaten. Es ist eine Erweiterung der Product-Klasse

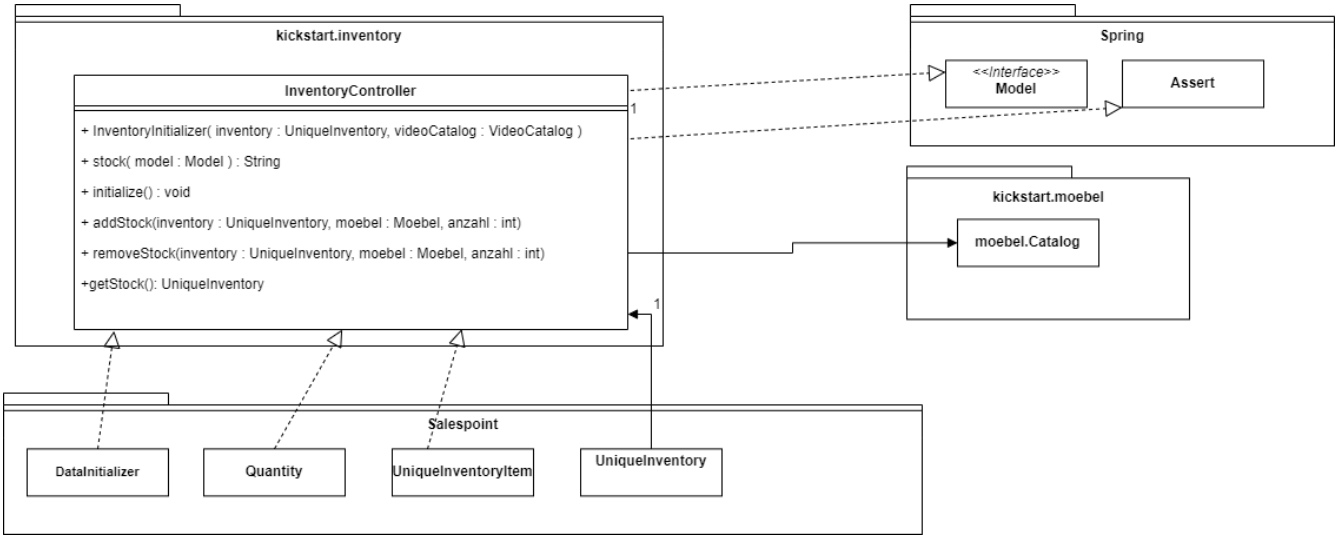
Klasse/Enumeration	Beschreibung
Controller	Ein Spring-Controller zum Anzeigen von Möbeldaten und Bundledaten
MoebelManagement	Eine Klasse zur Verarbeitung von Aufgaben der Möbels/Bundlespeicherung und -verwaltung
Initializer	Eine Klasse zur Speicherung erster Möbel, beim Start der Anwendun
MoebelRepository	Ein Repository-Interface zur Datenbankmanipulation
Warenkatalog	Ein Interface zur Erstellung eines Katalogs

7.4. Bestellung



Klasse/Enumeration	Beschreibung
BestellungsController	Ein Spring MVC Controller, um Bestellungen zu starten, Produkte hinzuzufügen und die Bestellung zu speichern.

7.5. Inventory



Klasse/Enumeration	Beschreibung
InventoryController	Ein Spring MVC Controller, um die Aufnahme und Entnahme von Bestand aus dem Inventar zu verwalten.

7.6. Rückverfolgbarkeit zwischen Analyse- und Entwurfsmodell

*Die folgende Tabelle zeigt die Rückverfolgbarkeit zwischen Entwurfs- und Analysemodell. Falls eine Klasse aus einem externen Framework im Entwurfsmodell eine Klasse des Analysemodells ersetzt, wird die Art der Verwendung dieser externen Klasse in der Spalte **Art der Verwendung** mithilfe der folgenden Begriffe definiert:*

- Inheritance/Interface-Implementation
- Class Attribute
- Method Parameter

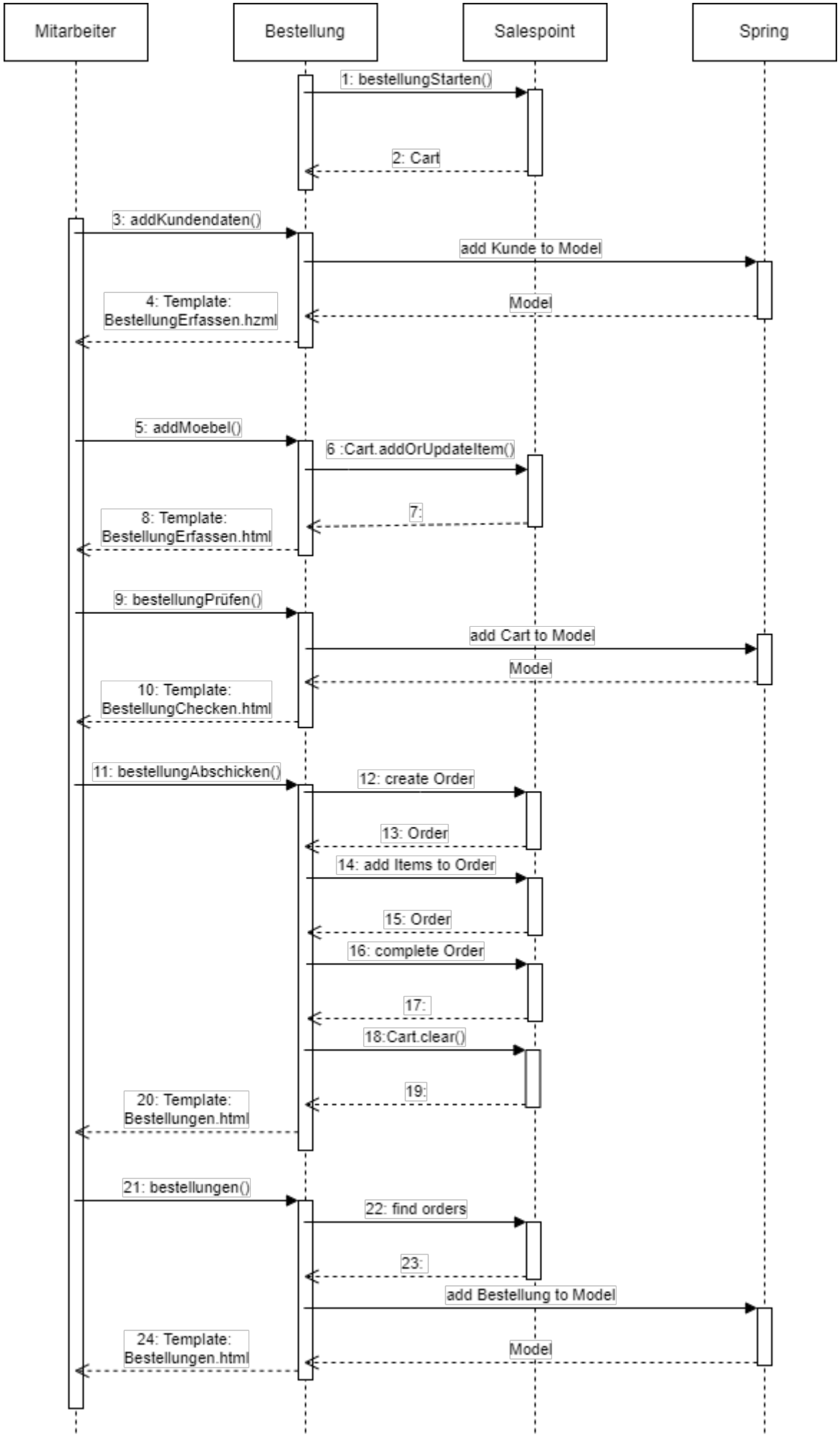
Klasse/Enumeration (Analysemodell)	Klasse/Enumeration (Entwurfsmodell)	Art der Verwendung
Kunde	Kunde extends salespoint.UserAccount	
-	KundenController	Klassenattribut
-	KundenManagement	Klassenattribut
-	KundenRepositiry	Interface
-	KundenForm	Interface
-	KundenInitializer	
Mitarbeiter	Mitarbeiter extends salespoint.UserAccount	
-	MitarbeiterController	Klassenattribut

Klasse/Enumeration (Analysemodell)	Klasse/Enumeration (Entwurfsmodell)	Art der Verwendung
-	MitarbeiterManagement	Klassenattribut
-	MitarbeiterRepository	Interface
-	RegistrationForm	Interface
-	MitarbeiterInitializer	
Möbel(komponente)	Moebel extends salespoint.Product	
Möbel-Bundle	Bundle	
-	moebel.Controller	Klassenattribut
-	moebel.Management	Klassenattribut
-	moebel.Initializer	
-	MoebelRepository	Interface
-	Warenkatalog	Interface
Bestellung	salespoint.Order	
-	BestellungsController	Klassenattribut
Lieferant	lieferung.Lieferant	
LKW	lieferung.LKW	
-	lieferung.Lieferung	
-	lieferung.Controller	KlassenAttribut
-	lieferung.Management	KlassenAttribut
-	LieferRepositry	Interface
-	LieferForm	Interface

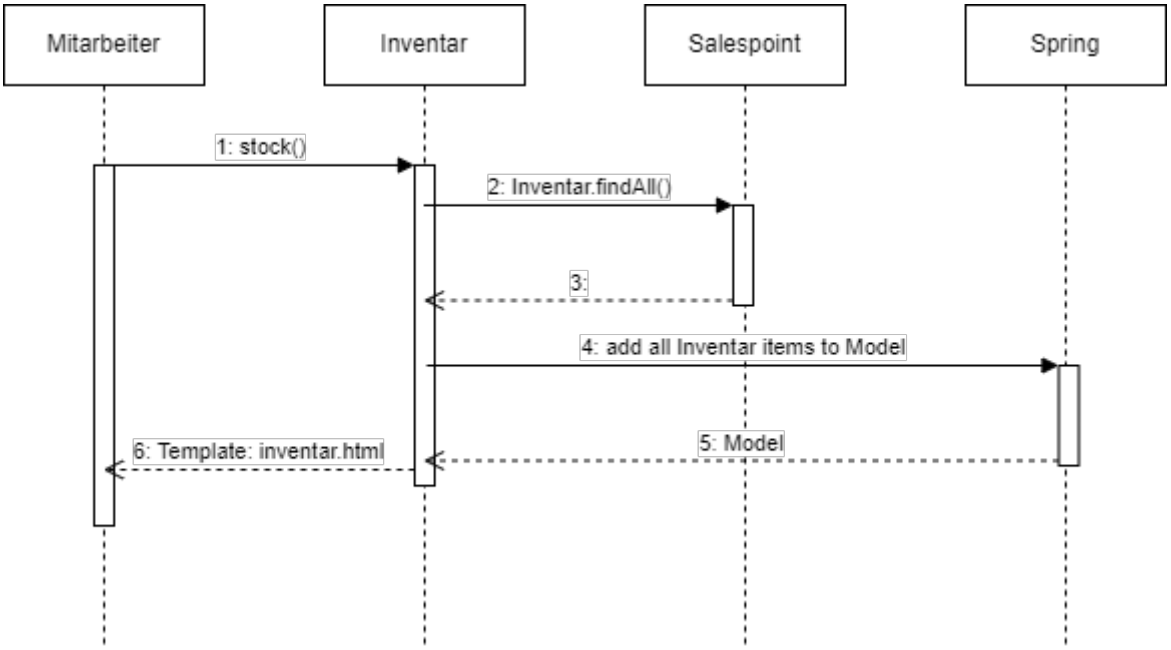
8. Laufzeitsicht

- Darstellung der Komponenteninteraktion anhand eines Sequenzdiagramms, welches die relevantesten Interaktionen darstellt.

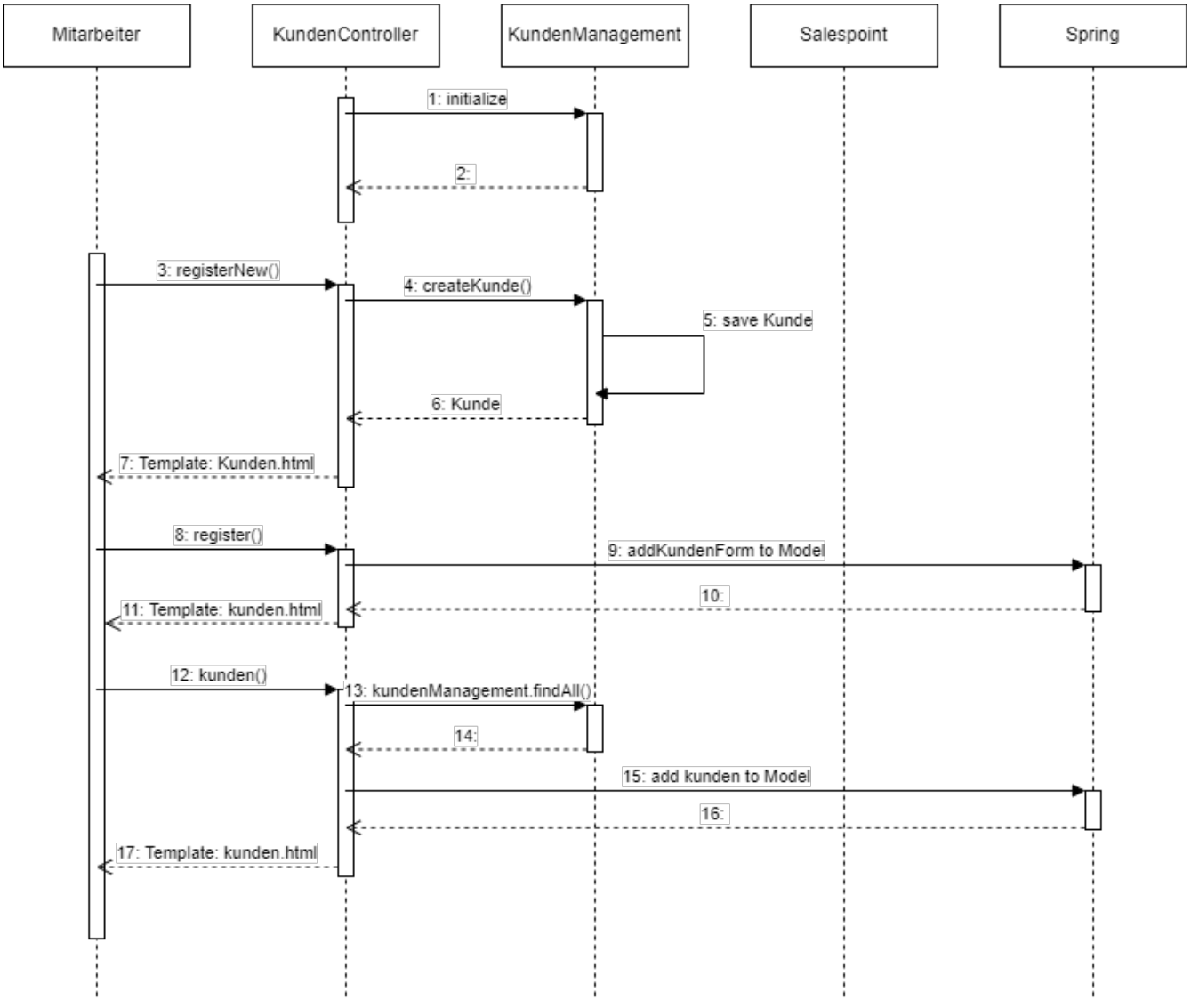
8.1. Bestellung


Mitarbeiter
Bestellung
Salespoint
Spring
1: bestellungStarten()
2: Cart
3: addKundendaten()
add Kunde to Model
Model
4: Template:
BestellungErfassen.hzml
5: addMoebel()
6 :Cart.addOrUpdateItem()
7:
8: Template:
BestellungErfassen.html
9: bestellungPrüfen()
add Cart to Model
Model
10: Template:
BestellungChecken.html
11: bestellungAbschicken()
12: create Order
13: Order
14: add Items to Order
15: Order
16: complete Order
17:
18:Cart.clear()
19:
20: Template:
Bestellungen.html
21: bestellungen()
22: find orders
23:
add Bestellung to Model
Model
24: Template:
Bestellungen.html

8.2. Inventar



8.3. Kunden



9. Technische Schulden

- Auflistung der nicht erreichten Quality Gates und der zugehörigen SonarQube Issues zum Zeitpunkt der Abgabe