

# Leistungsbeurteilung Projektarbeit

## Datenbank in Web-Auftritt einbinden

**Modul 151**

Name Vorname Klasse Name Aufsichtsperson Ich bestätige: 1. die Prüfungsbestimmungen zu kennen. 2. die Unterlagen für die Prüfung vollständig erhalten zu haben. 3. dass die vorgegebene Prüfungszeit eingehalten wurde. 4. dass die Durchführung der Prüfung ordnungsgemäss erfolgt ist. 5. <b>die Arbeit selbstständig ausgeführt zu haben.</b> 6. <b>alle externen Hilfestellungen und Quellen korrekt angegeben zu haben.</b>	Name	
	Vorname	
	Klasse	
	Name Aufsichtsperson	
	Datum und Unterschrift Teilnehmer / Teilnehmerin	

Prüfungsergebnis	Erreichte Punktzahl	
	$Note = \frac{5 * erreichte Punktzahl}{52} + 1$	
	Note gerundet auf $\frac{1}{10}$	
	Name und Vorname des Prüfungsleiters/der Prüfungsleiterin	
	Datum und Unterschrift des Prüfungsleiters/der Prüfungsleiterin	

Leistungsbeurteilung Projektarbeit Datenbank in Web-Auftritt einbinden		Modul 151
<b>Semester</b>	04	
<b>Gesamtdauer</b>	Ab 50 % des Moduls	
<b>Maximale Punktzahl</b>	52	
<b>Prüfungstyp</b>	Einzelarbeit	
<b>Bewertung</b>	Für jede dieser Teilaufgaben erhalten Sie Punkte. Diese Punkte sind jeweils bei der Aufgabe angegeben.	
<b>Erlaubte Unterlagen</b>	<ul style="list-style-type: none"> <li>_ Lehrmittel Modul</li> <li>_ Literatur aus Recherche</li> </ul>	
<b>Form der Prüfung</b>	<ul style="list-style-type: none"> <li>_ Projektarbeit</li> </ul>	
<b>Arbeitsplatz</b>	<ul style="list-style-type: none"> <li>_ Persönlicher PC mit Windows/Linux</li> <li>_ IntelliJ + Java</li> <li>_ MySQL-Server</li> </ul>	
<b>Organisation</b>	Bewertete Projektarbeit, Einzelarbeit, teilweise während der Unterrichtszeit und teilweise als Hausaufgabe zu erledigen.	
<b>Abzugebende Dokumente</b>	<ul style="list-style-type: none"> <li>_ dieses Dokument, unterschrieben</li> </ul>	
<b>Abzugebende Dateien</b>	Abzugebende Dateien (in das Verzeichnis mp151 auf dem Desktop): <ul style="list-style-type: none"> <li>_ Dokumentation (als PDF)</li> <li>_ IntelliJ-Projekt</li> </ul>	
<b>Abgabemodalitäten</b>	Art und Weise der Abgabe und der Zeitpunkt werden durch die Prüfungsleitung vorab festgelegt und kommuniziert. Nicht nach Vorgabe abgegebene Arbeiten werden grundsätzlich mit der Note 1 bewertet.	

---

# 1 Einführung

## 1.1 Ziel dieser Aufgabe

Durchführung eines grösseren zusammenhängenden Projekts als LB.

## 1.2 Aufgabenstellung und allgemeine Vorgehensweise

In diesem Projekt erstellen Sie einen Datenbankgestützten Online-Shop für eine Schreibwarenhandlung. Bewertet wird die fertige Lösung inklusive Code und Dokumentation.

## 1.3 Spielregeln

Diese Arbeit wird der zweiten Hälfte des Moduls während der Unterrichtszeit und teilweise als Hausaufgabe durchgeführt. Alle Lernenden müssen ein eigenes Projekt abgeben. Selbstverständlich dürfen Sie sich mit Ihren Mitlernenden beraten. Ebenfalls dürfen Sie Hilfestellungen und Ideen von anderen Personen und aus dem Internet beziehen. Auch der Dozent steht Ihnen per Mail als Ansprechpartner zur Verfügung.

**Achtung:** Jegliche Hilfestellung muss von Ihnen klar deklariert werden. Wenn festgestellt wird, dass entsprechende Angaben fehlen oder die Arbeit nicht durch Sie selbst erstellt worden ist, wird die ganze Arbeit als „nicht gemacht“ mit der Note 1 bewertet!

## 1.4 Zeitlicher Ablauf und Abgabe der Arbeit

Sie können sofort mit der Arbeit beginnen, sobald Sie die Aufgabenstellung erhalten haben. Diese wird im Lernraum der Klasse aufgeschaltet. Die Abgabe der Arbeit erfolgt wie mit der Prüfungsleitung vereinbart spätestens bei Unterrichtsende des letzten Blocks des Moduls. Frühere Abgaben sind natürlich erlaubt. Verspätete Abgaben werden in jedem Fall als „nicht abgegeben“ mit der Note 1 gewertet.

Für die Abgabe verlangt werden:

- ein importierbares Projekt (nicht den ganzen Workspace).
- das SQL-Skript zur Einrichtung der Datenbank (inklusive notwendiger Testdaten etc.).
- die Projektdokumentation in einem PDF-Dokument. Dieses enthält wie ein Betriebshandbuch die Anleitung zur Inbetriebnahme (falls Sie spezielle Anforderungen hierzu haben (z.B. Logindaten, zusätzliches Setup etc.) sowie alle Hilfestellungen.

## 1.5 Bewertet werden die folgenden Punkte:

Jede Position ergibt maximal 2 Punkte:

- Nicht erfüllt: 0 Punkte; teilweise erfüllt: 1 Punkt; vollständig erfüllt: 2 Punkte
- Die Note wird linear berechnet:  $1 + \text{Punkte\_Erreicht} / \text{Punkte\_Gesamt} * 5$

### 1.5.1 Allgemeine Kriterien

**16 Pt**

- |  |      |
|--|------|
| – Arbeit nach Vorgabe abgegeben, vollständig und importierbar.                             | 2 Pt |
| – Das Projekt hat keine Compiler- bzw. Syntaxfehler.                                       | 2 Pt |
| – Korrekte und vollständige Quellcode-Dokumentation (z.B. JavaDoc): (ausser getter/setter) | 2 Pt |
| – Applikation stürzt nicht ab, es treten keine Exceptions auf.                             | 2 Pt |
| – Gesamteindruck des Codes (Darstellung, Komplexität, Coding-Konventionen eingehalten)     | 2 Pt |
| – Optischer Gesamteindruck der Web-Applikation   | 2 Pt |
| – Datenbankskript vorhanden und vollständig (inkl. Testdaten und Fremdschlüssel)           | 2 Pt |
| – Projektdokumentation als PDF vorhanden und sinnvoll                                      | 2 Pt |

### 1.5.2 Einstiegsseite 8 Pt

- ☐ Layout vollständig und korrekt angezeigt 2 Pt
- ☐ Auswahl des Artikels funktioniert und verzweigt auf Artikelseite 2 Pt
- ☐ Bestellung wird vollständig angezeigt 2 Pt
- ☐ Preise der Bestellpositionen sowie Gesamtpreis werden korrekt berechnet und angezeigt 2 Pt

### 1.5.3 Artikelseite 6 Pt

- ☐ Layout vollständig und korrekt angezeigt inkl. Auswahl der möglichen Inhalte 2 Pt
- ☐ Auswahl der Inhalte wird übernommen 2 Pt
- ☐ Abbrechen ohne Datenübernahme ist möglich 2 Pt

### 1.5.4 Kundeninfoseite 6 Pt

- ☐ Layout vollständig und korrekt angezeigt 2 Pt
- ☐ Kundendaten können eingetragen werden, Daten werden übernommen 2 Pt
- ☐ Navigation gemäss Aufgabenstellung implementiert 2 Pt

### 1.5.5 Bestätigungsseite 6 Pt

- ☐ Layout vollständig und korrekt angezeigt 2 Pt
- ☐ Alle vorgesehenen Daten werden angezeigt und gespeichert 2 Pt
- ☐ Navigation gemäss Aufgabenstellung implementiert 2 Pt

### 1.5.6 Dankesseite 4 Pt

- ☐ Layout vollständig und korrekt angezeigt 2 Pt
- ☐ Session terminiert 2 Pt

### 1.5.7 Produktadministrationsseite 6 Pt

- ☐ Zugriff nur über Administrationslogin möglich 2 Pt
- ☐ Inhalte können persistent hinzugefügt und gelöscht werden 2 Pt
- ☐ Zuordnung Artikel und Inhalte ist änderbar 2 Pt

Erreichte Punktzahl Max. 52 Punkte	1.5.1	1.5.2	1.5.3	1.5.4	1.5.5	1.5.6	1.5.7	Total
---------------------------------------	-------	-------	-------	-------	-------	-------	-------	-------

## 2 „HappyWriter“ – Konzepte und Datenmodell

Lesen Sie dieses Dokument genau durch. Es gibt Ihnen eine Einführung in die Idee und die Anforderungen des Projekts und zeigt mögliche Lösungsansätze auf.

### 2.1 Projekt Grundidee

Die Papeterie „HappyWriter“ wünscht einen Online-Shop für Schreibwaren. Sie erstellen eine Vorab-Version mit einer begrenzten Anzahl Artikel. Wenn der Shop erfolgreich ist, soll er später im grossen Stil ausgebaut werden. Dieser Ausbau ist nicht Bestandteil der vorliegenden Projektarbeit.

Damit Sie nicht alles von Grund auf programmieren müssen, setzen Sie ein etabliertes Webframework ein. **Das zu verwendende Framework legt Ihre Lehrperson fest.**

### 2.2 Konzepte

Folgende Informationen sind zu modellieren:

- Artikel: im Moment werden nur Stiftschachteln aus Holz und Etuis aus Plastik verkauft.
- Inhalte: Die Artikel können leer oder mit Inhalt bestellt werden, wobei nicht jeder Inhalt für Schachtel und Etui möglich ist.
- Für einen Verkauf notwendig sind:
  - Bestellpositionen (was wird verkauft)
  - Verkaufsinformationen

Die folgenden Abschnitte beschreiben einen möglichen fachlichen Aufbau des Webshops:



### 2.3 Datenhaltung

Dieser Abschnitt enthält einen Lösungsvorschlag für einen Webshop basierend auf einem relationalem DBMS und OR-Mapping. Die im Klassendiagramm benutzten Klassen für Collections sind vom tatsächlich eingesetztem Framework abhängig.


#### 2.3.1 Artikel und Inhalt

Hier sehen Sie die Klassen *Artikel* und *Inhalt* mit geplanten Methoden (es sind nur die wichtigsten aufgeführt). Besonders interessant sind die beiden *icon()* Methoden sowie in *Artikel* die Methoden im Zusammenhang mit den möglichen Inhalten zu einem Artikel. (Attribute und Operations dürfen umbenannt werden)

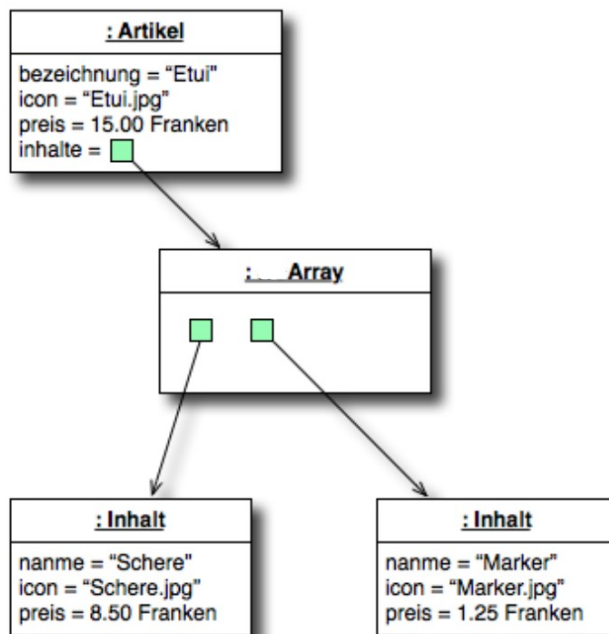
Artikel
bezeichnung : String
preis : BigDecimal
bezeichnung() : String
preis() : BigDecimal
icon() : String
inhalte() : Array<Inhalt>



Inhalt
name : String
preis : BigDecimal
name() : String
preis() : BigDecimal
icon() : String
artikel : Array<Artikel>



Das folgende Diagramm zeigt die Sicht auf die Klassen *Artikel* und *Inhalt*. *Artikel* hat eine Liste möglicher *Inhalte*, welche beispielsweise als *ArrayList* realisiert wird (dies ist der Standard bei einer to-many-Beziehung).



### 2.3.2 Bestellung und Bestellpositionen

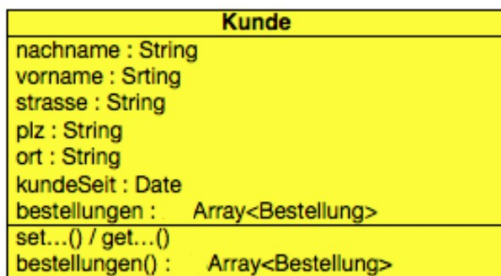
Hier sehen Sie die Klassen *Bestellung* und *Bestellposition*. Nebst den allgemeinen set- und get- Methoden soll jeweils die *toString()*-Methode überschrieben werden. Beide Klassen haben zudem eine Methode, die den Preis berechnet. Beide Klassen enthalten Methoden zum Verwalten der Listen von Bestellpositionen resp. bestellten Inhalten (auch das sind Standardmethoden für Objektbeziehungen). (Attribute und Operationen dürfen umbenannt werden)

Bestellung
datum : Date
kunde : Kunde
bestellPositionen : Array<BestellPosition>
datum() : Date
preis() : BigDecimal
kunde() : Kunde
bestellPositionen() : Array<BestellPosition>
toString() : String

BestellPosition
artikel : Artikel
bestellung : Bestellung
artikel() : Artikel
bestellung() : Bestellung
preis() : BigDecimal
icon() : String
inhalte : Array<Inhalt>

### 2.3.3 Kunde

Die Klasse *Kunde* enthält folgende Informationen:



### 2.3.4 Gesamtansicht auf eine Bestellung

So könnte eine komplette Bestellung aussehen. Die Beziehungen zwischen Objekten sind ganz "normale" Beziehungen: to-one Beziehungen werden als Instanzvariable im JavaCode erstellt (z.B. Bestellung -> Kunde), to-many Beziehungen sind Arrays (z.B. Bestellung -> Bestellpositionen).

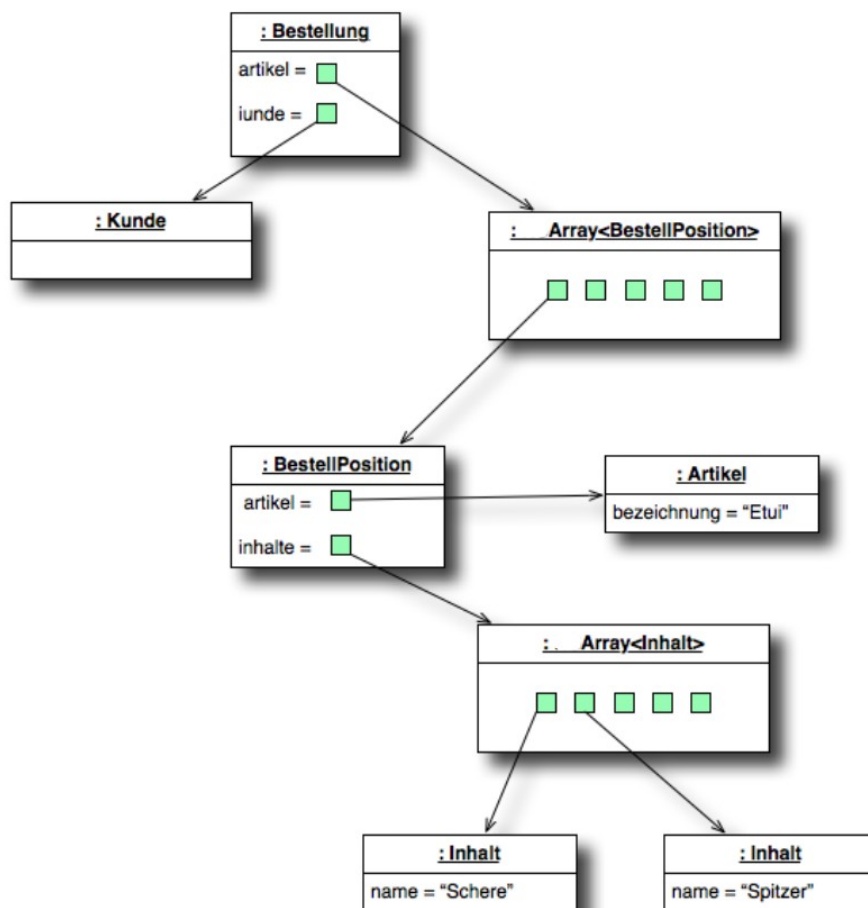


Illustration 1: Datenmodell

### 2.3.5 ERD und Klassendiagramm

Im nächsten Diagramm sehen Sie ein ERD und ein (vereinfachtes) Klassendiagramm. Im Klassendiagramm sind nicht alle Methoden eingetragen. ...set/get... steht als Platzhalter für alle notwendigen Zugriffsmethoden.

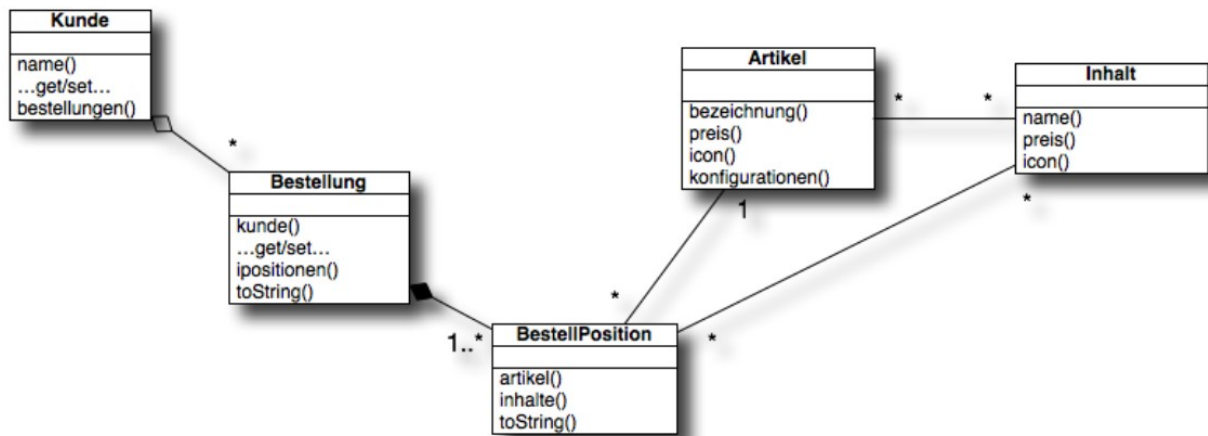


Illustration 2: Klassendiagramm

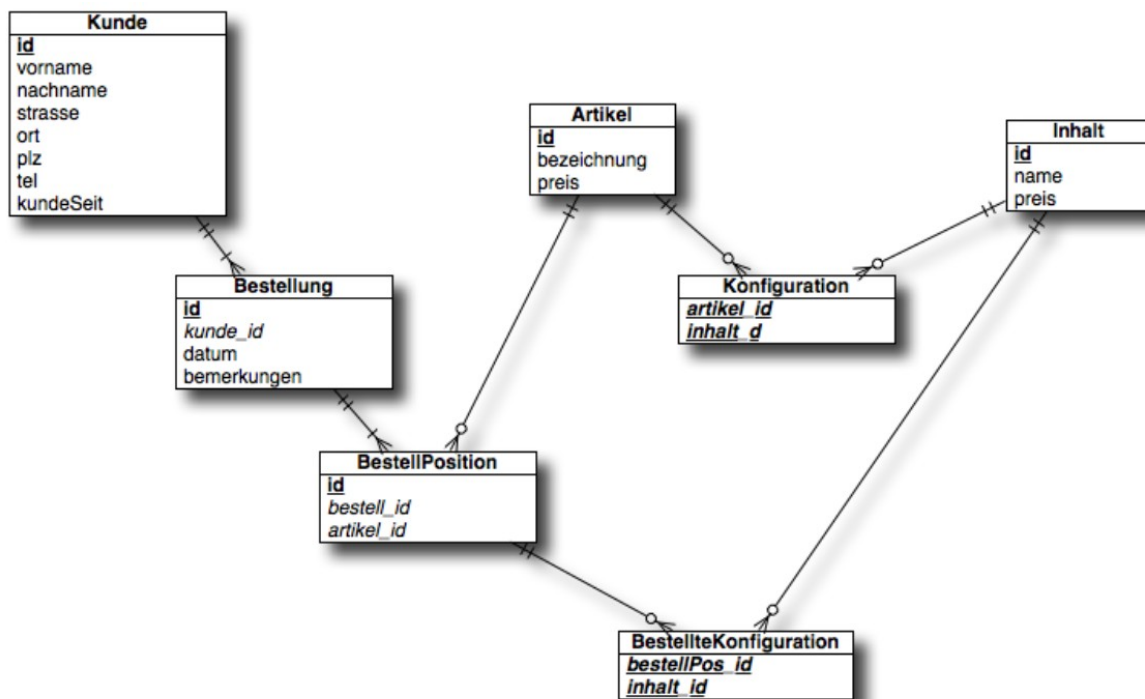


Illustration 3: Normalisiertes ERD

## 2.4 Storyboard und User Interface

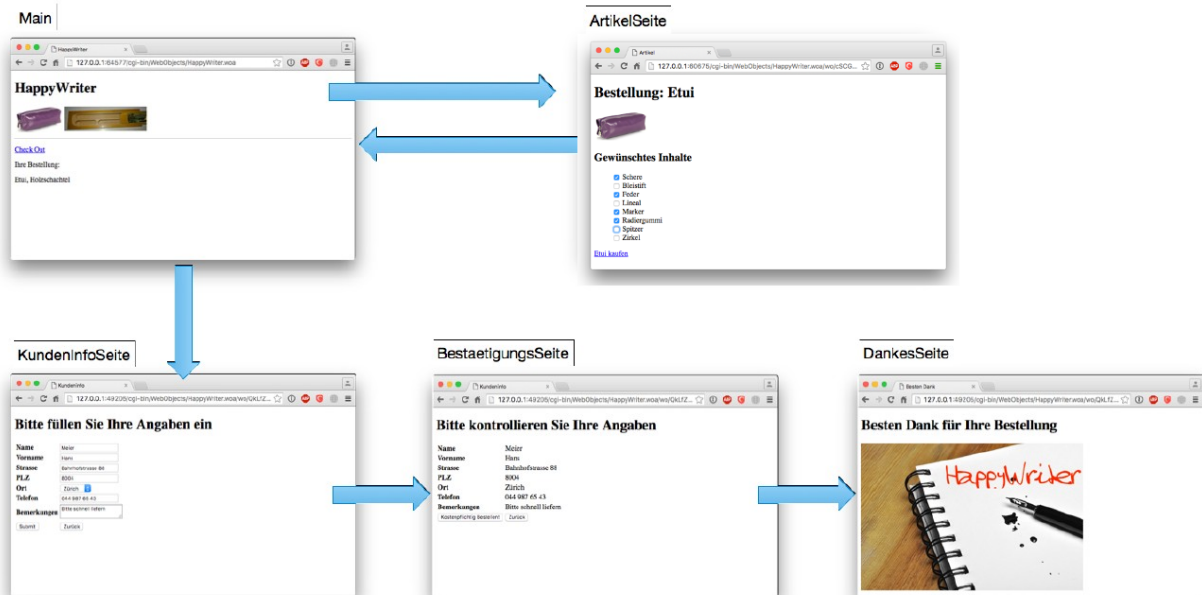
Das Storyboard (Illustration) zeigt die grundsätzlichen Abläufe des Webshops: Der Benutzer kann ein Etui oder eine Holzschachtel auswählen und den gewünschten Inhalt bestimmen. Auf der Startseite wird daraufhin ein Warenkorb mit den gekauften Gegenständen und dem Preis angezeigt.

Für den Einkauf müssen die persönlichen Daten eingegeben, verifiziert werden. Abschliessend kann der Artikel kostenpflichtig gekauft werden. **An jeder Stelle des Vorgangs soll ein Abbruch möglich sein.**

Die Abbildungen sind bewusst eine ganz triviale Darstellung des Webshops. **Erst nachdem Sie die Funktionalität implementiert haben gestalten Sie Ihre Webseite ansprechend** in einem einheitlichen Corporate Design mit einem CSS-Framework Ihrer Wahl (z.B. Bootstrap o.ä.). Für das Design werden maximal 2 Punkte gegeben, entsprechend halten Sie den Aufwand gering.

Zusätzlich erstellen Sie eine Administrationsseite nach den in Abschnitt 2.5 beschriebenen Anforderungen.





Illustra-

#### tion 4: Storyboard

Nachfolgend sind die einzelnen Seiten und Abläufe für den Webshop im Detail beschrieben.

### 2.4.1 Einstiegsseite

Hier kann der Benutzer wählen, welchen Artikel er wünscht. Zur Auswahl stehen Holzschachtel und Etui. Wenn der Benutzer auf ein Bild klickt, wird eine neue Seite verzweigt, in welcher der Benutzer dann die gewünschten Inhalte auswählen kann. Alternativ können Sie auch eine Listendarstellung der Artikel nutzen.

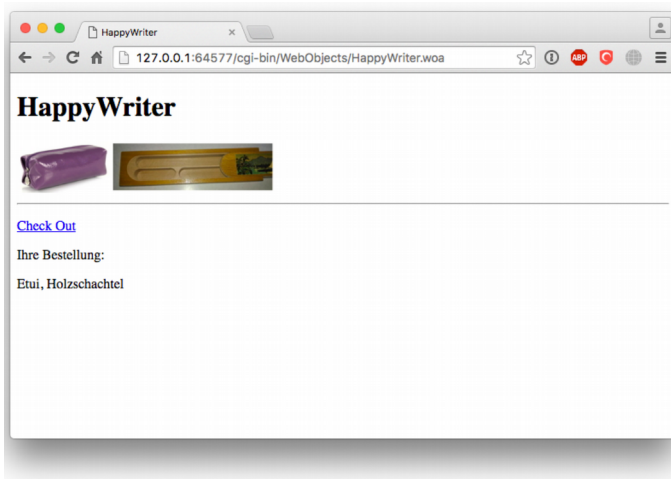


Illustration 5: Einstiegsseite

### 2.4.2 Die Detailseite zum Bestellen eines Artikels

Für beide Artikel wird die gleiche Webkomponente verwendet (siehe Illustration). Texte und Produktbilder werden je nach gewähltem Artikel aus der Datenbank geholt. Weiterhin können die gewünschten Inhalte für den Artikel per Checkbox ausgewählt werden. Der Hyperlink Check out (oder in den Warenkorb) führt zurück zur Einstiegsseite. Dabei wird für den gewählten Artikel eine Bestellposition erstellt.

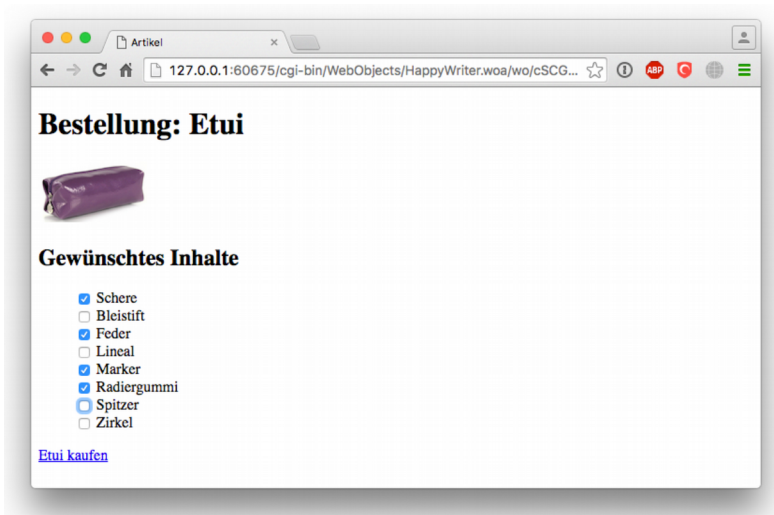
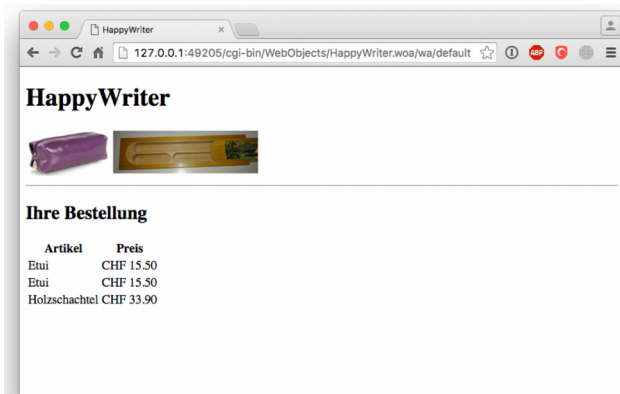


Illustration 6: Bestelldetails

### 2.4.3 Abschluss der Bestellung

Der Benutzer klickt auf der Hauptseite auf *CheckOut* und kommt zu einer Seite, wo er seine Kundendaten eingeben muss. Von da wird eine Bestätigungsseite angezeigt, wo der Benutzer seine Bestellung bestätigt. Die letzte Seite beendet die Session und bedankt sich beim Benutzer für den Kauf.



Illus-

tration 7: Bestellübersicht

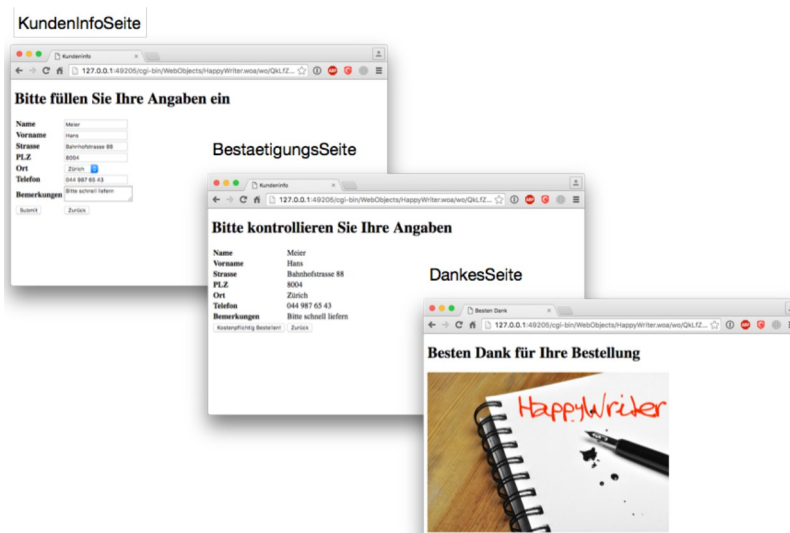


Illustration 8: Workflow

## 2.4.4 Session-Handling

In Illustration sehen Sie wie die Bestellung innerhalb der Applikation gehandhabt wird. Es bietet sich an, die Bestellung auf Session-Ebene zu verwalten. (<https://www.baeldung.com/spring-mvc-session-attributes>)

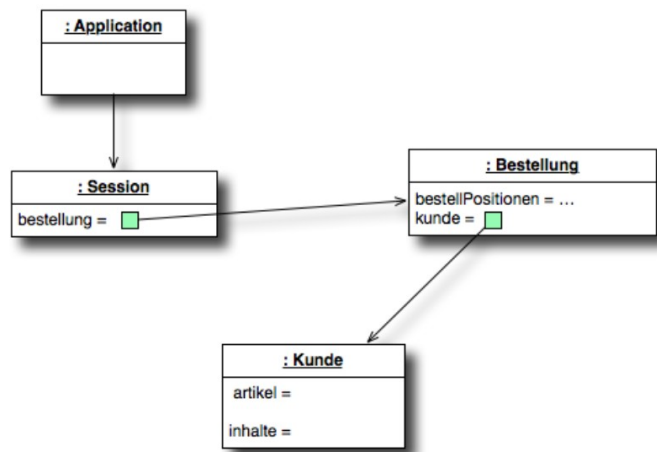


Illustration 9: Session Handling

## 2.5 Administrationsoberfläche

Erstellen Sie eine geschützte Seite zur Administration der Artikel und Inhalte. Diese Seite darf nur mit gültigem Login (admin, klapp42stuhl) erreicht werden. Benutzername und Passwort können in Admin-Seite fix kodiert werden, eine Datenbank-Unterstützung ist nicht gefordert.

Die Administrationsseite hat folgende Aufgaben:

- 1) Ändern, Hinzufügen und Löschen von Inhalten
- 2) Zuordnung der Inhalte zu den verschiedenen Artikeln.

Die Artikel selbst müssen zunächst nicht verwaltet werden. Layout und Workflow auf dieser Seite sind Ihnen überlassen.

## 2.6 Allgemeine Gedanken zur Datenspeicherung innerhalb der Applikation

Sie haben gelernt, dass Informationen auf verschiedenen Stufen verwaltet und zwischengespeichert werden können. Stellen Sie sich folgende Fragen:

- Wann sollten Sie Daten aus der Datenbank abfragen?
- Sollen Sie dieselben Daten immer wieder von der Datenbank lesen oder gibt es Daten, welche über die Laufzeit der Applikation gleich bleiben? Diese könnten Sie dann beim Applikationsstart einmalig lesen.
- Kann es sein, dass diese Daten trotzdem im Laufe der Applikationslaufzeit geändert werden müssen (denken Sie daran, dass Ihre Applikation womöglich tage oder wochenlang läuft)?

Diese und ähnliche Fragen können nur im Kontext Ihrer Applikation beantwortet werden, keine Applikation hat dieselben Daten und Bedürfnisse wie eine andere. Es gibt keine allgemein gültige Antwort, aber es ist wichtig, dass Sie sich diese Fragen stellen und für Ihre Applikation beantworten.