

**Festiva Webshop**

Projektdokumentation, Projekte der Wirtschaftsinformatik

**Entwicklung eines Bestellsystems**

Fachhochschule der Wirtschaft Paderborn

4. Semester

Pfbw115a

WS 16/17

Marcus Becker

**Entwicklung durch:**

Alina Fankhänel

Nicola Kloke

Timo Schlüter

Inhaltsverzeichnis

[1 Einleitung 5](#_Toc468212537)

[1.1 Allgemein 5](#_Toc468212538)

[1.2 Systemumgebung 5](#_Toc468212539)

[1.2.1 Software 5](#_Toc468212540)

[1.2.2 Hardware 5](#_Toc468212541)

[1.2.3 Organisation und Hilfsmittel 5](#_Toc468212542)

[1.3 Zielgruppe 5](#_Toc468212543)

[2 Architektur der Anwendung 5](#_Toc468212544)

[2.1 Allgemeine Struktur 5](#_Toc468212545)

[2.2 Datenbank-Struktur 6](#_Toc468212546)

[3 Oberfläche der Anwendung 7](#_Toc468212547)

[3.1 Planung 7](#_Toc468212548)

[3.2 Realisierung 7](#_Toc468212549)

[4 Funktionen der Anwendung 7](#_Toc468212550)

[4.1 Kundenfunktionen 8](#_Toc468212551)

[4.1.1 Planung 8](#_Toc468212552)

[4.1.2 Realisierung 9](#_Toc468212553)

[4.1.2.1 Kunde registrieren 9](#_Toc468212554)

[4.1.2.2 Kunde anmelden/abmelden 9](#_Toc468212555)

[4.1.2.3 Kundendaten ändern 10](#_Toc468212556)

[4.1.2.4 Festival suchen 10](#_Toc468212557)

[4.1.2.5 Festival anzeigen 11](#_Toc468212558)

[4.1.2.6 Artikel zu Warenkorb hinzufügen 11](#_Toc468212559)

[4.1.2.7 Warenkorb bearbeiten 12](#_Toc468212560)

[4.1.2.8 Bestellung ausführen 12](#_Toc468212561)

[4.2 Administrationsfunktionen 12](#_Toc468212562)

[4.2.1 Planung 12](#_Toc468212563)

[4.2.2 Realisierung 14](#_Toc468212564)

[4.2.2.1 Kundenverwaltung 14](#_Toc468212565)

[4.2.2.2 Kategorienverwaltung 14](#_Toc468212566)

[4.2.2.3 Festivalverwaltung 15](#_Toc468212567)

[4.2.2.4 Artikelverwaltung (unabhängige Artikel) 16](#_Toc468212568)

[4.2.3 Übergreifende Funktionen 17](#_Toc468212569)

[4.3 Technische Funktionen 17](#_Toc468212570)

[5 Testprotokoll 17](#_Toc468212571)

[6 Projektmanagement 23](#_Toc468212572)

[6.1 Soll-Ist-Vergleich 23](#_Toc468212573)

[6.1.1 Projektplan 23](#_Toc468212574)

[6.1.2 Muss/Kann-Kriterien 25](#_Toc468212575)

[6.2 Ressourcenzuordnung 26](#_Toc468212576)

[7 Fazit/Bewertung 26](#_Toc468212577)

[8 Sonstiges 26](#_Toc468212578)

[9 Eigenständigkeitserklärung 27](#_Toc468212579)

[10 Quellenverzeichnis 27](#_Toc468212580)

[11 Glossar 27](#_Toc468212581)

Abbildungsverzeichnis

[Abbildung 1 – Modell-View-Controll 5](#_Toc466844894)

[Abbildung 2 – ER-Modell (geplant) 5](#_Toc466844895)

[Abbildung 3 – Use-Case Diagramm (Besucher) 6](#_Toc466844896)

[Abbildung 4 – Use-Case Diagramm (registrierter Kunde) 7](#_Toc466844897)

[Abbildung 5 – Use-Case Diagramm (Admin – Kundenverwaltung) 7](#_Toc466844898)

[Abbildung 6 – Use-Case Diagramm (Admin - Kategorieverwaltung) 8](#_Toc466844899)

[Abbildung 7 – Use-Case Diagramm (Admin – Festivalverwaltung) 8](#_Toc466844900)

[Abbildung 8 – Carl-Steinweg Phasenmodell 14](#_Toc466844901)

**Änderungshistorie**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Verfasser | Grund/Inhalt der Änderung |
| 1.0 | 07.11.2016 | Timo Schlüter | Initialerstellung (Struktur) |

# Einleitung

## Allgemein

Das im Folgenden beschriebene Projekt wird im Rahmen des Dualen Studiums an der Fachhochschule der Wirtschaft Paderborn im Studienfach „Projekte der Wirtschaftsinformatik“ durchgeführt. Auftraggeber des Projekts ist der Dozent Marcus Becker. Dieser übernimmt zugleich die Rolle des Ansprechpartners während des Projekts.

Inhalt des Projekts ist die Entwicklung eines Bestellsystems auf Basis von Java. Konkret wurde ein Onlineshop für das fiktive Unternehmen Festiva entwickelt, welches Festivaltickets und -zubehör über eine Onlineplattform verkaufen möchte.

## Systemumgebung

### Software

Der Onlineshop wurde mit Hilfe der Entwicklungsumgebung Eclipse Neon entwickelt und in Java, Java Script, sowie HTML 5 und CSS programmiert. Als Datenbanksystem wird MySQL und als Webserver Tomcat 7.0 aus der Software XAMPP verwendet. Somit kann eine lokale Datenbank sowie ein lokaler Webserver (Tomcat 7.0) simuliert werden.

Der Benutzer benötigt Google Chrome oder Mozilla Firefox um den Onlineshop bedienen zu können.

### Hardware

Für die Entwicklung wird ein Rechner benötigt, der die Programme Eclipse Neon und XAMPP ausführen kann und installiert hat.

Als Hardware für den Endnutzer wird ein Rechner vorausgesetzt, der die Software Google Chrome sowie Mozilla Firefox ausführen kann. Zudem kann ein Smartphone verwendet werden, dass ebenfalls einen der beiden genannten Browser besitzt.

### Organisation und Hilfsmittel

Der Ansprechpartner seitens des Auftraggebers ist der Dozent Marcus Becker. Durch ihn wird auch die Projektabnahme erfolgen. Projektverantwortlicher innerhalb des Projektes ist ebenfalls Marcus Becker. Er steht uns in sämtlichen Fragen, sei es organisatorisch oder fachlich, als Ansprechpartner zur Verfügung.

Für die Planung, Organisation und Dokumentation werden MS Word, MS Excel, MS Visio, MS Project, StarUML sowie Moqups.com verwendet. Zur Versionsverwaltung und als Synchronisationstool für den Quellcode werden Git und GitHub.com verwendet.

## Zielgruppe

Zielgruppe für den Onlineshop sind Kunden von Festiva aller Altersgruppen, da eine große Bandbreite an Festivals angeboten ist. Für den Administrationsbereich sind ausgewählte Mitarbeiter von Festiva verantwortlich, die die Rolle des Admin übernehmen.

# Architektur der Anwendung

## Allgemeine Struktur

Die Anwendung wurde auf Grundlage des Model-View-Controll Konzepts entwickelt.

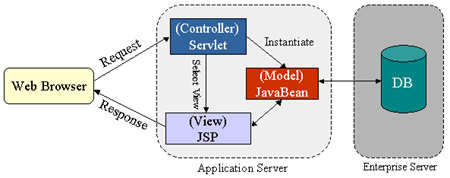


Abbildung 1 – Modell-View-Controll

Dabei wird der Quellcode entsprechend der verschiedenen Ebenen voneinander getrennt (3 Schichten Architektur).

## Datenbank-Struktur

Das ursprünglich geplante Datenmodell:

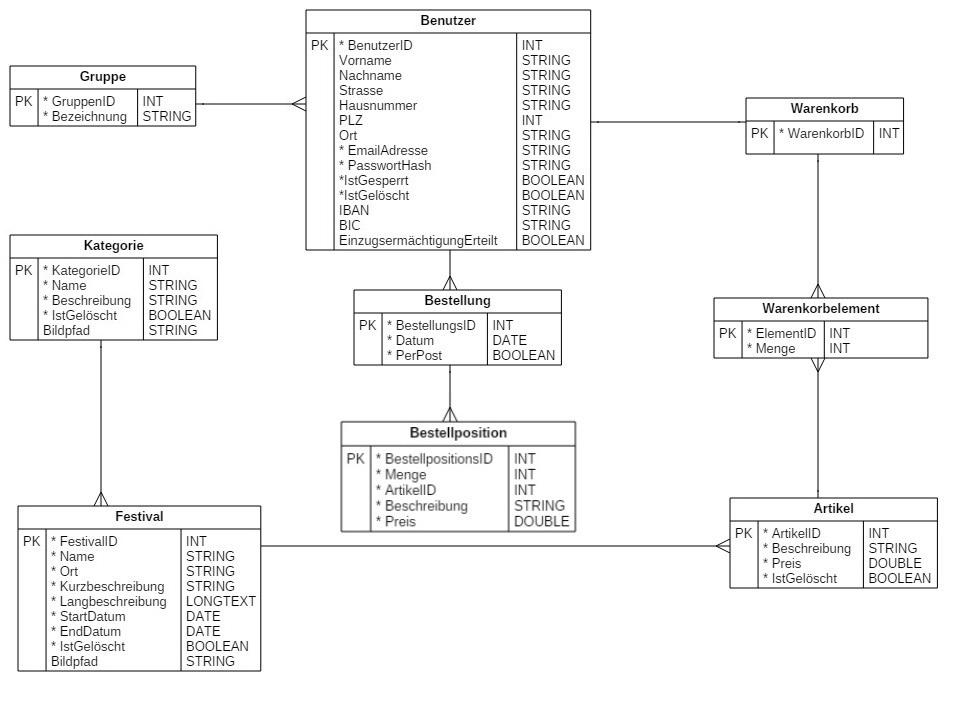


Abbildung 2 – ER-Modell (geplant)

Dieses wurde im Laufe des Projektes angepasst um weitere Kann-Kriterien, sowie die Suche nach dem maximalen Preis eines Festivals realisieren zu können.

Bei der Tabelle Benutzer wurde das Attribut „Passwortzaehler“ hinzugefügt um ein automatisches Sperren des Users nach drei falschen Passworteingaben zu realisieren.

Der Tabelle Artikel wurde das Attribut „Bildpfad“ hinzugefügt, um einem Zubehörartikel ein Bild zu geben, welches für das Kann-Kriterium des Festival-Zubehörs erforderlich war. Artikel die keinem Festival zugeordnet sind, werden im „Zubehörshop“ und in der Artikeldetailansicht mit Bild angezeigt.

Aktuelle Datenbankstruktur:

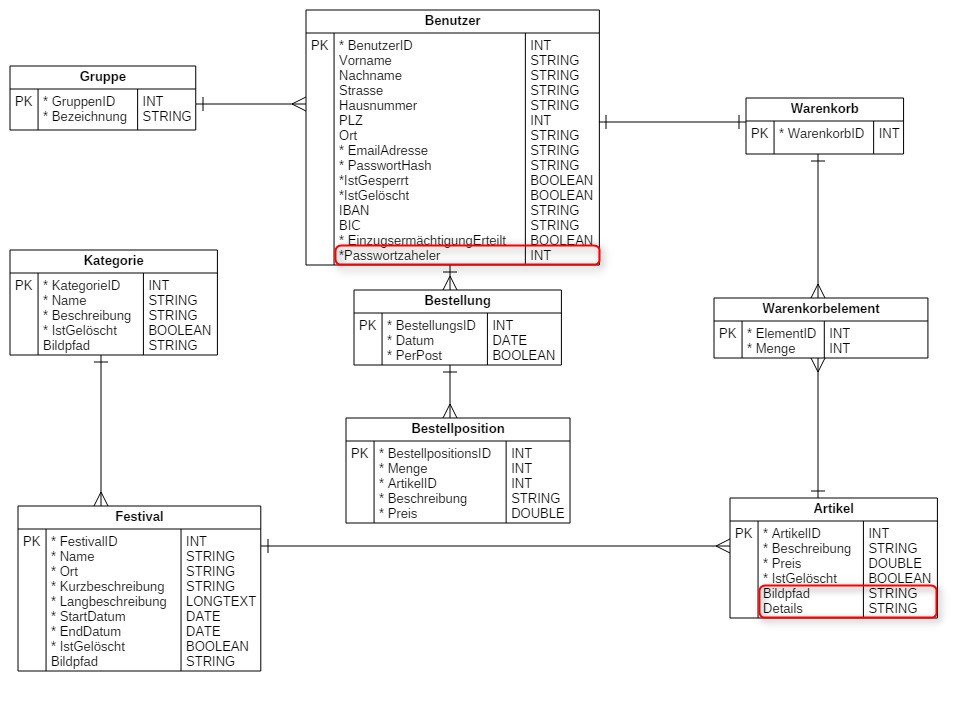


Abbildung – ER-Modell (neu)

# Oberfläche der Anwendung

## Planung

## Realisierung

# Funktionen der Anwendung

## Kundenfunktionen

### Planung

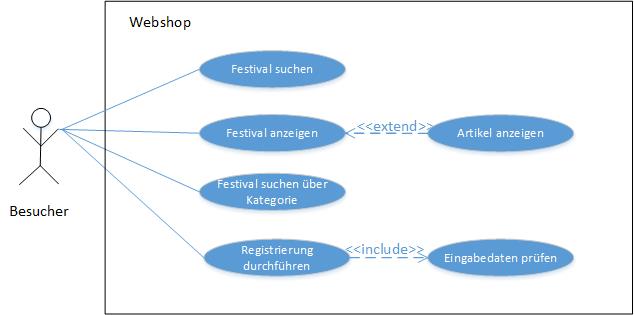


Abbildung 4 – Use-Case Diagramm (Besucher)

Ein Besucher hat im Webshop nur eine begrenzte Sicht. Er kann sich alle Festival mit ihren Artikeln ansehen und auch nach ihnen über diverse Suchfunktionen suchen. Wenn er die Möglichkeit zum Kauf eines Artikels haben möchte, muss er sich registrieren.

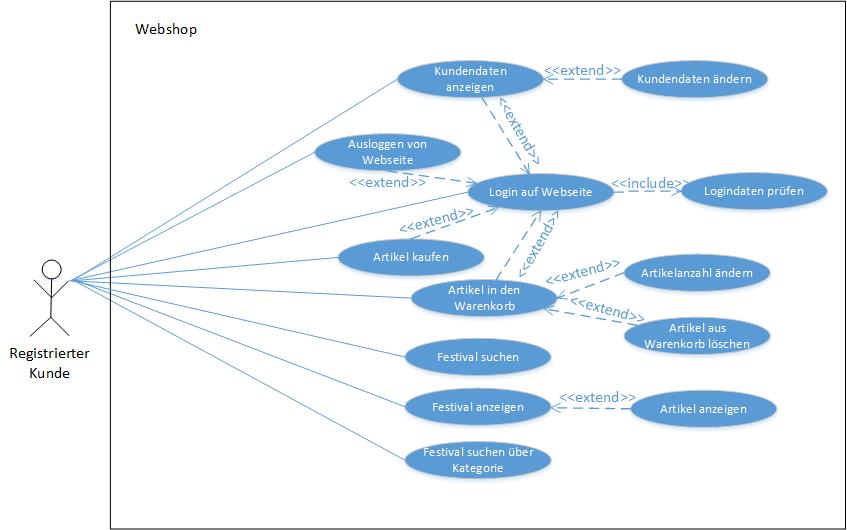


Abbildung 5 – Use-Case Diagramm (registrierter Kunde)

Zusätzlich zu den Benutzerfunktionen hat ein angemeldeter Kunde weitere Möglichkeiten. Sobald man registriert ist und sich mit seinen Benutzerdaten angemeldet hat, kann eine Bestellung durchgeführt werden. Dafür können Artikel in den Warenkorb gelegt und auch wieder gelöscht werden, sowie die gewünschte Artikelanzahl ausgewählt werden.

Der Kunde hat zu jeder Zeit die Möglichkeit seine persönlichen Daten einzusehen und zu ändern.

### Realisierung

Im Folgenden wird die konkrete Art der Realisierung an zentralen Anwendungsfällen betrachtet. Dies soll verdeutlichen, wie der Webshop insgesamt aufgebaut ist und wie die Zusammenhänge der Klassen (siehe TeilMitKlassendiagramm) gestaltet wurden.

Dieser Teil umfasst nicht die Darstellung aller Funktionalitäten, die der Webshop bietet. Es werden beispielsweise die Möglichkeiten im Bereich des Bestellens von festivalübergreifenden Artikeln oder der Einsatz von verschiedenen Versandmöglichkeiten wird nicht behandelt.

#### Kunden registrieren

**Verwendete JSPs:** k\_registrieren, k\_startseite, (k\_festivaldetails, k\_artikeldetails)

**Verwendete Servlets:** Registrierung, Login, Produktverwaltung, (Ticketverwaltung)

**Verwendete Managerklassen:** BenutzerManager, WarenkorbManager, KategorienManager, (ArtikelManager, FestivalManager)

Damit sich ein Anwender registrieren kann, werden sowohl die E-Mail-Adresse als auch das gewünschte Passwort benötigt. Beides muss zweimal eingegeben werden, damit Fehleingaben möglichst vermieden werden können. Die Prüfung auf die Kennwortrichtlinie geschieht vor dem Absenden des Formulars mit Hilfe von HTML5-Eingabefeldern und Patterns. Sobald der Anwender auf den Button „Registrieren“ klickt, wird das Servlet „Registrierung“ aufgerufen. Hier wird geprüft, ob die beiden E-Mail-Adressen und die beiden Passwörter übereinstimmen. Zusätzlich wird geprüft, ob die eingegebene E-Mail-Adresse bereits bei einem anderen Benutzer verwendet wird. Sollten die Eingaben richtig sein, werden anschließend ein neuer Kunde sowie ein leerer Warenkorb für den Kunden erstellt. Das Passwort wird dabei als Salted SHA1-Hash in der Datenbank gespeichert. Anschließend findet eine Weiterleitung zum Login-Servlet statt und der Anwender wird automatisch angemeldet. Dann wird das Servlet „Produktverwaltung“ mit dem Parameter „aktion=s\_anzeigen“ aufgerufen, sodass die aktuellen Kategorien ermittelt werden können und diese dem Kunden dann auf der Oberfläche für die Startseite (k\_startseite) in der Slideshow präsentiert werden können.

Falls der unangemeldete Kunde durch einen Klick auf den Button „In den Warenkorb“ zu der Registrierungs-Seite gelangt ist, wird er wieder zu genau dieser Seite zurückgeleitet, damit er die Warenkorb-Aktion nun durchführen kann. In diesem Fall kommen auch die JSPs, Servlets und Manager-Klassen, die oben in Klammern angegeben wurden, zum Einsatz.

Bei einem erfolgreichen Registrierungs-Vorgang erhält der Kunde in jedem Fall eine Meldung über die erfolgreiche Registrierung und wird auf die Startseite weitergeleitet.

Sollte der Anwender zwei ungleiche E-Mail-Adressen oder Passwörter eingegeben haben oder seine eingegebene E-Mail-Adresse bereits im System verwendet werden, erhält er eine entsprechende Fehlermeldung.

#### Kunden anmelden / abmelden

**Verwendete JSPs:** k\_anmelden, k\_startseite, (k\_festivaldetails, k\_artikeldetails)

**Verwendete Servlets:** Login, Logout, Produktverwaltung, (Ticketverwaltung)

**Verwendete Managerklassen:** BenutzerManager, KategorienManager, (ArtikelManager, FestivalManager,

WarenkorbManager)

Damit sich ein Kunde anmelden kann, wird die E-Mail-Adresse und das zugehörige Passwort benötigt. Wenn der Kunde diese in der Anmeldemaske (k\_anmelden) eingegeben hat und auf den Button „Anmelden“ klickt, wird das Servlet „Login“ aufgerufen. Hier werden auf Grundlage der E-Mail-Adresse die Daten des Benutzers ermittelt. Sollte kein Benutzer mit dieser E-Mail-Adresse gefunden werden oder das eingegebene Passwort nicht zu dem in der Datenbank gespeicherten passen, erhält der Benutzer eine passende Fehlermeldung. Darüber hinaus erhält der Kunde auch eine Fehlermeldung, wenn er gesperrt ist oder sein Benutzerkonto gelöscht wurde. Wenn die Eingaben korrekt waren, wird eine neue Session erstellt und der Kunde standardmäßig auf die Kundenstartseite geleitet. Dies findet analog zu dem in 4.1.2.1 beschriebenen Verfahren statt. Auch an dieser Stelle hat der Kunde die Möglichkeit, auf die Seite des Artikels zurückzukehren, wenn er unangemeldet einen Artikel in den Warenkorb legen wollte.

Beim Abmelden wird die Session des Kunden geschlossen und dieser zurück auf die Kundenstartseite geleitet.

#### Kundendaten ändern

**Verwendete JSPs:** k\_kundendaten

**Verwendete Servlets:** Benutzerdaten, Logout, Produktverwaltung

**Verwendete Managerklassen:** BenutzerManager, KategorienManager

Wenn der Kunde über „Mein Konto“ zu „Meine Daten“ navigiert, wird das Servlet „Benutzerdaten“ mit dem Parameter „aktion=anzeigen“ aufgerufen. Zu dem aktuellen Kunden werden dann die Daten ermittelt und der Kunde wird auf die passende Oberfläche (k\_kundendaten), auf der seine aktuellen persönlichen Daten angezeigt werden, weitergeleitet. Möchte der Kunde seine Daten ändern, tut er dies in der Benutzeroberfläche und klickt dann auf den Button „Änderungen speichern“. Dadurch wird das Servlet „Benutzerdaten“ mit dem Parameter „aktion=aendern“ aufgerufen. Anschließend wird im Servlet ein grober Teil zur Validierung (Der Großteil findet bereits zuvor mit HTML5-Eingabefeldern und Patterns statt) und die Verarbeitung der aktuellen Daten durchgeführt. Danach wird der Kunde wieder zur Anzeige der aktuellen Daten geleitet. Hier erhält der Kunde eine Rückmeldung. Entweder konnten die eingegebenen Daten erfolgreich gespeichert werden oder es erscheint eine Fehlermeldung, weil beispielsweise eine E-Mail-Adresse, die bereits einem anderen Kunden zugeordnet wurde, eingegeben wurde und die Änderung deswegen nicht durchgeführt werden konnte.

Zusätzlich hat der Kunde die Möglichkeit, sein Passwort zu ändern. Erforderlich dafür sind die Eingabe seines alten Passworts und eine zweimalige Eingabe des neuen, gewünschten Passworts. Mit dem Klick auf den Button „Passwort ändern“, wird das Servlet „Benutzerdaten“ mit dem Parameter „aktion=p\_aendern“ aufgerufen. Die Prüfung auf die Kennwortrichtlinie geschieht dabei wieder vor dem Absenden des Formulars. Im Servlet findet dann die Prüfung statt, ob das eingegebene alte Passwort korrekt ist und ob die beiden Eingaben für das neue Passwort gleich sind. Wenn die Eingaben korrekt sind, werden die Änderungen durchgeführt und der Kunde erhält eine entsprechende Rückmeldung. Wenn eine Prüfung einen Fehler wirft, erhält der Kunde eine passende Fehlermeldung.

Darüber hinaus hat der Kunde die Möglichkeit, sein Konto zu löschen. Mit dem Klick auf den Button „Mein Benutzerkonto löschen“ und der anschließenden Bestätigung des Pop-Ups wird das Servlet „Benutzerdaten“ mit dem Parameter „aktion=loeschen“ aufgerufen. Im Servlet wird der Kunde dann logisch gelöscht. Danach findet eine Weiterleitung zu dem Logout-Servlet statt, sodass die aktuelle Session des Kunden auch automatisch beendet wird. Der Kunde wird zum Schluss wieder auf die Kunden-Startseite geleitet. Die Anzeige der Startseite folgt dem Verfahren in 4.1.2.1

#### Festivals suchen

**Verwendete JSPs:** k\_ticketShop

**Verwendete Servlets:** Ticketverwaltung

**Verwendete Managerklassen:** FestivalManager, KategorienManager

Bei der Suche kann der Kunde verschiedene Suchkriterien (Name, Kategorie, Ort, Maximalpreis, Von- & Bis-Datum) in der Suchmaske der Oberfläche (k\_ticketShop) eingeben und miteinander kombinieren. Mit dem Klick auf den Button „Suchen“ wird das Servlet „Ticketverwaltung“ mit dem Parameter „aktion=t\_anzeigen“ aufgerufen. Die Validierung der Benutzer-Eingaben finden wieder primär vor dem Absenden des Formulars statt. Im Servlet wird nur noch geprüft, ob das Startdatum vor dem Enddatum liegt oder gleich dem Enddatum ist. Ist dem nicht so, wird die Suche nicht ausgeführt und der Kunde erhält eine passende Fehlermeldung. Sollten die Eingaben korrekt sein, werden die Festivals, die den Eingaben entsprechen, ermittelt und an die JSP zurückgegeben, sodass diese auf der Oberfläche des Ticket-Shops angezeigt werden können.

#### Festivaldetails anzeigen

**Verwendete JSPs:** k\_ticketShop, k\_festivaldetails

**Verwendete Servlets:** Ticketverwaltung

**Verwendete Managerklassen:** FestivalManager, WarenkorbManager, ArtikelManager

Hat der Kunde ein Festival ausgewählt (Klick auf den Namen oder das Bild eines Suchergebnisses aus 4.1.2.4), wird das Servlet „Ticketverwaltung“ mit dem Parameter „aktion=f\_anzeigen“ aufgerufen. Hier werden alle nötigen Daten zu dem Festival und auch (falls ein Kunde in der Session eingeloggt ist) der Warenkorbinhalt des aktuellen Kunden. Danach wird eine Liste aller verfügbaren Artikel des Festivals ermittelt.

Wird ein Maximalpreis angegeben, werden zwei Artikellisten erzeugt. Die eine Liste beinhaltet alle Artikel, die kleiner oder gleich dem Maximalpreis sind und die andere Artikelliste alle Artikel, die über dem maximalen Preis liegen.

Die JSP „k\_festivaldetails“ wird anschließend mit den ermittelten Daten aufgerufen und kann dem Anwender das Ergebnis anzeigen.

#### Artikel zum Warenkorb hinzufügen

**Verwendete JSPs:** k\_festivaldetails

**Verwendete Servlets:** Warenkorbverwaltung, Ticketverwaltung

**Verwendete Managerklassen:** WarenkorbManager, ArtikelManager, FestivalManager

Möchte der Kunde einen Artikel in den Warenkorb legen, muss er die gewünschte Menge auswählen und klickt auf den Button „In den Warenkorb“. Sollte der Anwender noch nicht eingeloggt sein, wird er zu der Anmelden-Seite weitergeleitet (siehe 4.1.2.2). Anschließend wird geprüft, ob sich der gewünschte Artikel bereits im Warenkorb befindet.

1. Ist dieses der Fall erhält der Kunde ein Pop-Up mit der Nachfrage, ob der Artikel trotzdem dem Warenkorb hinzugefügt werden soll. Wird dies bestätigt, wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=aktualisieren“ aufgerufen. Im Servlet wird anschließend der passende Artikel und das Warenkorbelement des Kunden, welches sich auf den Artikel bezieht, ermittelt. Anschließend wird die Anzahl um die gewünschte Menge erhöht und der Kunde erhält eine Rückmeldung über das Aktualisieren der Anzahl des Artikels. Sollte bei dem Vorgang die maximale Anzahl von 10 überschritten werden, wird der Vorgang abgebrochen und der Kunde erhält eine passende Fehlermeldung. Fall bekommt der Kunde entsprechendes Feedback und die Menge wird auf zehn gesetzt.

2. Wenn der Artikel sich noch nicht im Warenkorb befindet, wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=hinzufuegen“ aufgerufen. Im Servlet wird dem Warenkorb des aktuellen Kunden anschließend ein neues Warenkorbelement mit der ausgewählten Menge hinzugefügt. Anschließend erhält der Kunde eine Rückmeldung zu dem erfolgreichen Hinzufügen der Menge des Artikels.

Der Kunde wird in jedem Fall zurück zu der Detail-Seite (k\_festivaldetails) des Festivals geleitet und erhält eine Rückmeldung.

#### Warenkorb anzeigen / bearbeiten

**Verwendete JSPs:** k\_warenkorb, k\_festivaldetails

**Verwendete Servlets:** Warenkorbverwaltung, Ticketverwaltung

**Verwendete Managerklassen:** WarenkorbManager, ArtikelManager, FestivalManager

Zur Anzeige des Warenkorbs wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=anzeigen“ aufgerufen. Anschließend werden alle Warenkorbelemente des Warenkorbs des aktuell angemeldeten Kunden ermittelt. Dann findet eine Weiterleitung an die JSP „k\_warenkorb“ statt. Diese stellt dann die übermittelten Daten dar und bietet dem Kunden beispielsweise die Möglichkeit durch einen Klick auf den Artikelnamen wieder in die Detail-Informationen zu dem Artikel navigieren zu können.

Wenn ein Kunde den Warenkorb bearbeiten möchte, muss er auf der Warenkorb-Oberfläche entweder die Menge des Artikels über das Dropdown-Menü ändern oder auf den „Löschen“-Button klicken. Ändert er die Menge, wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=aendern“ aufgerufen. Dann wird bei dem betroffenen Warenkorbelement die Menge angepasst. Klickt der Kunde auf den „Löschen“-Button, wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=loeschen“ aufgerufen. Dann wird das betroffene Warenkorbelement aus dem Warenkorb des Kunden gelöscht.

In jedem Fall wird der Kunde nach einer Änderung oder Löschung wieder zurück zur Warenkorb-Oberfläche mit den aktuellen Daten (Aufruf von „Warenkorbverwaltung“ mit dem Parameter „aktion=anzeigen“) geleitet.

#### Bestellung durchführen

**Verwendete JSPs:** k\_warenkorb, k\_kasse

**Verwendete Servlets:** Warenkorbverwaltung, Bestellverwaltung

**Verwendete Managerklassen:** BestellungsManager, WarenkorbManager

Von dem Warenkorb kann der Kunde mit dem Button „Zur Kasse“ weiter navigieren. Diese Möglichkeit bietet sich dem Kunden allerdings nur, wenn er Artikel in seinem Warenkorb hat. Anderenfalls ist der Button deaktiviert.

Mit dem Klick auf den Button wird das Servlet „Warenkorbverwaltung“ mit dem Parameter „aktion=k\_anzeigen“ aufgerufen. An dieser Stelle wird geprüft, ob die Bestellung festivalübergreifende Artikel beinhaltet. In diesem Fall wird der Post-Versand festgesetzt. Andernfalls wird der kostenlose Mail-Versand als Standard vorgeschlagen und kann später noch durch den Kunden geändert werden. Zusätzlich wird geprüft, ob alle Kundendaten für den Abschluss einer Bestellung vorliegen. In einem letzten Schritt werden alle Warenkorbelemente ermittelt und alle Daten an die JSP „k\_kasse“ übergeben. Hier werden die Daten dann dargestellt und es wird beispielsweise im Fall von unvollständigen Daten ein entsprechender Hinweis eingeblendet sowie die Möglichkeit zur Änderung der Daten geboten. Der Button „Verbindlich bestellen“ ist in diesem Fall deaktiviert.

Wenn jedoch alle Daten vorliegen, der Button „Verbindlich bestellen“ verfügbar ist und der Kunde auf den Button klickt und das erscheinende Pop-Up-Fenster bestätigt, wird das Servlet „Bestellverwaltung“ mit dem Parameter „aktion=anlegen“ aufgerufen. Aus den übermittelten Daten wird eine Bestellung mit den entsprechenden Bestellpostionen angelegt. Anschließend wird das Servlet „Bestellverwaltung“ mit dem Parameter „aktion=anzeigen“ aufgerufen. Hier werden alle bisherigen Bestellungen des aktuell angemeldeten Kunden ermittelt und an die JSP „k\_bestellungen“ übergeben, um diese darzustellen.

## Administrationsfunktionen

### Planung

Ein Admin meldet sich über das Admin-Login an und erhält andere Sichten als ein Kunde.

Er hat die drei Menüpunkte Kunden-, Kategorie- und Festivalverwaltung.

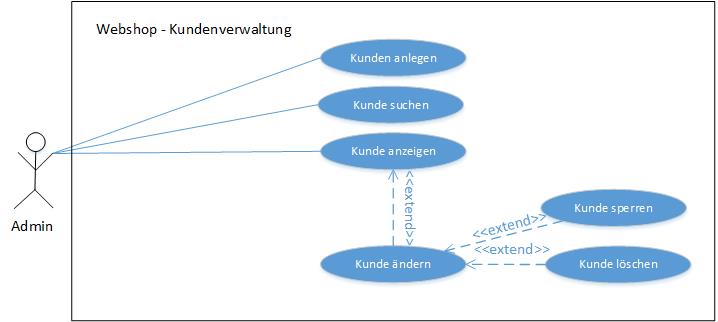


Abbildung 6 – Use-Case Diagramm (Admin – Kundenverwaltung)

Ein Admin kann einen Kunden anlegen, suchen, anzeigen und desweitern über den Änderungsmodus sperren und löschen.

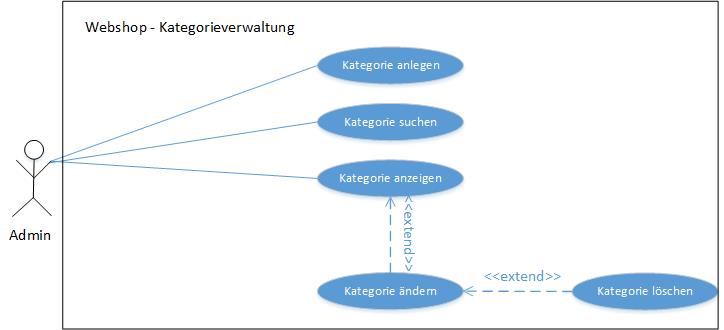


Abbildung 7 – Use-Case Diagramm (Admin - Kategorieverwaltung)

Ein Admin kann eine Kategorie anlegen, suchen, anzeigen und desweitern über den Änderungsmodus löschen.

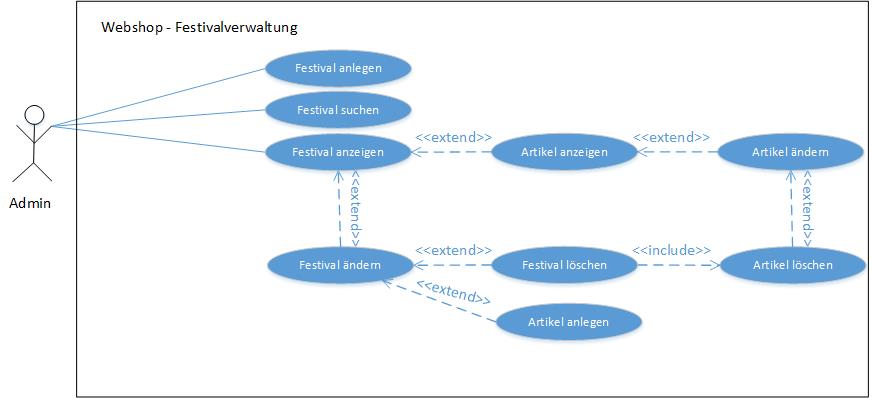


Abbildung 8 – Use-Case Diagramm (Admin – Festivalverwaltung)

Die Festivalverwaltung weist ebenfalls die Standardfunktionen zum Anlegen, Suchen, Anzeigen und Ändern und Löschen von Festivals und ihren zugehörigen Artikeln auf.

Hinzukommt, dass automatisch alle zu dem Festival zugehörigen Artikel gelöscht werden, sobald das Festival gelöscht wird.

### Realisierung

#### Kundenverwaltung

JSP: a\_kundenverwaltung, a\_kundeAendern, a\_kundeAnlegen

Servlet: Kundenverwaltung

Java-Klasse: BenutzerManager

In der Kundenverwaltung kann der Admin Kunden anlegen und Kundendaten ändern. Klickt der Admin auf den Button „Neuen Kunden anlegen“ wird das JSP „a\_kundeAnlegen“ aufgerufen. Hier kann der Admin die Daten eingeben und auf „anlegen“ klicken. Dadurch wird das Servlet „Registrierung“ aufgerufen. Es geht weiter wie in 4.1.2.1, nur, dass am Ende das JSP „a\_kundenAnlegen“ aufgerufen wird.

Sollen Kundendaten geändert werden, muss der Admin auf die ID des Kunden klicken. Dabei wird das Servlet „Kundenverwaltung“ mit dem Attribut „aendern“ aufgerufen, die wiederum das JSP „a\_kundenAendern“ aufruft. Hier werden alle Daten des Kunden angezeigt. Diese können wie in 4.1.2.3 geändert werden. Auch der Servletaufruf ist derselbe, nur, dass am Ende das JSP „a\_kundenAendern“ aufgerufen wird.

#### Kategorienverwaltung

JSP: a\_kategorieverwaltung, a\_kategorieAendern, a\_kategorieAnlegen

Servlet: Kategorienverwaltung

Java-Klasse: Kategorienmanager

In der Kategorienverwaltung kann der Admin Kategorien anlegen, ändern und löschen. Klickt der Admin auf den Button „Neue Kategorie anlegen“ wird das JSP „a\_kategorieAnlegen“ aufgerufen. Hier kann der Admin die Daten eingeben, ein Bild hochladen und auf „anlegen“ klicken. Dadurch wird das Servlet „Kategorieverwaltung“ mit dem Attribut „anlegen“ ausgeführt. Es werden die Parameter eingelesen, zwischgespeichert und auf Vollständigkeit geprüft. Sind diese nicht vollständig wird ein entsprechendes Feedback zurückgegeben. Sind die Daten vollständig, wird zunächst eine neue Kategorie ohne den Bildpfad mit der Mehtode „erstelleKategorie“ der Klasse „KategorienManager“ angelegt. Anschließend wird das Bild in den Ordner „WebContent\Bilder“ hochgeladen und die Kategorie mit der Methode „aktualisiereKategorie“ aktualisiert. Abschließend wird das JSP „a\_kategorieAnlegen“ mit einem entsprechenden Feedback aufgerufen.

Möchte der Admin eine Kategorie ändern klickt er im JSP „a\_kategorieverwaltung“ auf die ID der entsprechenden Kategorie und führt das Servlet „Kategorieverwaltung“ mit dem Attribut „aendern“ aus. Hier werden zunächst die Parameter eingelesen und es wird geprüft, ob die notwendigen Informationen eingetragen wurden. Ist dem nicht so, wird ein entsprechendes Feedback an den Kunden gegeben und nichts geändert. Liegen alle Informationen vor, wird zunächst das Bild hochgeladen und geändert – sollte ein neues Bild vorliegen. Danach werden die anderen Informationen aktualisiert und durch die Methode „aktualisiereKategorie()“ auch in der Datenbank aktualisiert. Abschließend wird wieder das JSP „a\_kategorieAendern“ zusammen mit dem entsprechenden Feedback angezeigt.

Auch kann der Admin im JSP „a\_kategorieAendern“ das aktuelle Bild löschen. Durch Klick auf den Button wird das Servlet „Kategorieverwaltung“ mit dem Attribut „b\_loeschen“ aufgerufen. In diesem wird das Bild gelöscht und der Bildpfad der Kategorie auf „“ gesetzt. Dieses wird dann durch die Methode „aktualisiereKategorie()“ der Klasse „Kategoriemanager“ auch in die Datenbank geschrieben.

Gleiches gilt für den Button „Kategorie löschen“. Hier wird das Servlet „Kategorieverwaltung“ mit dem Attribut „loeschen“ aufgerufen. Zunächst werden alle Festivals der Kategorie mittels der Methode „selektiereAlleFestivalObjekteVonKategorie“ der Klasse „FestivalManager“ in eine Liste geschrieben und geprüft ob keine aktiven Festivals mehr verfügbar sind. Ist dies der Fall, wird die Kategorie durch die Methode „löscheKategorie()“ der Klasse „KategorieManager“ aus der Datenbank logisch gelöscht. Sollten noch aktive Festivals verfügbar sein, kann die Kategorie nicht gelöscht werden. Abschließend wird auf das JSP „a\_kategorieAendern“ mit einem entsprechenden Feedback gegangen.

Besonders muss beachtet werden, dass die ersten 4 Kategorien (ID 1-4) nicht gelöscht werden können, da diese unsere Standardkategorien sind. Gleiches gilt für die Löschung des Bildes.

#### Festivalverwaltung

JSP: a\_festivalverwaltung, a\_festivalAendern, a\_festivalAnlegen

Servlet: Festivalverwaltung

Java-Klasse: FestivalManager

In der Festivalverwaltung kann der Admin Festivals bzw. deren Artikel anlegen, ändern und löschen. Klickt der Admin auf den Button „Neues Festival anlegen“ wird das Servlet „Festivalverwaltung“ mit dem Attribut „anlegenanzeigen“ aufgerufen, das wiederum das JSP „a\_festivalAnlegen“ mit einer Liste der Kategorien (für die Kategorienzuordnung nötig) öffnet. Hier kann der Admin die Daten eingeben, ein Bild hochladen und auf „anlegen“ klicken. Dadurch wird das Servlet „Festivalverwaltung“ mit dem Attribut „anlegen“ ausgeführt. Es werden die Parameter eingelesen, zwischengespeichert und auf Vollständigkeit geprüft. Sind diese nicht vollständig wird ein entsprechendes Feedback zurückgegeben. Sind die Daten vollständig, wird zunächst ein neues Festival ohne den Bildpfad mit der Methode „erstelleFestival“ der Klasse „FestivalManager“ angelegt. Anschließend wird das Bild in den Ordner „WebContent\Bilder“ hochgeladen und das Festival mit der Methode „aktualisiereFestival“ aktualisiert. Abschließend wird das JSP „a\_festivalAnlegen“ mit einem entsprechenden Feedback aufgerufen.

Möchte der Admin ein Festival ändern, klickt er im JSP „a\_festivalverwaltung“ auf die ID des entsprechenden Festivals und führt das Servlet „Festivalverwaltung“ mit dem Attribut „aendern“ aus. Hier werden zunächst die Parameter eingelesen und geprüft, ob die notwendigen Informationen eingetragen wurden. Ist dem nicht so, wird ein entsprechendes Feedback an den Kunden gegeben und nichts geändert. Liegen alle Informationen vor, wird zunächst das Bild hochgeladen und geändert – sollte ein neues Bild vorliegen. Danach werden die anderen Informationen aktualisiert und durch die Methode „aktualisiereFestival()“ auch in der Datenbank aktualisiert. Abschließend wird wieder das JSP „a\_festivalAendern“ zusammen mit dem entsprechenden Feedback angezeigt.

Auch kann der Admin im JSP „a\_festivalAendern“ das aktuelle Bild löschen. Durch Klick auf den Button wird das Servlet „Festivalverwaltung“ mit dem Attribut „b\_loeschen“ aufgerufen. In diesem wird das Bild gelöscht und der Bildpfad des Festivals auf „“ gesetzt. Dieses wird dann durch die Methode „aktualisiereFestival()“ der Klasse „FestivalManager“ auch in die Datenbank geschrieben.

Für den Button „Festival löschen“ gilt, dass das Servlet „Festivalverwaltung“ mit dem Attribut „loeschen“ aufgerufen wird, nachdem man den Hinweis bestätigt hat, dass alle Artikel automatisch mitgelöscht werden. Zunächst wird das Festival durch die Methode „löscheFestival“ der Klasse „FestivalManager“ in der Datenbank logisch gelöscht. Anschließend werden alle Artikel des Festivals durch die Methode „löscheArtikel“ der Klasse „ArtikelManger“ logisch gelöscht. Abschließend wird das JSP „a\_festivalAendern“ mit einem entsprechenden Feedback für den Kunden aufgerufen.

Zusätzlich können noch Artikel zu den Festivals erstellt, geändert und gelöscht werden. Dieses läuft äquivalent zu dem Kapitel 4.2.2.4, nur, dass eine Festival-ID mit berücksichtigt wird und keine Bilder hinzugefügt werden können. Die Funktionsweise wird im folgenden Kapitel erklärt.

#### Artikelverwaltung (unabhängige Artikel)

JSP: a\_artikelverwaltung, a\_artikelAendern, a\_artikelAnlegen

Servlet: Artikelverwaltung

Java-Klasse: ArtikelManager

In der Artikelverwaltung kann der Admin festivalunabhängige Artikel, wie beispielsweise Regencapes und Zelte anlegen, ändern und löschen. Klickt der Admin auf den Button „Neuen Artikel anlegen“ wird das JSP „a\_artikelAnlegen“ aufgerufen. Hier kann der Admin die Daten eingeben und auf „anlegen“ klicken. Dadurch wird das Servlet „Artikelverwaltung“ mit dem Attribut „anlegen“ ausgeführt. Es werden die Parameter eingelesen, zwischengespeichert und auf Vollständigkeit geprüft. Sind diese nicht vollständig wird ein entsprechendes Feedback zurückgegeben. Sind die Daten vollständig, wird zunächst ein neuer Artikel ohne den Bildpfad mit der Methode „erstelleArtikel“ der Klasse „ArtikelManager“ angelegt. Anschließend wird das Bild in den Ordner „WebContent\Bilder“ hochgeladen und der Artikel mit der Methode „aktualisiereArtikel“ aktualisiert. Abschließend wird das JSP „a\_ArtikelAnlegen“ mit einem entsprechenden Feedback aufgerufen.

Möchte der Admin einen Artikel ändern, klickt er im JSP „a\_artikelverwaltung“ auf die ID des entsprechenden Artikels und führt das Servlet „Artikelverwaltung“ mit dem Attribut „aendern“ aus. Hier werden zunächst die Parameter eingelesen und es wird geprüft, ob die notwendigen Informationen eingetragen wurden. Ist dem nicht so, wird ein entsprechendes Feedback an den Kunden gegeben und nichts geändert. Liegen alle Informationen vor, wird zunächst das Bild hochgeladen und geändert – sollte ein neues Bild vorliegen. Danach werden die anderen Informationen aktualisiert und durch die Methode „aktualisiereArtikel()“ auch in der Datenbank aktualisiert. Abschließend wird wieder das JSP „a\_artikelAendern“ zusammen mit dem entsprechenden Feedback angezeigt.

Auch kann der Admin im JSP „a\_artikelAendern“ das aktuelle Bild löschen. Durch Klick auf den Button wird das Servlet „Artikelverwaltung“ mit dem Attribut „b\_loeschen“ aufgerufen. In diesem wird das Bild gelöscht und der Bildpfad des Artikels auf „“ gesetzt. Dieses wird dann durch die Methode „aktualisiereArtikel()“ der Klasse „ArtikelManager“ auch in die Datenbank geschrieben.

Für den Button „Artikel löschen“ gilt, dass das Servlet „Artikelverwaltung“ mit dem Attribut „loeschen“ aufgerufen wird, nachdem man den Hinweis bestätigt hat, dass der Artikel unwiderruflich gelöscht wird. Dadurch wird das Festival durch die Methode „löscheArtikel“ der Klasse „ArtikellManager“ in der Datenbank logisch gelöscht. Abschließend wird das JSP „a\_festivalAendern“ mit einem entsprechenden Feedback für den Kunden aufgerufen. Das Löschen des Artikels mit der ID 6 ist nicht möglich, da dieses unseren Postversand darstellt.

### Übergreifende Funktionen

## Technische Funktionen

Technische Klassen, Datenbankzugriffe etc.

# Testprotokoll

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Funktion** | **M/K** | **Beschreibung** | **Erwartetes Ergebnis** | **Ergebnis** |
| T01 | Erfassen von Kundendaten (Kunde selbst) | M | Der Kunde registriert sich mit der eMailadresse [max.mustermann@test.de](mailto:max.mustermann@test.de) und dem Passwort „123456“ | Der Kunde ist registriert und kann sich mit den Anmeldedaten anmelden | Ging auf Grund der Kennwortrichtlinie nicht (siehe T02) |
| T02 | Kennwortrichtlinien prüfen | K | Der Kunde benutzt zunächst das Passwort aus T01 und dann „Test123!“ | Der Kunde bekommt zunächst eine Fehlermeldung auf Grund eines unzulässigem Passwort und kann sich beim 2. Passwort registrieren |  |
| T03 | Passwort zurücksetzen | K | Der Kunde klickt auf Passwort zurücksetzen | Der Kunde gelangt in eine Maske in der er sein Passwort zurücksetzen kann |  |
| T04 | Kunde anmelden | M | Der Kunde meldet sich mit den in T01 bzw. T02 genannten Anmeldedaten an | Der Kunde gelangt auf die Startseite und ist angemeldet |  |
| T05 | Kunde abmelden | M | Der Kunde klickt auf abmelden | Der Kunde befindet sich auf der Startseite und ist abgemeldet |  |
| T06 | Admin anmelden | M | Der Admin meldet sich mit den Admindaten an | Der Admin gelangt ins Adminmenü und ist angemeldet |  |
| T07 | Admin abmelden | M | Der Admin klickt auf abmelden | Der Admin befindet sich auf der normalen Startseite und ist abgemeldet |  |
| T08 | Anzeigen und bearbeiten von Kundendaten (Kunde selbst) | M | Der Kunde klickt auf „mein Konto“ und gibt als Daten ein:  Vorname: Max  Nachname: Mustermann  IBAN: DE 1234567891011121314  BIC: TESTDJF14F  Einzugsermächtigung: check  Adresse: Musterstraße 1  PLZ: 12345  Ort: Musterstadt | Nachdem der Kunde auf „Speichern“ geklickt hat befinden sich die Daten in der Datenbank und werden unter „Mein Konto“ angezeigt. |  |
| T09 | Löschen von Kundendaten (Kunde selbst) | M | Der Kunde klickt in „Mein Konto“ auf „löschen“ und bestätigt die Löschung mit „OK“ | Der Kunde wird in der Datenbank logisch gelöscht und kann sich nicht mehr anmelden |  |
| T10 | Erfassen von Kundendaten (Admin) | M | Der Admin geht in die Kundenverwaltung, klickt auf „Kunde anlegen“ und legt einen Kunden mit den in T01 bzw. T02 genannten Daten einen Kunden an mit der eMailadresse max.mustermann1@test.de | Der Kunde kann sich mit den passenden Anmeldedaten anmelden |  |
| T11 | Anlegen eines Adminkonto | K | Der Admin legt einen Admin mit den Daten [admin@festiva.de](mailto:admin@festiva.de) und dem Passwort „123Test!“ an | Der Admin kann sich mit den Anmeldedaten in den Adminbereich anmelden |  |
| T12 | Anzeigen und bearbeiten von Kundendaten (Admin) | M | Der Admin ändert die Kundendaten von [max.mustermann1@test.de](mailto:max.mustermann1@test.de) gemäß T08 um | Die Kundendaten sind entsprechend geändert |  |
| T13 | Löschen von Kundendaten (Admin) | M | Der Admin löscht den Account des Kunden max.mustermann1@test.de | Der Kunde ist logisch gelöscht und kann sich nicht mehr anmelden |  |
| T14 | Kundenkonto Sperrung manuell durch den Admin | M | Der Admin setzt bei dem Kunden [max.mustermann@test.de](mailto:max.mustermann@test.de) das „Sperren-Häkchen“ | Der Kunde ist gesperrt und kann sich nicht mehr anmelden |  |
| T15 | Kundenkonto Sperrung durch 3-malig falschem Passwort | K | Der Kunde gibt 3-mal das falsche Passwort ein | Der Kunde wird gesperrt und kann sich nicht mehr anmelden |  |
| T16 | Kundenentsperrung durch den Admin | M | Der Admin entsperrt den gesperrten Kunden in dem er das „Sperren-Häkchen“ entfernt | Der Kunde ist wieder entsperrt und kann sich wieder anmelden |  |
| T17 | Anlegen einer Kategorie (Admin) | M | Der Admin klickt in der Kategorieverwaltung auf Kategorie anlegen und gibt folgende Daten ein:  Name: Jazz  Beschreibung: Jazz Sessions  Bild: Jazz.jpg | Die Kategorie ist angelegt und wird bei der Kategorieverwaltung angezeigt |  |
| T18 | Anzeigen und bearbeiten einer Kategorie (Admin) | M | Die Kategorie „Jazz“ wird ausgewählt und bearbeitet. Die Beschreibung wird in „Jazz Sessions und Festivals“ geändert und gespeichert | Die Beschreibung ist geändert und in der Kategorieverwaltung zu finden |  |
| T19 | Anlegen eines Festivals (Admin) | M | Der Admin geht in die Festivalverwaltung, klickt auf „Festival anlegen“, gibt folgende Daten ein und speichert diese:  Name: Jazzed Off Session  Bild: Jazz\_off.jpg  Startdatum: 27.01.2017  Enddatum: 27.01.2017  Ort: Gütersloh  Kategorie: Jazz  Kurzbeschreibung: Jazz Off Session in der Weberei  Langbeschreibung: Jazz Off Session in der Weberei. Einlass: 17:00 Uhr Beginn 17:30 Uhr  Bogenstraße 1-8  33330 Gütersoh | Das Festival ist angelegt und in der Festivalverwaltung, sowie im Shop zu sehen |  |
| T20 | Anzeigen und bearbeiten eines Festivals (Admin) | M | Der Admin wählt das Festival „Jazz off Session“ aus und bearbeitet diese indem sie in der Langbeschreibung um die „Endzeit 22:00 Uhr“ ergänzt und speichert | Die Änderung ist auf der Eventseite sichtbar |  |
| T21 | Anlegen eines Artikels (Admin) | M | Der Admin wählt das Festival „Jazz off Session“ aus und geht in den Bearbeitenmodus. Hier klickt der Admin auf Artikel hinzufügen und fügt folgende Daten ein und speichert diese:  Name: Standardticket  Beschreibung: freie Platzwahl  Preis: 17,95 | Der Artikel ist angelegt und sowohl in der Festivalsicht des Admins als auch in der Eventsicht des Kunden zu sehen. | Fehlermeldung „Seite nicht verfügbar“ |
| T22 | Anlegen eines Merchandise Artikels (Admin) | K | Der Admin geht in die Artikelverwaltung, klickt auf „neuen Artikel anlegen“ und fügt folgende Daten ein und speichert diese:  Name: Regencape  Bild: Regencape.jpg  Preis: 25,99 | Der Artikel ist angelegt und sowohl in der Festivalsicht des Admins als auch in der Eventsicht des Kunden zu sehen. |  |
| T23 | Anzeigen und bearbeiten eines Artikels(Admin) | M | Der Admin geht zunächst auf das Festival „Jazz off Session“ und dann auf den Artikel Standardticket um ihn zu bearbeiten. Er ändert den Preis auf 15,95 und speichert diese | Die Änderung ist gespeichert und sowohl in der Festivalsicht des Admins als auch in der Eventsicht des Kunden zu sehen |  |
| T24 | Löschen eines Artikels (Admin) | M | Der Admin geht zunächst auf das Festival „Jazz off Session“ und dann auf den Artikel Standardticket um ihn zu bearbeiten. Er klickt auf „löschen“ und bestätigt dieses durch „OK“ um den Artikel zu löschen | Der Artikel ist logisch gelöscht und sowohl in der Festivalsicht des Admins als auch in der Eventsicht des Kunden nicht mehr zu sehen |  |
| T25 | Löschen eines Festivals (Admin) | M | Der Admin wählt das Festival „Jazz off Session“ aus um es zu bearbeiten. Er klickt anschließend auf „löschen“ und bestätigt dieses anschließend durch „OK“ um das Festival zu löschen | Das Festival ist logisch gelöscht und sowohl in der Festivalsicht des Admins als auch in der Eventsicht des Kunden nicht mehr zu sehen |  |
| T26 | Löschen einer Kategorie (Admin) | M | Die Kategorie „Jazz“ wird zum Bearbeiten geöffnet und durch einen Klick auf „löschen“ sowie die Bestätigung durch „OK“ gelöscht | Die Kategorie „Jazz“ wird logisch gelöscht und ist nicht mehr in der Kategorieverwaltung zu finden |  |
| T27 | Festivalsuche nach Name | M | Es wird im Shop nach dem Begriff „Session“ gesucht | Mindestens das Festival „Jazz off Session“ ist gefunden worden und Festivals auf dem das Suchkriterium passt |  |
| T28 | Festivalsuche nach Ort | M | Es wird im Shop nach dem Ort „Gütersloh“ gesucht | Mindestens das Festival „Jazz off Session“ ist gefunden worden und Festivals auf dem das Suchkriterium passt |  |
| T29 | Festivalsuche nach Kategorie | M | Es wird im Shop nach der Kategorie „Jazz“ gesucht | Mindestens das Festival „Jazz off Session“ ist gefunden worden und Festivals auf dem das Suchkriterium passt |  |
| T30 | Festivalsuche nach Datum | M | Es wird nach dem Zeitraum 22.01.2016 – 31.01.2016 gesucht | Mindestens das Festival „Jazz off Session“ ist gefunden worden und Festivals auf dem das Suchkriterium passt |  |
| T31 | Direktnavigation von der Startseite nach Kategorie | M | Es wird auf der Startseite auf die Kategorie „Jazz“ geklickt | Es wird in den Shop gewechselt und die Kategoriesuche ist auf „Jazz“ voreingestellt. Mindestens das Festival „Jazz off Session“ ist gefunden worden und Festivals auf dem das Suchkriterium passt |  |
| T32 | Artikel zum Warenkorb hinzufügen | M | Es wird das Festival „Jazz off Session“ ausgewählt und der Artikel „Standartticket“ mit 2 Tickets in den Warenkorb gelegt | 2 Standarttickets für das Festival „Jazz off Session“ liegen in dem Warenkorb |  |
| T33 | Artikel im Warenkorb anzeigen und bearbeiten | M | Es wird in den Warenkorb gegangen und die Anzahl der Standarttickets für das Festival „Jazz of Session“ auf 4 erhöht | Die Ticketanzahl liegt jetzt bei 4 |  |
| T34 | Artikel aus dem Warenkorb löschen | M | Der Artikel „Standartticket“ wird aus dem Warenkorb durch einen Klick auf „löschen“ gelöscht | Der Artikel befindet sich nicht mehr im Warenkorb |  |
| T35 | Bestellung ausführen | M | Nachdem der Artikel „Standartticket“ wieder hinzugefügt wurde, auf „zur Kasse“ geklickt, Post als Lieferoption ausgewählt und dann auf „bezahlen geklickt. | Alle Informationen werden richtig angezeigt und berechnet. Die Bestellung ist ausgeführt, der Warenkorb gelehrt und die Bestellung in der Datenbank abgelegt. |  |
| T36 | Bestellungen einsehen (Kunde) | M | Der Kunde klickt auf „Meine Bestellungen“ | Die eben abgegebene Bestellung wird angezeigt |  |
| T37 | Bestellungen einsehen (Admin) | K | Der Admin klickt auf „Bestellungen“. | Die eben abgegebene Bestellung wird angezeigt | Wurde nicht implementiert, da die Anwendungsfälle unrealistisch sind |
| T38 | Seite auf Mobile Device starten | M | Die Seite wird über einen Emulator gestartet und angezeigt | Die Seite entspricht den Anforderungen eines Responsiv Designs |  |

Zusätzlich zu dem oben zusehenden Testplan wurden während der Realisierungsphase White-Boxtests der Komponenten von den Entwicklern durchgeführt

# Projektmanagement

Im Rahmen der Projektplanung wird das Carl-Steinweg-Phasenmodell eingesetzt:

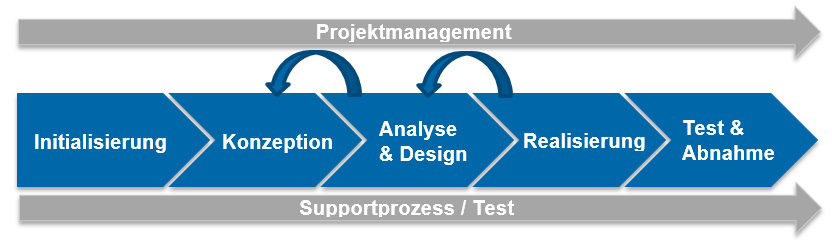


Abbildung 9 – Carl-Steinweg Phasenmodell

Der Projektmanagementbereich dient während des gesamten Projektzeitraumes der Leitung und Steuerung des Projektes. Der blau dargestellte Kernbereich ist in mehrere Phasen, in denen es gegebenenfalls Rücksprünge geben kann, gegliedert. Parallel dazu verläuft der Support- bzw. Testprozess, der insbesondere eine frühe Fehlererkennung ermöglicht.

## Soll-Ist-Vergleich

### Projektplan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Vorgangsname | Dauer | Anfang | Ende | Ressourcennamen | Geplante Arbeit | Benötigte Arbeit | Abweichung Arbeit (Std.) |
| **Projektinitialisierung** | **1 Tag** | **Mit 05.10.16** | **Mit 05.10.16** |  | **3 Std.** | **3 Std.** | **0** |
| Kickoff-Meeting | 1 Tag | Mit 05.10.16 | Mit 05.10.16 | Alina; Nicola; Timo | 3 Std. | 3 Std. | 0 |
| **Konzeption** | **2 Tage** | **Don 06.10.16** | **Fre 07.10.16** |  | **22 Std.** | **22 Std.** | **0** |
| Pflichtenheft | 1 Tag | Don 06.10.16 | Don 06.10.16 | Alina; Nicola; Timo | 6 Std. | 7 Std. | 1 |
| Projektplan | 1 Tag | Fre 07.10.16 | Fre 07.10.16 | Alina; Nicola; Timo | 16 Std. | 15 Std. | -1 |
| **Design** | **6 Tage** | **Fre 07.10.16** | **Fre 14.10.16** |  | **53 Std.** | **54 Std.** | **1** |
| Festlegen der Namenskonventionen | 1 Tag | Fre 07.10.16 | Fre 07.10.16 | Alina; Nicola; Timo | 3 Std. | 3 Std. | 0 |
| Erstellung ER-Diagramm | 4 Tage | Fre 07.10.16 | Mit 12.10.16 | Alina | 7 Std. | 6 Std. | -1 |
| Erstellung Klassendiagramm | 4 Tage | Fre 07.10.16 | Mit 12.10.16 | Alina | 13 Std. | 13 Std. | 0 |
| Erstellung Use-Case-Diagramme | 4 Tage | Fre 07.10.16 | Mit 12.10.16 | Nicola | 8 Std. | 9 Std. | 1 |
| Erstellung von Mock-Ups für die Oberfläche | 5 Tage | Fre 07.10.16 | Don 13.10.16 | Timo | 10 Std. | 11 Std. | 1 |
| Entwurf Testplan | 5 Tage | Fre 07.10.16 | Don 13.10.16 | Alina; Nicola; Timo | 6 Std. | 5 Std. | -1 |
| Zusammenfassung Diagramme / Anforderungen / Testplan | 1 Tag | Fre 14.10.16 | Fre 14.10.16 | Alina; Nicola; Timo | 6 Std. | 7 Std. | 1 |
| **Realisierung** | **20 Tage** | **Fre 14.10.16** | **Don 10.11.16** |  | **176 Std.** |  |  |
| Installation und Konfiguration von Eclipse, Tomcat, MySQL… | 2 Tage | Fre 14.10.16 | Mon 17.10.16 | Alina; Nicola; Timo | 15 Std. | 12 Std. | -3 |
| Erstellung der Tabellendefinitionen in der Datenbank | 1 Tag | Die 18.10.16 | Die 18.10.16 | Alina | 4 Std. |  |  |
| Datenbankanbindung | 1 Tag | Die 18.10.16 | Die 18.10.16 | Alina; Nicola; Timo | 6 Std. |  |  |
| Umsetzung Oberfläche | 8 Tage | Die 18.10.16 | Don 27.10.16 | Nicola | 35 Std. |  |  |
| Oberflächen-Logik | 10 Tage | Die 18.10.16 | Mon 31.10.16 | Alina; Nicola; Timo | 21 Std. |  |  |
| Backend-Entwicklung | 13 Tage | Mit 19.10.16 | Fre 04.11.16 | Alina; Timo | 50 Std. |  |  |
| Verbindung Frontend & Backend | 3 Tage | Mon 07.11.16 | Mit 09.11.16 | Alina; Nicola | 10 Std. |  |  |
| Responsive Design | 6 Tage | Fre 28.10.16 | Fre 04.11.16 | Timo | 29 Std. |  |  |
| Dokumentation / Refactoring | 1 Tag | Don 10.11.16 | Don 10.11.16 | Alina; Nicola; Timo | 6 Std. |  |  |
| **Test** | **5 Tage** | **Fre 11.11.16** | **Don 17.11.16** |  | **33 Std.** |  |  |
| Testkonzeption | 1 Tag | Fre 11.11.16 | Fre 11.11.16 | Alina; Nicola; Timo | 15 Std. |  |  |
| Testdaten pflegen | 2 Tage | Mon 14.11.16 | Die 15.11.16 | Nicola | 8 Std. |  |  |
| Integrationstest | 2 Tage | Mit 16.11.16 | Don 17.11.16 | Timo | 10 Std. |  |  |
| **Dokumentation** | **21 Tage** | **Fre 18.11.16** | **Fre 16.12.16** |  | **45 Std.** |  |  |
| Soll- / Ist-Vergleich | 1 Tag | Fre 18.11.16 | Fre 18.11.16 | Alina; Nicola; Timo | 5 Std. |  |  |
| Präsentation | 2 Tage | Fre 18.11.16 | Mon 21.11.16 | Timo | 15 Std. |  |  |
| Dokumentation | 19 Tage | Die 22.11.16 | Fre 16.12.16 | Alina; Nicola; Timo | 25 Std. |  |  |
| **dauernde Projektleitung (Abstimmungen, Entscheidungen,…)** | **52 Tage** | **Don 06.10.16** | **Fre 16.12.16** | **Alina; Nicola; Timo** | **28 Std.** | **2 Std.** | **-26** |

### Muss/Kann-Kriterien

**Muss-Kriterien:**

* Erfassen, anzeigen, bearbeiten, löschen der Kundendaten (Registrierung/Anmeldung/Verwaltung durch den Kunden selbst und durch den Admin) 
* Der Admin kann den Kunden manuell bei der Bearbeitung der Kundendaten sperren 
* Die Passwörter werden als Hash-Wert in der Datenbanktabelle abgelegt. 
* Erfassen, anzeigen, bearbeiten, löschen der Kategoriendaten (durch den Admin)
* Erfassen, anzeigen, bearbeiten, löschen der Festivaldaten (durch den Admin)
* Erfassen, anzeigen, bearbeiten, löschen der Artikeldaten (durch den Admin)
* Die Suche nach Festivals kann über Name, Ort, Kategorie und Datum eingegrenzt werden.
* Es gibt eine direkte Navigation zur Festivalsuche über die Startseite. Dort kann eine Kategorie angeklickt werden. Anschließend sind weitere Eingrenzungen über die Kriterien (s.o.) möglich.
* Erfassen von Bestellungen und anzeigen von vergangenen Bestelldaten
* Artikel in den Warenkorb legen, Warenkorb anzeigen und bearbeiten 
* Benutzerauthentifizierung, Benutzer mit unterschiedlichen Rechten (Kunden und Admin inkl. Log-In und Log-Out)
* Responsive Design auf Desktop und Android-Smartphone

**Kann-Kriterien:**

* Kunden werden automatisch gesperrt (z.B. nach 3-mal falsch eingegebenem Kennwort)
* Bei der Festivalsuche kann in einem bestimmten Umkreis gesucht werden
* Prüfung, ob eine zulässige IBAN eingegeben wurde
* Kennwortrichtlinien festlegen und prüfen
* Automatisierte Möglichkeit um das Passwort zurückzusetzen
* Bewertungen für Artikel abgeben und einsehen können
* Bestandsführung der Artikel
* Verwaltung der Kundenbestellungen aus Administrator-Sicht
* Anlage von Admin-Konten über eine Oberfläche 
* Einbinden von zusätzlichen Festival-Artikeln (Regencapes etc.)

## Ressourcenzuordnung

Wer hat wie viele Stunden gemacht

|  |  |  |
| --- | --- | --- |
| **Name** | **Soll** | **Ist** |
| Alina Fankhänel | 120 |  |
| Nicola Kloke | 120 |  |
| Timo Schlüter | 120 |  |

# Fazit/Bewertung

Zusammenfassend lässt sich sagen, dass das Projekt fristgerecht, mit einer leichten Überschreitung der zuvor kalkulierten 360 Stunden, abgeschlossen wurde. Des Weiteren konnten neben den Muss-Kriterien, einige Kann-Kriterien erfolgreich umgesetzt werden.

Die gestalteten Diagramme in der Designphase bildeten eine gute Orientierung und entsprachen am Ende des Projekts mit wenigen Anpassungen dem Endresultat.

Zeitverluste mussten unter anderem durch Einarbeitung in Tools und fehlenden Programmierkenntnissten, sowie Differenzen innerhalb des Teams in Kauf genommen werden. Insgesamt entstand jedoch ein großer Lerneffekt, dadurch dass vieles von Grund auf erlernt werden musste. Ebenfalls wurde uns bewusst, wie wichtig eine gute Planung innerhalb des Teams ist.

*Alles in allem sind wir trotz leichter Abweichungen im Projektplan mit dem Endprojekt zufrieden. Wir haben alle Muss-Kriterien und auch einige Kann-Kriterien erfüllen können. Auch wenn zum Beginn der Realisierungsphase wir hinter dem Plan lagen, konnten wir durch einen stetigen Lernprozess den Rückstand zum Ende hin gut aufholen. Zu Beginn hat das fehlende Wissen den Projektplan ins Schwanken gebracht. Ein Problem während des Projekts war die Kommunikation innerhalb des Teams. Hier wurden Absprachen falsch verstanden, was zu Missverständnissen und Unmut geführt hat. Das hatte zur Folge, dass Termine nicht eingehalten worden sind und Aufgaben nicht richtig erfüllt wurden. Dieses hätte eventuell durch eine bessere und regelmäßigere Kommunikation zwischen den Mitgliedern des Teams verhindert werden können. Letztendlich haben wir den Unmut und die Probleme durch ein Gespräch zusammen mit Marcus Becker lösen können. In Zukunft müssen wir darauf achten, dass ähnliches nicht nochmal vorkommt. Zusammenfassend kann man sagen, dass das Projekt das gemacht hat was es machen sollte. Unsere Fähigkeiten haben sich in verschiedenen Disziplinen wie Projektplanung oder auch Programmierung über das Model-View-Control Konzept verbessert. Auch wurde unsere Fähigkeit Konflikte lösen zu können durch unser zwischenzeitiges Tief gestärkt.*

# Sonstiges

Sollten sich Screenshots oder Funktionalitäten ändern, sowie anderer Fehler in dem Dokument auftauchen, informieren Sie bitte den zuletzt genannten Autor.

Bei Verbesserungsvorschlägen wenden Sie sich bitte, an dem zuletzt genannten Autor.

# Eigenständigkeitserklärung

|  |
| --- |
| **Persönliche Erklärung der Studenten:**  Wir versichern durch unsere Unterschriften, dass wir das Projekt und die dazugehörige Dokumentation selbstständig und ohne fremde Hilfe angefertigt haben. Alle Stellen, die wir wörtlich aus Veröffentlichungen entnommen haben, sind als solche kenntlich gemacht worden. Die Arbeit hat in dieser Form keiner anderen Prüfungsinstitution vorgelegen.  .......................................................... ......................................................................................................  Ort, Datum Unterschrift der Studenten |

# Quellenverzeichnis

* Brandt-Pook | Kollmeier - Softwareentwicklung kompakt und verständlich (1. Auflage 2008) für die Projektor-ganisation
* http://www.oracle.com/technetwork/articles/javase/servlets-jsp- 140445.html

# Glossar

|  |  |
| --- | --- |
| **Begriff** | **Beschreibung** |
|  |  |