Hochschule Ostwestfalen-Lippe
Fachbereich 8 - Umweltingeneurwesen und Angewandte Informatik
Fachgebiet Angewandte Informatik
Programmieren 3
5. Semester WS 2015/16

Dokumentation

Implementierung eines Prototypen für eine Verwaltungssoftware für ein Call-Center

Von

Robin Hake
15306070

und

Benedikt Brüntrup
15306067

Erstprüfer: Prof. Dr. Ralf Hesse
Eingereicht am: 1. Januar 2016

Inhaltsverzeichnis

1 Aufgabenstellung

Als Abschlussarbeit des Moduls „Programmiersprachen 3“ sieht die Prüfungsordnung die Abgabe eines in JavaEE implementierten Software-Projekts vor. Von einer Auswahlliste sollte sich ein Thema ausgesucht werden und zwei Anwendungsszenarien des Themas in einen Software-Prototyp umgesetzt werden. Der Software-Prototyp war dabei auf Basis eines dreischichtigen Ansatz umzusetzen. Die Gruppe Robin Hake und Benedikt Brüntrup entschloss sich das Thema „Verwaltungssoftware für ein Call-Center“ umzusetzen.

2 Projektplanung

Dieser Abschnitt beschreibt, wie bei der Projektplanung vorgegangen wurde. Es wird auf die Anforderungsanalyse, den ersten Designentwurf, den Datenbankentwurf und das gewählte Entwurfsmuster eingegangen.

2.1 Anforderungsanalyse

Zu Beginn der Projektplanung wurde mit Hilfe eines Use-Case-Diagramms ermittelt was das Projekt alles können soll. Das Ergebnis der Anforderungsanalyse kann der *Abbildung 1* entnommen werden.

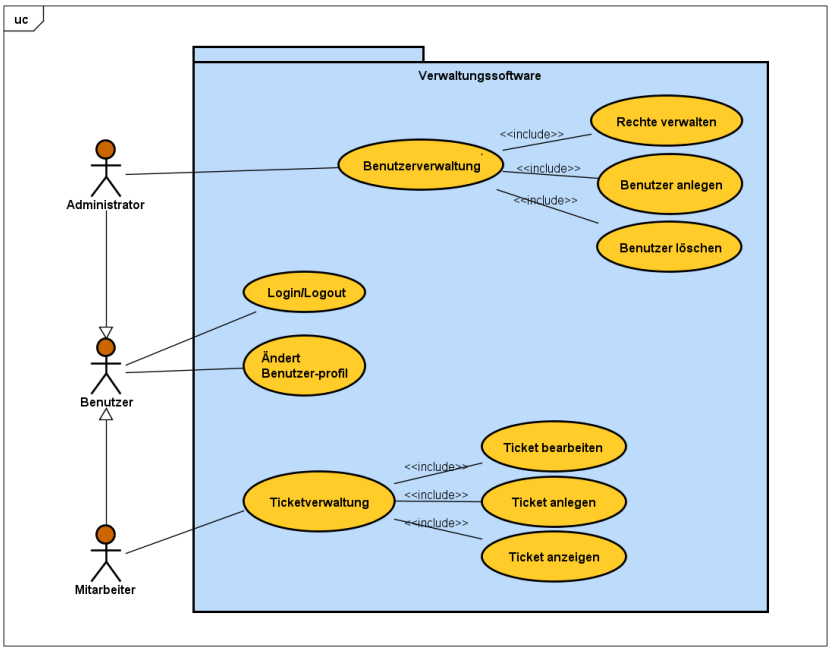

Abbildung 1: Anforderungsanalyse als Use-Case-Diagramm

2.2 Erster Designentwurf

Als nächstes wurde mit dem Programm PowerPoint ein erster Designentwurf für die Weboberfläche des Projektes entworfen. Hierbei wurde sich für ein fensterbasiertes Design ent-

schlossen. Mehr zum Desktop-Ansatz ist im *Abschnitt 2.6* zu lesen. Die folgende Abbildung zeigt den ersten Designentwurf in PowerPoint.

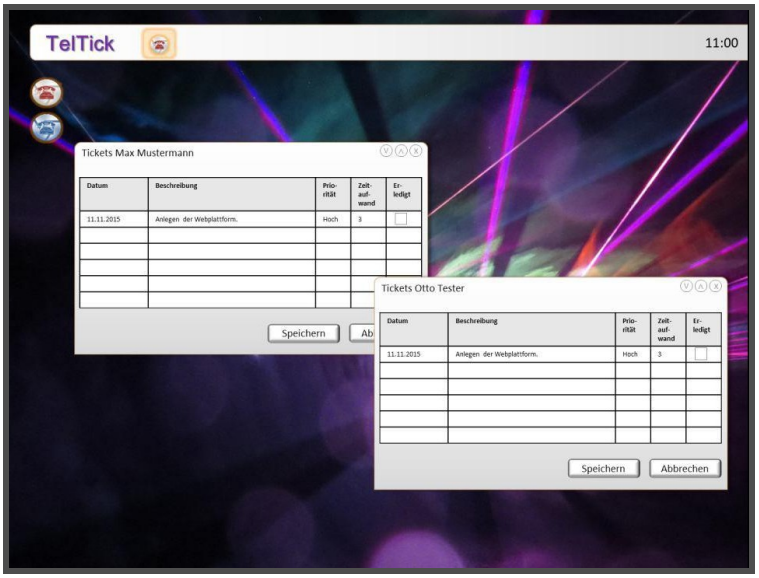

Abbildung 2: Erster Designentwurf in PowerPoint

2.3 Datenbankentwurf

Nachdem das grundlegende Design der Webseite festgelegt worden war, wurde sich über die Datenhaltung Gedanken gemacht. Es wurde ermittelt welche Daten das Software-Projekt persistent speichern soll und wie diese Daten im Zusammenhang stehen. Aus dieser Erkenntnis wurde die in der *Abbildung 3* gezeigte Datenbankstruktur festgelegt.
Die *Tabelle 1* beschreibt wofür welche Relation in der Datenbank benötigt wird.

Tabellenname	Beschreibung
Mitarbeiter	In dieser Relation werden die Mitarbeiter hinterlegt, die sich an der Webseite anmelden können sollen.
Rechte	In dieser Relation werden die Fenster eingetragen, auf die ein bestimmter Mitarbeiter Zugriffsrechte haben soll.
Ticket	Enthält die erstellten Tickets.
Fenster	Enthält die Abmaße, die Titel und die Pfade zun den JSP-Dateien der Fenster, die auf der Webseite angezeigt werden sollen.
Ticketzuweisung	Weißt ein Ticket einen bestimmten Benutzer zu.

Tabelle 1: Beschreibung der einzelnen Relationen

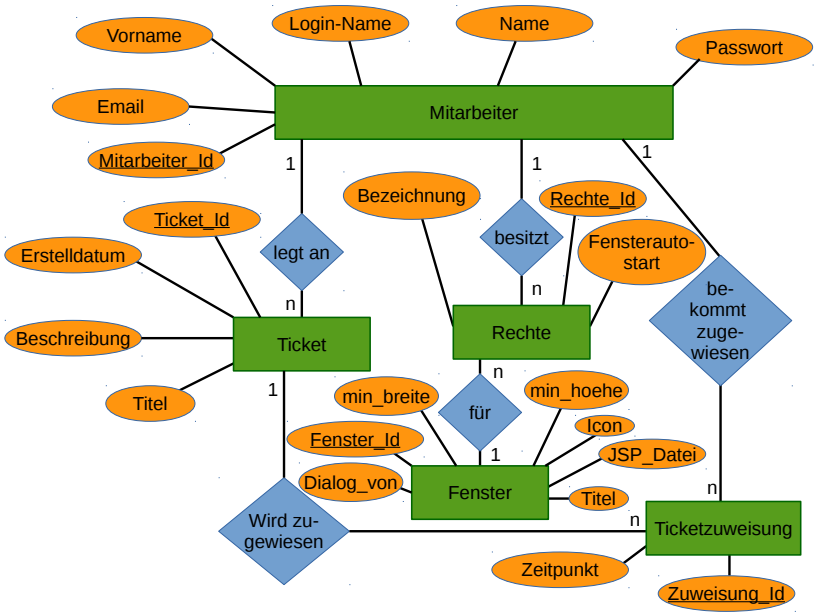

Abbildung 3: Datenbankstruktur

2.4 Architektur

Als Architektur-Modell wurde das MVC-Modell (**M**odel, **V**iew, **C**ontrol) gewählt. Die *Abbildung 4* stellt da, wie das MVC-Modell im Projekt umgesetzt wurde. Die „Views“ wurden im Projekt durch JSP-Seiten umgesetzt, die „Controller“ durch Servlets und das „Model“ durch DAO-Objekte. Diese DAO-Objekte mappen die Datenbank in normale Java-Objekte, sodass der Anwendungsentwickler wie gewohnt mit Java-Klassen arbeiten kann und nicht im Quellcode direkt Datenbankabfragen senden muss. Dafür bieten die DAO-Objekte einfache „Store“- und „Load“-Methoden. Diese Methoden mappen dann ein Java-Objekt in der Datenbank.

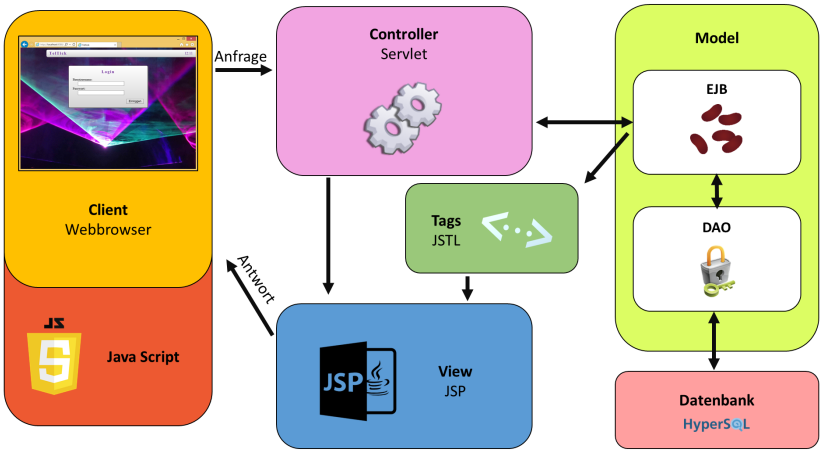

Abbildung 4: Architektur des Projektes

2.5 Factory Pattern

Um Klassen möglichst austauschbar in das Projekt einzubinden, wurde auf die Verwendung des „new-Operators“ verzichtet. Wie in der *Abbildung 5* zu sehen ist werden beim Projekt „Factory-Klassen“ verwendet um Klassen zu instanziieren. Soll eine Klasse ausgetauscht werden muss nur die Factory-Klasse angepasst werden und in der neuen Klasse das Interface implementiert sein.

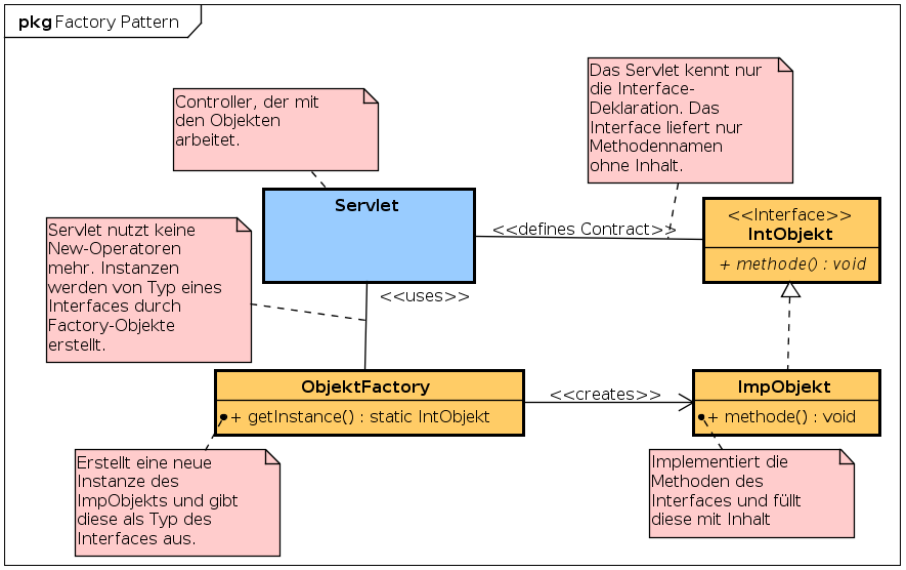

Abbildung 5: FactoryPattern

2.6 Der Desktopansatz

Beim Projekt wurde sich dazu entschieden einen Desktop-Fenster-Ansatz zu benutzen. Dies bietet den Vorteil, dass das Projekt modular erweiterbar ist. Soll ein neues Modul zum Projekt hinzugefügt werden, muss nur auf den Desktop ein neues Icon hinzugefügt werden, welches ein Fenster mit dem neuen Modul öffnet. Somit können Module auch problemlos ausgetauscht werden.

3 Implementierung

In diesem Abschnitt wird beschrieben, wie Einzelheiten des Projekts in JavaEE umgesetzt wurden. Es wird auf den Desktop-Aufbau, Use Cases und auf das Objektmodell eingegangen.

3.1 Umsetzung des Desktops

Der Desktop besteht, wie in der *Abbildung 6* gezeigt, aus mehreren verschachtelten JSP-Dateien. Nach dem Anmelden wird standardmäßig nur die „desktop.jsp“ geladen. Die Verschaltung wird erstellt wenn der Benutzer, wie bei *Abbildung 7*, ein neues Fenster öffnet.

Abbildung 6: Aufbau des Desktops

Die in dem Fenster integrierten JSP-Dateien enthalten Formulare. Damit bei einen Submit des Formulars nicht die ganze Seite neu geladen wird, sondern nur das Fenster direkt, wird beim Submit die Funktion „submitUmleiten“ aufgerufen. Diese leitet den Submit via Ajax um, sodass nur das Formular neugeladen wird und das Ergebnis im Fenster zu sehen ist.

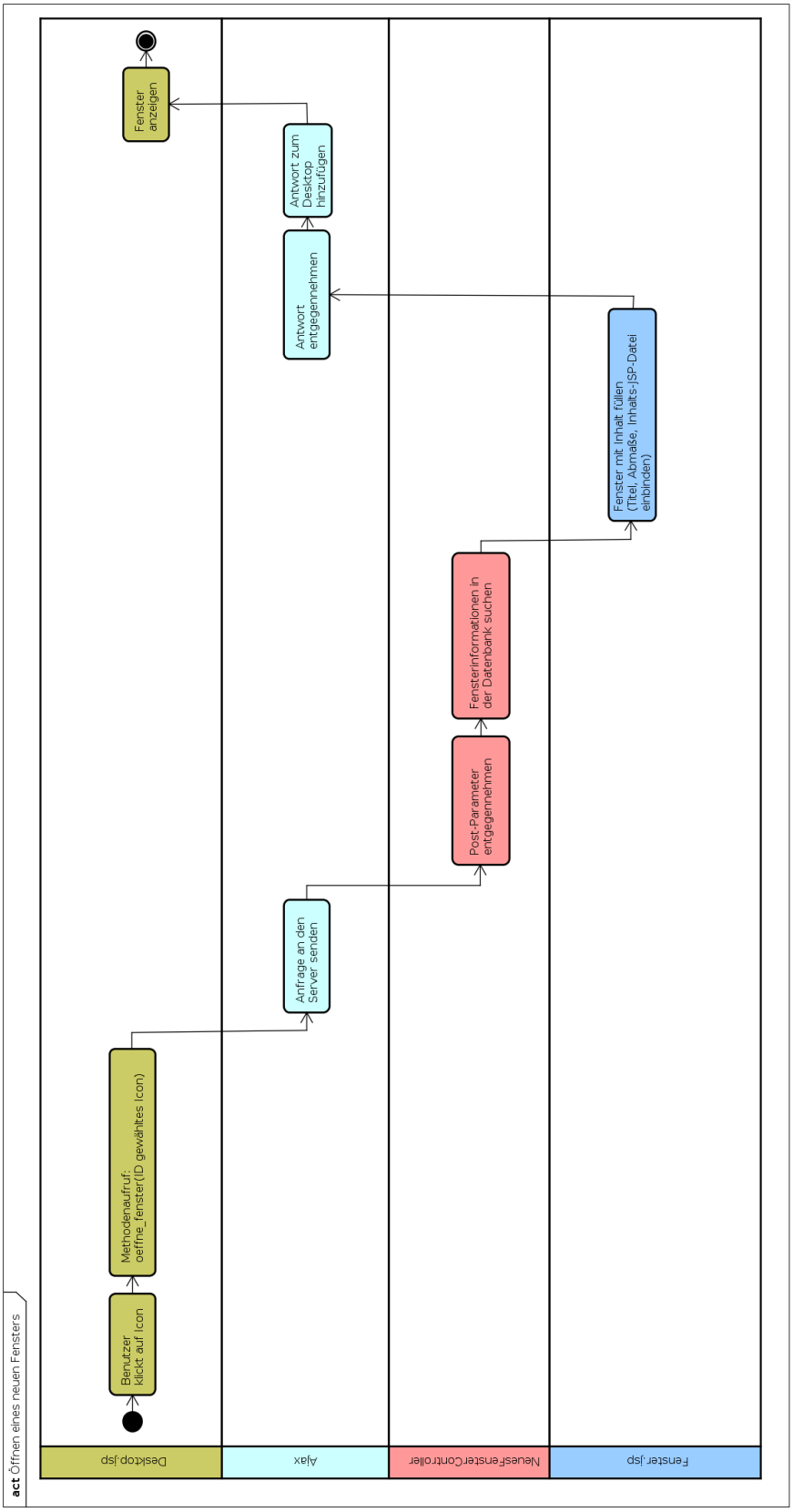

Abbildung 7: Öffnen eines neuen Fensters

3.2 Use Cases (Umsetzung)

In der Anforderungsanalyse wurden mehrere Use Cases festgelegt. Es wurde sich dazu entschieden folgende Use Cases umzusetzten:

- Login\Logout (Grundlegende Anforderung)
- Benutzerverwaltung
  - Benutzer anlegen
  - Benutzer ändern
  - Benutzer löschen
- Ticketverwaltung
  - Ticket anlegen
  - Ticket anzeigen

Im folgendem werden Teile der Use Cases anhand von Tabellen näher erläutert.

3.2.1 Benutzerverwaltung

Use Case:	Benutzer anlegen
Ziel:	Anlegen eines neuen Benutzers
Haupt-Anwender:	Admin, Mitarbeiter
Auslöser:	Button: „Benutzerverwaltung“, „Neuer Benutzer“
Vorbedingungen:	Mitarbeiter ist eingeloggt, hat Rechte zum Benutzer anlegen
Nachbedingungen:	Benutzer erfolgreich angelegt
Normalablauf:	1. [AKTEUR]: Klick auf Button „Neuer Benutzer“ 2. [SYSTEM]: Öffnen des Formulars für einen neuen Benutzer 3. [AKTEUR]: Ausfüllen des Formulars und Bestätigung 4. [SYSTEM]: Überprüfung der Eingaben 5. [SYSTEM]: Weiterleitung zur Benutzerübersicht 6. [AKTEUR]: Ansicht des Benutzers, Fenster schließen
Fehlerablauf:	Fehlerbehandlung „Formulareingaben unvollständig“ Einsprung bei (4) 1. [SYSTEM]: Öffnen des Formulars für einen neuen Benutzer 2. [SYSTEM]: Fehlermeldung welche Eingaben unvollständig waren 3. [AKTEUR]: Vervollständigt seine Eingaben 4. [SYSTEM]: Überprüfung der Eingaben 5. [SYSTEM]: Weiterleitung zur Benutzerübersicht 6. [AKTEUR]: Ansicht des Benutzers, Fenster schließen
Zusätzliche Abläufe:	Abbruch der Bearbeitung 1. [AKTEUR]: Wählt Button zum schließen des Fensters 2. [SYSTEM]: Schließt Fenster

3.2.2 Ticketverwaltung

Use Case:	Ticket anlegen
Ziel:	Anlegen eines neuen Tickets
Haupt-Anwender:	Mitarbeiter
Auslöser:	Button: „Neues Ticket“
Vorbedingungen:	Mitarbeiter ist eingeloggt, hat Rechte zum Ticket anlegen
Nachbedingungen:	Ticket erfolgreich angelegt
Normalablauf:	1. [AKTEUR]: Klick auf Button „Neues Ticket“ 2. [SYSTEM]: Öffnen des Formulars für ein neues Ticket 3. [AKTEUR]: Ausfüllen des Formulars und Bestätigung 4. [SYSTEM]: Überprüfung der Eingaben 5. [SYSTEM]: Weiterleitung zum Anzeigen des neuen Tickets 6. [AKTEUR]: Ansicht des Tickets, Fenster schließen
Fehlerablauf:	Fehlerbehandlung „Formulareingaben unvollständig“ Einsprung bei (4) 1. [SYSTEM]: Öffnen des Formulars für ein neues Ticket 2. [SYSTEM]: Fehlermeldung welche Eingaben unvollständig waren 3. [AKTEUR]: Vervollständigt seine Eingaben 4. [SYSTEM]: Überprüfung der Eingaben 5. [SYSTEM]: Weiterleitung zum Anzeigen des neuen Tickets 6. [AKTEUR]: Ansicht des Tickets, Fenster schließen
Zusätzliche Abläufe:	Abbruch der Bearbeitung 1. [AKTEUR]: Wählt Button zum schließen des Fensters 2. [SYSTEM]: Schließt Fenster

Use Case:	Ticket anzeigen
Ziel:	Anzeigen eines ausgewählten Tickets
Haupt-Anwender:	Mitarbeiter
Auslöser:	Button: „Ticket lesen“
Vorbedingungen:	Mitarbeiter ist eingeloggt, hat Rechte zum Ticket lesen
Nachbedingungen:	Ticket wurde erfolgreich angezeigt
Normalablauf:	1. [AKTEUR]: Klick auf Button „Ticket lesen“ 2. [SYSTEM]: Öffnen der Ticketverwaltung 3. [AKTEUR]: Auswählen des gewünschten Tickets (Filterauswahl) 4. [AKTEUR]: „Anzeigen“Button klicken 5. [SYSTEM]: Laden des gewünschten Tickets 6. [AKTEUR]: Ansicht des Tickets, Fenster schließen
Zusätzliche Abläufe:	Abbruch der Bearbeitung 1. [AKTEUR]: Wählt Button zum schließen des Fensters 2. [SYSTEM]: Schließt Fenster

3.2.3 Objektdiagramm

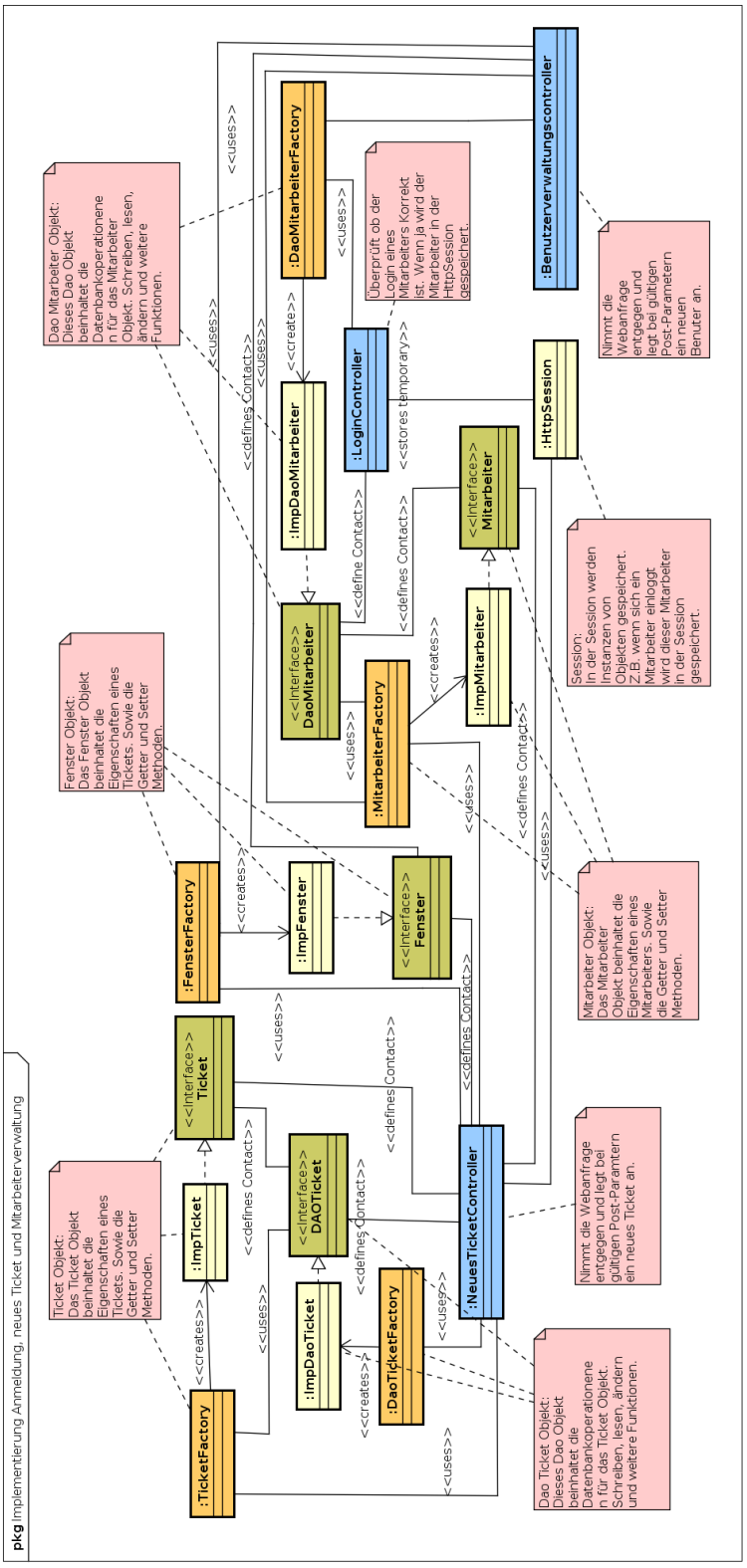

Abbildung 8: Objektdiagramm

Das Objektdiagramm (siehe *Abbildung 8*) stellt dar, wie die erläuterten Use-Cases Softwaretechnisch umgesetzt wurden. Es definiert die einzelnen Objekte der Umsetzung und beschreibt, wie diese miteinander in Beziehung stehen.

4 Projekt auf dem Server einrichten

- Systemvorraussetzung:
  - Tomcat 8 und Java 8 installiert
- Das gegebene war-File muss beim „DocumentRoot-Verzeichnis“ des Tomcats ins Unterverzeichnis „webapps“ eingefügt werden.
  - z.B.: „C:\apache-tomcat\webapps\teltick.war“
- Anschließend kann der Tomcat gestart werden.
- Admin-Zugangsdaten:
  - Benutzername: Admin
  - Passwort: P@ssw0rd

5 Anhang

Im Anhang sind folgende Dokumente zu finden:

- Benutzerverwaltung Designentwurf (benutzerverwaltung.png)