Software Engineering & Projektmanagement

Einzelbeispiel, Anwendungsfall 5: Versicherung

07.04.2013

Technische Universität Wien

Englisch Jakob, 1125164

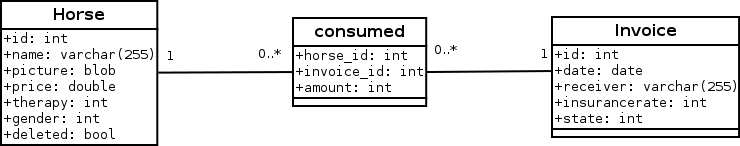
e1125164@student.tuwien.ac.at



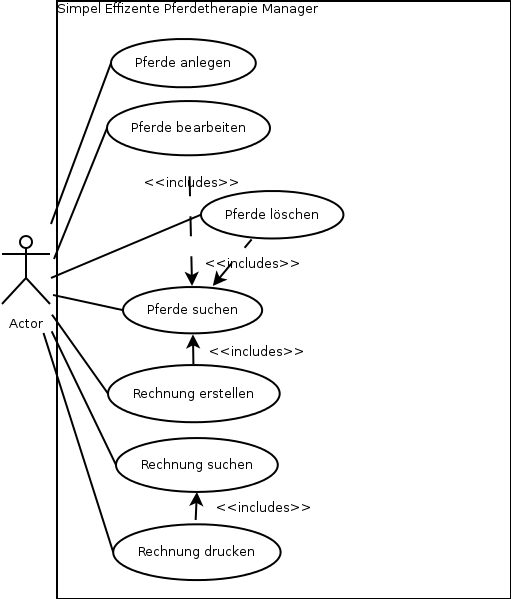
# Stundenliste

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Anfang | Dauer | Beschreibung |
| 12.3.2013 | 15:00 | 0:30 | Buildscripts, UI Test |
| 16.3.2013 | 16:00 | 0:45 | Swing Tests |
| 16.3.2013 | 21:00 | 4:00 | Domain Model – Java Klassen und SQL Schema, DAO Interfaces |
| 18.3.2013 | 21:00 | 3:00 | HSQL DB Connection und erste DAOs implementiert |
| 02.4.2013 | 0:00 | 2:00 | Service Layer angefangen, DB Jar ausgetauscht, Pferde UI |
| 05.4.2013 | 12:00 | 4:00 | Rechnungs UI, DAO rewrite |
| 06.4.2013 | 17:00 | 6:00 | Versicherungs Anwendungsfall, Rechnungsdruck UI, UI Änderungen, Unit Tests |
| 07.4.2013 | 17:00 | 3:00 | Anwendungsfälle, Dokumentations Formatierung |

# Domänenmodell



# Anwendungsfalldiagramm



# Anwendungsfallbeschreibungen

## Pferd anlegen

**Beschreibung:** Für ein neues Pferd werden Name, Therapieart, Preis, Geschlecht und Bild eingegeben. Dieses Pferd wird anschließend gespeichert.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will ein neues Pferd in der Software erfassen.

**Ablauf:** Der Benutzer gibt einen Name, Therapieart, Preis, Geschlecht und ein Bild für das Pferd an und kann dieses über einen Button abspeichern.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung

**Nachbedingung:** Das Pferd kann nun in der Software gefunden werden.

## Pferd bearbeiten

**Beschreibung:** Für ein bereits existierendes werden Name, Therapieart, Preis, Geschlecht und Bild eingegeben. Dieses Pferd wird anschließend gespeichert.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will ein bereits existierendes Pferd bearbeiten.

**Ablauf:** Der Benutzer öffnet ein Pferd und gibt einen Namen, einen Preis und ein Bild für dieses an und kann es über einen Button abspeichern.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung. Das Pferd kann im Programm gefunden und geöffnet werden.

**Nachbedingung:** Das Pferd wurde verändert und als neues Pferd abgespeichert.

## Pferd löschen

**Beschreibung:** Ein bereits existierendes Pferd wird gelöscht.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will ein bereits existierendes Pferd löschen.

**Ablauf:** Der Benutzer öffnet ein Pferd und kann es über einen Button löschen

**Alternativer Ablauf:** Der Benutzer löscht das Pferd über die Pferdeauswahl.

**Vorbedingung:** Es besteht eine Datenbankverbindung. Das Pferd kann im Programm gefunden und geöffnet werden.

**Nachbedingung:** Das Pferd kann nicht mehr gesucht oder verwendet werden.

## Pferd suchen

**Beschreibung:** Ein Benutzer sucht und öffnet ein existierendes Pferd

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will ein bereits existierendes Pferd suchen und öffnen.

**Ablauf:** Der Benutzer kann eine Attribut angeben und damit Pferde suchen und öffnen.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung. Das Pferd ist in der Datenbank angelegt

**Nachbedingung:** Keine

## Rechnung anlegen

**Beschreibung:** Pferde werden über Therapieeinheiten an eine Rechnung angefügt und diese Rechnung gespeichert.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine Rechnung anlegen.

**Ablauf:** Der Benutzer sucht Pferde und fügt diese an eine Rechnung an. Er legt fest, wie oft welches Pferd therapiert wird, zusätzlich kann er Versichung und Empfänger angeben und speichert die Rechnung.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung. Pferde können im Programm gefunden und geöffnet werden.

**Nachbedingung:** Die Rechnung wurde angelegt und kann im Programm gefunden werden.

## Rechnung bearbeiten

**Beschreibung:** Rechnungen dürfen, solang sie noch nicht abgeschossen oder bezahlt sind bearbeitet werden.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine Rechnung anlegen.

**Ablauf:** Der Benutzer sucht Pferde und fügt diese an eine Rechnung an. Er legt fest, wie oft welches Pferd therapiert wird, zusätzlich kann er Versichung und Empfänger angeben und speichert die Rechnung.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung. Pferde können im Programm gefunden und geöffnet werden, die Rechnung muss im Zustand „Erstellt“ sein.

**Nachbedingung:** Die Rechnung wurde bearbeitet.

## Rechnung austellen

**Beschreibung:** Rechnungen die erstellt wurden müssen bevor sie gezahlt werden, ausgestellt werden.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine Rechnung aussstellen.

**Ablauf:** Der Benutzer sucht eine Rechnung die sich im Status „Erstellt“ befindet aus der Rechnungsliste, öffnet diese und ändert den Status auf „Ausgestellt“.

**Alternativer Ablauf:** Die Rechnung wird direkt beim Erstellen auf den Status ausgestellt gesetzt.

**Vorbedingung:** Es besteht eine Datenbankverbindung. Pferde können im Programm gefunden und geöffnet werden, die Rechnung muss im Zustand „Erstellt“ sein.

**Nachbedingung:** Die Rechnung wurde ausgestellt.

## Rechnung bezahlen

**Beschreibung:** Rechnungen die erstellt wurden müssen bevor sie gezahlt werden, ausgestellt werden.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine Rechnung bezahlen.

**Ablauf:** Der Benutzer sucht eine Rechnung aus der Liste und kann diese per Kopf auf Bezahlt setzen.

**Alternativer Ablauf:** Die Rechnung wird direkt beim Erstellen auf den Status bezahlt gesetzt.

**Vorbedingung:** Es besteht eine Datenbankverbindung. Die Rechnung muss im Zustand „Ausgestellt“ sein.

**Nachbedingung:** Die Rechnung wurde bezahlt.

## Rechnung drucken

**Beschreibung:** Ausdruck einer Rechnung

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine Rechnung ausdrucken.

**Ablauf:** Der Benutzer sucht eine Rechnung aus der Liste und kann diese per Kopf drucken.

**Alternativer Ablauf:** Keiner.

**Vorbedingung:** Es besteht eine Datenbankverbindung. Die Rechnung muss vorhanden sein.

**Nachbedingung:** Die Rechnung kann nun gedruckt werden.

## Rechnung suchen

**Beschreibung:** Eine bereits existierende Rechnung wird gesucht und geöffnet.

**Akteure:** Benutzer

**Auslöser:** Der Benutzer will eine bereits existierende Rechnung suchen und öffnen.

**Ablauf:** Der Benutzer kann aus eine Liste die gewünschte Rechnung öffnen.

**Alternativer Ablauf:** Keiner

**Vorbedingung:** Es besteht eine Datenbankverbindung. Die Rechnung existiert in der Datenbank.

**Nachbedingung:** Keine