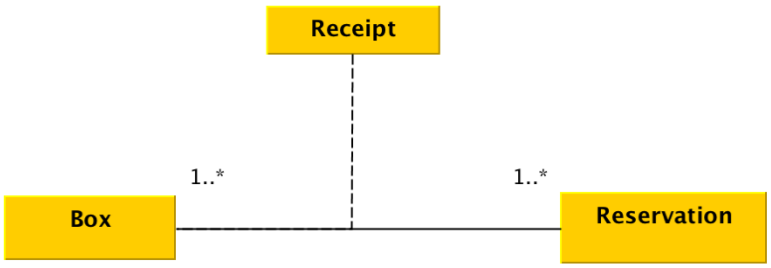
Software Engineering & Projektmanagement, PR 6.0/4.0, Sommersemester 2017
Einzelbeispiel (SEPM/JAVA)
Wendy's Pferdepension
Dokumentation

Manpreet Singh
1428182
e1428182@student.tuwien.ac.at
25. März 2017

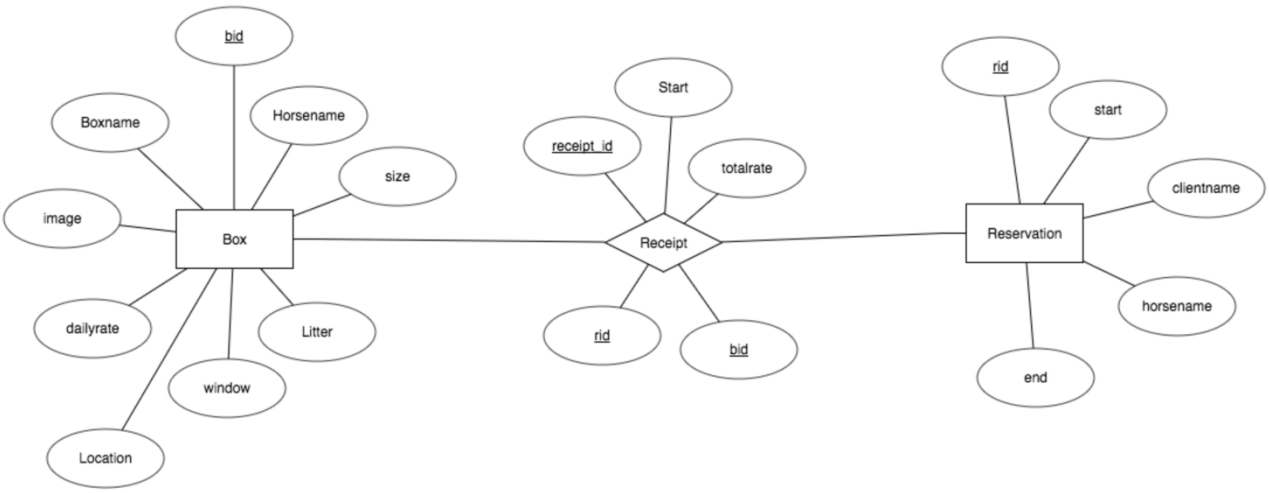
1. Stundenliste

Schritt	Beschreibung	Datum	Aufwand
1	Domänenmodell Erstellen mittels DIA	06.03.2017	1
2	ER-Diagramm Erstellen mittels DIA	06.03.2017	1
3	Anwendungsfalldiagramm erstellen mittels Google Docs	06.03.2017	1,5
4	Erstellung der SQL CREATE & INSERT Script	07.03.2017	2
5	Speichern von Daten in H2 Database	07.03.2017	2
6	JDBC Verbindung zur Database	08.03.2017	2
7	Erstellung von Box, Receipt und Reservation Entities	08.03.2017	1
8	Erstellen von DAO Interface	09.03.2017	3
9	Implementierung von BoxDAOImpl	10.03.2017	4
10	Implementierung von ReservationDAOImpl	10.03.2017	3
11	Implementierung von ReceiptDAOImpl	11.03.2017	2
12	Implementierung von Service-Interfaces	11.03.2017	2
13	Implementierung der Methoden für BoxServiceImpl, ReservationServiceImpl und ReceiptServiceImpl	12.03.2017	5
14	JUnit Test	14.03.2017	4
15	Alle GUI Frames und Controller	15.03.2017-20.03.2017	18
16	Statistik Erstellen	24.03.2017	3
17	Fehlersuche und Tests	06.03.2017-24.03.2017	20
18	Code Dokumentation	25.03.2017	2
19	Dokumentation	25.03.2017	4
		Summe:	80,5

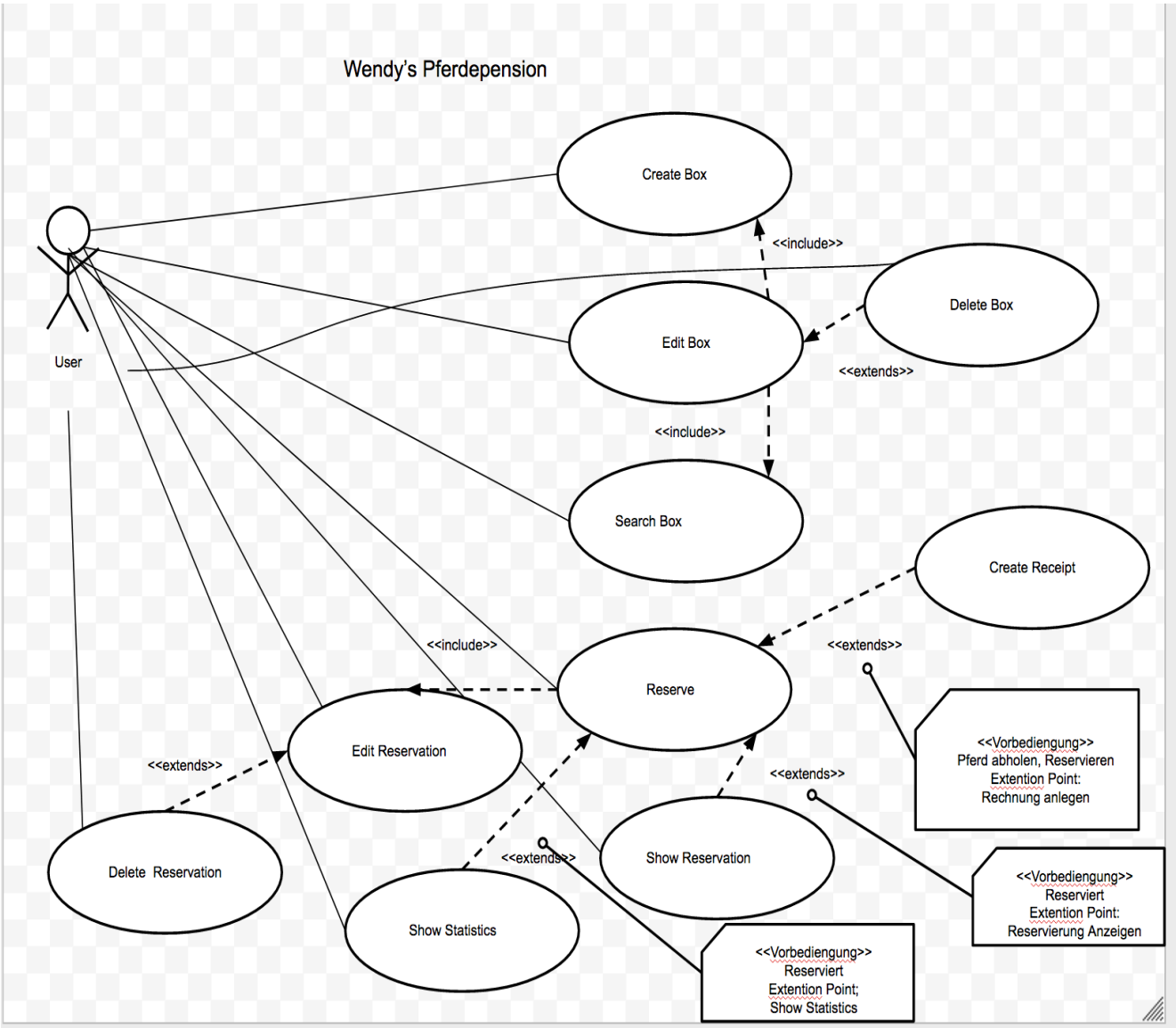
2. Domänenmodell



3. Entity Relationship Diagramm (ER Diagramm)



4. Anwendungsfalldiagramm



5. Anwendungsfall-Beschreibung

5.1. Create Box

5.1.1. Identification Summary

5.1.1.1. Titel: Create Box

5.1.1.2. Scope: Box

5.1.1.3. Level: User Goal

5.1.1.4. Aktoren: User

5.1.1.5. Kurzbeschreibung: Eine neue Box wird angelegt.

5.1.2. Scenarios

5.1.2.1. Vorbedingungen: Es besteht eine DB-Verbindung.

5.1.2.2. Hauptszenario: Der User gibt den Name der Boxes, Name des Pferdes, die Größe, den Tagessatz (bzw. in Preis pro Stunde), Einstreu, die Lage und ob es ein Fenster hat oder nicht in entsprechenden Feld ein und ein Bild hinzufügen zu können, muss der User auf „Choose Image“ klicken, wo er sich ein Bild aussuchen kann. Dann klickt der User auf „Add“ somit werden die eingegebene Daten in Datenbank hinzugefügt.

5.1.2.3. Fehlersituation: Es wurden keine gültigen Daten eingegeben oder Felder sind leer. Es wird eine Fehlermeldung angezeigt.

5.1.2.4. Alternativszenario: Keine.

5.1.2.5. Nachbedingung: Der User kann jetzt das Pferd in der Datenbank finden.

5.1.3. Non-functional constraints: keine.

5.2. Edit Box

5.2.1. Identification Summary

5.2.1.1. Titel: Edit Box

5.2.1.2. Scope: Box

5.2.1.3. Level: User Goal

5.2.1.4. Aktoren: User

5.2.1.5. Kurzbeschreibung: Der User ändert die Daten einer bereits existierenden Box.

5.2.2. Scenarios

5.2.2.1. Vorbedingung: Es besteht eine DB-Verbindung und Box sollte sich im System existieren und der User hat die Box in der liste selektiert.

5.2.2.2. Hauptszenario: Der User klickt auf „Edit box“, welche er bearbeiten möchte. Ein neues Fenster wird geöffnet und der User gibt Daten ein, die er ändern möchte und das Bild ändern zu können, muss der User auf „Choose Image“ klicken und nach dem die Daten eingegeben sind klickt der User auf „Save“ somit werden die Daten in DB geändert.

5.2.2.3. Fehlersituationen: Es wurden keine gültigen Daten eingegeben.

5.2.2.4. Alternativszenario: Keine.

5.2.2.5. Nachbedingung: Der User kann die geänderte Werte der Box sich ansehen.

5.2.3. Non-functional constraints: Keine.

5.3. Search Box

5.3.1. Identification Summary

5.3.1.1. Titel: Search Box

5.3.1.2. Scope: Box

5.3.1.3. Level: User Goal

5.3.1.4. Aktoren: User

5.3.1.5. Kurzbeschreibung: Der User sucht nach bestimmten Box im Datenbank.

5.3.2. Scenarios

5.3.2.1. Vorbedingung: Es besteht eine DB-Verbindung und der befindet sich im System.

5.3.2.2. Hauptszenario: Der User gibt die Kriterien ein nach dem er die Box sucht also Einstreu, die Lage und ob es Fenster hat oder nicht. Die gesuchten Boxen werden aufgelistet und der User kann sich entscheiden ob er die Box bearbeiten will oder Löschen will. Wenn er auf „Edit“

klickt siehe punkt 5.2. Wenn er auf „delete“ klickt, siehe punkt 5.4.

5.3.2.3. Fehlersituationen: Es besteht keine DB-Verbindung.

5.3.2.4. Alternativszenario: Keine.

5.3.2.5. Nachbedingung: Der User findet die gesuchte Box.

5.3.3. Non-functional constraints: Keine.

5.4. Delete Box

5.4.1. Identification Summary

5.4.1.1. Titel: Search Box

5.4.1.2. Scope: Box

5.4.1.3. Level: User Goal

5.4.1.4. Aktoren: User

5.4.1.5. Kurzbeschreibung: Der User entfernt die Box, wenn vorhanden ist

5.4.2. Scenarios

5.4.2.1. Vorbedingungen: Es besteht ein DB-Verbindung und der User hat die Box in der Liste selektiert um zu löschen und die Box, die gelöscht werden muss, befindet sich im System.

5.4.2.2. Hauptszenario: Der User klickt auf „Delete“ Button, dann wird durch ein Dialogfenster gefragt ob er diese Box wirklich löschen möchte. Wenn er dies bestätigt, wird die Box gelöscht und im System als gelöscht markiert.

5.4.2.3. Fehlersituationen: Es besteht keine DB-Verbindung.

5.4.2.4. Alternativszenario: Keine.

5.4.2.5. Nachbedingung: Die Box erscheint nicht im System aber erscheint auf Rechnungen und Reservierungen.

5.4.3. Non-functional constraints: Keine.

5.5. Create Reservation

5.5.1. Indentification Summary

5.5.1.1. Titel: New Reservation

5.5.1.2. Scope: Reservation

5.5.1.3. Level: User Goal

5.5.1.4. Aktoren: User

5.5.1.5. Kurzbeschreibung: Eine neue Reservierung wird angelegt.

5.5.2. Scenarios

5.5.2.1. Vorbedingung: Es besteht eine DB-Verbindung

5.5.2.2. Hauptszenario: Der User gibt Name der Kunde ein, Start Datum und Zeit, End Datum und Zeit, sucht die Name des Pferdes aus Liste und klickt auf Button „check“ um nachzusehen ob das Pferd in gegebenen Zeitraum verfügbar ist oder nicht. Der User klickt dann auf Button „Reserve“ und somit wird die Reservierung im System gespeichert.

5.5.2.3. Fehlersituationen: Es wurde keine gültigen Daten eingeben oder die Felder sind Leer. Eine Fehlermeldung wird angezeigt.

5.5.2.4. Alternativszenario: Keine.

5.5.2.5. Nachbedingungen: Der User kann die Reservierungen im System ansehen.

5.5.3. Non-funktional constraints: Keine.

5.6. Edit Reservation

5.6.1. Identification Summary

5.6.1.1. Titel: Edit Reservation

5.6.1.2. Scope Reservation

5.6.1.3. Level: User Goal

5.6.1.4. Aktoren: User

5.6.1.5. Kurzbeschreibung: Der User ändert die Werte einer bereits existierenden Reservierung.

5.6.2. Scenarios

5.6.2.1. Vorbedingungen: Es besteht die DB-Verbindung. Reservierung die man bearbeiten möchte befindet sich im System und der selektiert die zu bearbeitende Reservierung.

5.6.2.2. Hauptszenario: Der User klickt auf „Edit“ und es erscheint ein neues Fenster. Der User gibt Werte ein, die er ändern möchte und dann klickt er auf „Save“ und die Änderungen werden gespeichert.

5.6.2.3. Fehlersituationen: Es wurden keine gültigen Daten eingegeben.
5.6.2.4. Alternativszenario: Keine.
5.6.2.5. Nachbedingung: Der User kann die geänderte Werte ansehen.

5.6.3. Non-functional constraints: Keine.

5.7. Delete Reservation

5.7.1. Identification Summary

5.7.1.1. Titel: View Reservations
5.7.1.2. Scope: Reservation
5.7.1.3. Level: User Goal
5.7.1.4. Aktoren: User
5.7.1.5. Kurzbeschreibung: Der User löscht die ausgewählte Reservierung.

5.7.2. Scenarios

5.7.2.1. Vorbedingungen: Es besteht eine DB-Verbindung.
5.7.2.2. Hauptszenario: Der User klickt auf Button „Delete“, wenn er eine Reservierung ausgewählt hat. Er erscheint ein Dialog Fenster, das dem User nachfragt ob er die Reservierung wirklich löschen will und bestätigt er dies, dann wird diese gelöscht.
5.7.2.3. Fehlersituation: Es besteht keine DB-Verbindung.
5.7.2.4. Alternativszenario: Keine.
5.7.2.5. Nachbedingung: Die Reservierung wird gelöscht.

5.7.3. Non-functional constraints: Keine.

5.8. Search Reservation

5.8.1. Identification Summary

5.8.1.1. Titel: View Reservations
5.8.1.2. Scope: Reservation
5.8.1.3. Level: User Goal
5.8.1.4. Aktoren: User
5.8.1.5. Kurzbeschreibung: Der User gibt ein Zeitraum ein, wonach er Reservierungen sucht.

5.8.2. Scenarios

5.8.2.1. Vorbedingung: Es Besteht eine DB-Verbindung.

5.8.2.2. Hauptszenario: Der User klickt auf „View Reservations“ und gibt ein Start und End Datum ein. Nach dem klickt er auf „Search“ und es werden Reservierungen angezeigt, die diesen gegebenen Zeitraum entsprechen.

5.8.2.3. Fehlersituationen: Es besteht keine DB-Verbindung.

5.8.2.4. Nachbedingung: Der User kann die gesuchte Reservierung entweder löschen oder bearbeiten.

5.8.3. Non-functional constraints: Keine.

5.9. Receipts

5.9.1. Identification Summary

5.9.1.1. Titel: Receipt

5.9.1.2. Scope: Receipt

5.9.1.3. Level: User Goal

5.9.1.4. Aktoren: User

5.9.1.5. Kurzbeschreibung: Der sieht die aktuellste bezahlte und vorherig bezahlte Reservierungen an.

5.9.2. Scenarios:

5.9.2.1. Vorbedingung: Es besteht eine DB-Verbindung.

5.9.2.2. Hauptszenario: Nach dem eine Reservierung bezahlt worden ist, dann lässt sich das, durch klick auf „Receipt“ Button, anzeigen.

5.9.2.3. Fehlersituationen: Es besteht keine DB-Verbindung.

5.9.2.4. Alternativszenario: Keine.

5.9.2.5. Nachbedingung: Keine.

5.9.3. Non-functional constraints: Keine.

5.10. Show Statistics

5.10.1. Identification Summary

5.10.1.1. Titel: Show Statistics

5.10.1.2. Scope: Static Evaluation

5.10.1.3. Level: User Goal

5.10.1.4. Aktoren: User

5.10.1.5. Kurzbeschreibung: Der sieht die statistische Auswertung grafisch, welche zeigt an welchem

Wochentag eine bestimmte Box bzw. ein Pferd reserviert worden ist und der Box, der am besten gebucht ist.

5.10.2. Scenarios

5.10.2.1. Vorbedingungen: Es besteht ein DB-Verbindung.

5.10.2.2. Hauptszenario: Der User klick auf Menü die Option „View Statistics“ und es öffnet sich ein neues Fenster und der sucht sich ein Pferd aus klickt auf „Show statistics“ und es wird grafisch dargestellt am welchem Tag und wie oft wurde dieses Pferd gebucht worden. Weiteres kann der User das Button „best booked“ klicken und es werden alle Box grafisch dargestellt.

5.10.2.3. Fehlersituationen: Es besteht keine DB-Verbindung

5.10.2.4. Alternativszenario: Keine.

5.10.2.5. Nachbedingung: Keine.

5.10.3. Non-Functional Constraints: Keine.