

# **UE Software Engineering 050052 WS 2015/16**

LV-Leiter: Dipl.-Ing. Dr. Hans Moritsch

**Projektname: FLY HIGH**

**Projektteam:**

Nachname	Vorname	Matrikelnummer	E-Mail-Adresse
Binder	Paul	1401147	a1401147@unet.unvie.ac.at
Dangl	Philipp	1400930	a1400930@unet.unvie.ac.at
Mittendorfer	Patrick	1408492	a1408492@unet.unvie.ac.at
Myroshnykova	Olena	1408516	a1408516@unet.unvie.ac.at

**CEWebS-Teamseite:**

[https://cewebs.cs.univie.ac.at/SWE/ws15/index.php?m=D&t=info&c=show&CEWebS\\_c=g050052-3t4](https://cewebs.cs.univie.ac.at/SWE/ws15/index.php?m=D&t=info&c=show&CEWebS_c=g050052-3t4)

Datum: 10.11.2015

# 1. Funktionale Anforderungen

Die funktionalen Anforderungen wurden durch Brainstorming bei einer Teamsitzung erhoben. In unserem Team befinden sich Mitglieder die bereits Erfahrung mit der Erstellung von größeren Softwareprojekten haben.

## 1.1. Funktionalität

### 1.1.1. Fluggesellschaftsmitarbeiter

#### 1.1.1.1. *Flug eintragen*

Der Mitarbeiter kann neue Flüge eintragen. Dabei muss er den Startflughafen, den Zielflughafen, den Flugzeugtyp, die Flugzeit und den Preis für die Tickets festlegen

#### 1.1.1.2. *Flug bearbeiten*

Der Mitarbeiter kann, solange der Flug noch nicht innerhalb 3 Monate stattfindet, den Flug bearbeiten.

### 1.1.2. Kunde

#### 1.1.2.1. *Flug suchen und Ticket reservieren*

Ein Kunde sucht mit den gewählten Suchoptionen nach passenden Flügen. Wenn in der danach angezeigten Liste nach den Suchkriterien ein für den Kunden passenden Flug dabei ist, kann der Kunde für diesen Flug ein Ticket reservieren. Dabei muss er die Beförderungsklasse, Sitzplatz und die Zahlungsart wählen. Danach erhält der Kunde nach dem Bezahlen des Tickets die Option das Ticket auszudrucken.

#### 1.1.2.2. *Flug bearbeiten*

Der Kunde kann, nachdem er einen Flug gebucht hat, den Flug bearbeiten oder stornieren.

### 1.1.3. Stammkunde

#### 1.1.3.1. *Flug suchen und Ticket reservieren mit Verwendung des Vielflugmeilensystems*

Ein Kunde sucht mit den gewählten Suchoptionen nach passenden Flügen. Wenn in der danach angezeigten Liste nach den Suchkriterien ein für den Kunden passenden Flug dabei ist, kann der Kunde für diesen Flug ein Ticket reservieren. Dabei muss er die Beförderungsklasse, Sitzplatz und die Zahlungsart wählen. Falls der Kunde bereits mehr als 10.000 Meilen hat und diese nicht länger als 1 Jahr in der Vergangenheit liegen, kann er wählen, welche Rabattstufe er haben möchte. Diese sind immer nur in 1.000er Schritten zu je 1 % Rabatt möglich mit einer Obergrenze von 30%. Danach erhält der Kunde nach dem Bezahlen des Tickets die Option das Ticket auszudrucken.

#### 1.1.3.2. *Technische Voraussetzungen*

Kunden bekommen für jeden Flug Flugmeilen. Diese werden gespeichert und zusammengezählt. Erreichte Meilen können erst ab > 10.000 eingelöst werden. Man kann immer nur in 1.000er Schritten Meilen einlösen. Für jeden 1.000er Schritt bekommt man 1 % Rabatt für den gewählten Flug. Maximal können nur 30.000 Meilen pro Flug eingelöst werden. Jede erhaltene Flugmeile für einen Flug die  $\geq 1$  Jahr in der Vergangenheit liegt wird verworfen.

#### 1.1.4. Statistiker

##### 1.1.4.1. *Flugdaten und Kundendaten abrufen*

Der Statistiker kann jederzeit die Flugdaten und Kundendaten abrufen. Er kann aus den danach erhaltenen Daten Statistiken bzw. Mobilitätsverhalten auswerten.

## 1.2. Bedienoberfläche

**Startseite**

Suchoptionen

Aktionen


Milen

private Daten

Abflughafen

Zielflughafen

suchoptionen



## 2. Use-Case-Modell

### Primäre Use Cases:

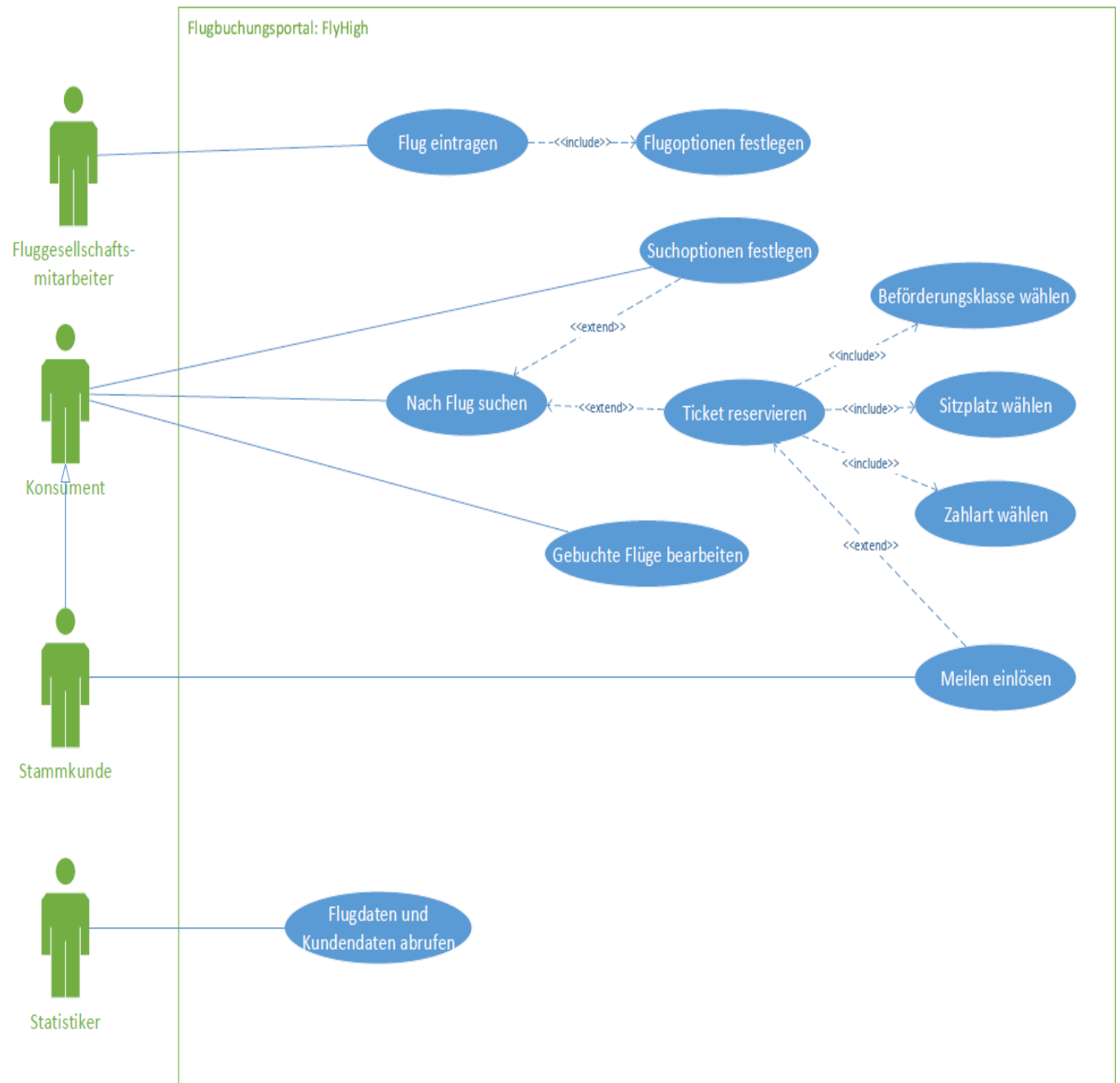
- Flug eintragen
- Flugoptionen festlegen
- Nach Flug suchen
- Meilen einlösen
- Ticket reservieren

### Sekundäre Use Cases:

- Gebuchte Flüge bearbeiten
- Beförderungsklasse wählen
- Sitzplatz wählen
- Suchoptionen festlegen
- Zahlart wählen
- Flugdaten und Kundendaten abrufen



## 1.1. Use-Case-Diagramm



## 1.2. Use-Case Beschreibungen

### Use-Case 1: Flug eintragen

<b>Ziel</b>	Einen Flug in unserem Datensatz aufnehmen
<b>Kurzbeschreibung</b>	Es soll möglich sein, Flüge einzutragen
<b>Vorbedingung</b>	Der Fluggesellschaftsmitarbeiter muss eingeloggt sein
<b>Nachbedingung, bei Erfolg</b>	Flug eingetragen; Flugoptionen festlegen
<b>Fehlersituationen</b>	Flugnummer wurde mehrfach eingetragen
<b>Nachzustand im Fehlerfall</b>	Neue Eingabe erwartet
<b>Akteure</b>	Fluggesellschaftsmitarbeiter
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Neue Fluginformation von Flugunternehmen erhalten
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Fluginformationen überprüfen 2. Flug erstellen
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

The image shows a graphical user interface for entering airport information. The window is titled "Flughafen eintragen". It contains four labeled input fields: "Country" (a dropdown menu with "Austria" selected), "Name", "Klass", and "Contact nummer". At the bottom of the window are two buttons: "Cancel" and "Ok".

**Flug eintragen**

Startflughafen:  Zielflughafen:

Flugzeugtype:

Flugzeit: Von  Bis

Manage	Type von Price	Price
<input checked="" type="checkbox"/>	Business	300
<input checked="" type="checkbox"/>	Econom	200

## Use-Case 2: Flugoptionen festlegen

<b>Ziel</b>	Flugoptionen zu einem Flug angeben
<b>Kurzbeschreibung</b>	Zu einem Flug werden Daten, wie Startflughafen, Zielflughafen, Dauer, Meilen, usw. festgelegt
<b>Vorbedingung</b>	Der Fluggesellschaftsmitarbeiter muss eingeloggt sein & Flug eingetragen
<b>Nachbedingung, bei Erfolg</b>	Flugoptionen eingetragen;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Fluggesellschaftsmitarbeiter
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Ein neuer Flug wurde eingetragen
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Erweiterte Flugoptionen eintragen
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	



Flugoptionen			
Dauer	<input type="text"/>	Model	<input type="text"/>
Milen	<input type="text"/>	Belastungsfähigkeit	<input type="text"/>
Startflughafen	Schwechat ▼	Baujahr	<input type="text"/>
Endflughafen	Malpensa ▼		
			<input type="button" value="Save"/>

### Use-Case 3: Nach Flug suchen

<b>Ziel</b>	Flüge anzeigen
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, nach Flügen zu suchen und diese in einer Liste angezeigt zu bekommen
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein
<b>Nachbedingung, bei Erfolg</b>	Flugliste angezeigt;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer möchte mittels Flugzeug transportiert werden
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt die Flugsuche aus 2. Flugliste wird angezeigt
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Nach Flug suchen

von  Nach

Date

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
6	7	8	9	10	11	12

Company	Zeit	Flughafen	Price	Choose
Austrian Airlines	7:00-08.25	Malpensa	105	<input checked="" type="checkbox"/>
Austrian Airlines	19:00-20.25	Malpensa	150	<input checked="" type="checkbox"/>
Lufthansa	04.05-05.10	Malpensa	96	<input checked="" type="checkbox"/>
Wizzair	06.15-07.20	Malpensa	60	<input checked="" type="checkbox"/>



#### Use-Case 4: Suchoptionen festlegen

<b>Ziel</b>	Suchoptionen für die Flugsuche festlegen
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, Suchoptionen für die Flugsuche einzugeben
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & sich in der Flugsuche-Maske befinden
<b>Nachbedingung, bei Erfolg</b>	Flugliste wird angepasst;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer möchte die Flugsuche verfeinern
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	<ol style="list-style-type: none"> <li>Benutzer wählt Suchoptionen aus z.B. Non-Stop, etc.</li> <li>Aktualisierte Flugliste wird angezeigt</li> </ol>
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Search form

☐ Round Trip ☐ Direktflug bevorzugt

Abflughafen  Zielflughafen

Abflugdatum  Rückflugdatum   +/- some tage

Klasse

Reisende

## Use-Case 5: Ticket reservieren

<b>Ziel</b>	Ein Ticket für einen Flug reservieren
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, ein Ticket für einen bestimmten Flug zu reservieren
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & in der Flugsuch-Maske einen Flug ausgewählt haben
<b>Nachbedingung, bei Erfolg</b>	Ticketreservierung vorgemerkt;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer will ein Flugticket kaufen
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt Ticket-Reservierung aus 2. Ticket-Reservierung wird erstellt
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Ticket reservieren

Nachnahme

Vornahme

Adresse

☐ EU BewohnerIn

Reisepass
▼

Pass nummer

☐ Haben miles

summe

Reservieren

## Use-Case 6: Beförderungsklasse wählen

<b>Ziel</b>	Eine bestimmte Beförderungsklasse für einen Flug auswählen
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, eine Beförderungsklasse für einen vorreservierten Flug auszuwählen
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & in der Flugsuch-Maske einen Flug ausgewählt haben & sich in der Ticketreservierung befinden
<b>Nachbedingung, bei Erfolg</b>	Vorgemerkte Flugreservierung wird aktualisiert;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer will ein Ticket reservieren
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt eine Beförderungsklasse aus 2. Ticket wird aktualisiert
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Beförderungsklasse wählen

business ▼

Weiter

## Use-Case 7: Sitzplatz wählen

<b>Ziel</b>	Eine bestimmten Sitzplatz für einen Flug auswählen
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, eine Sitzplatz für einen vorreservierten Flug auszuwählen
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & in der Flugsuch-Maske einen Flug ausgewählt haben & sich in der Ticketreservierung befinden
<b>Nachbedingung, bei Erfolg</b>	Vorgemerkte Flugreservierung wird aktualisiert;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer will ein Ticket reservieren
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt einen Sitzplatz aus 2. Ticket wird aktualisiert
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Sitze wählen


Sitz nummer


Submit


## Use-Case 8: Zahlart wählen


<b>Ziel</b>	Eine bestimmte Zahlart für einen Flug festlegen
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, eine Zahlart für einen vorreservierten Flug anzugeben
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & in der Flugsuch-Maske einen Flug ausgewählt haben & sich in der Ticketreservierung befinden
<b>Nachbedingung, bei Erfolg</b>	Vorgemerkte Flugreservierung wird aktualisiert;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer will ein Ticket reservieren
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt eine Zahlart aus 2. Ticket wird aktualisiert
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	


### Zahlart wählen


☐ Sofortüberweisung


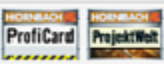
☐ Kreditkarte


☐ Bankeinzug


☐ Vorauszahlung per Überweisung


☐ PayPal


☐ Finanz-Kauf


☐ Kundenkarte


## Use-Case 9: Meilen einlösen

<b>Ziel</b>	Ein Stammkunde soll bestimmte Rabatte bekommen
<b>Kurzbeschreibung</b>	Es soll einem Stammkunden möglich sein,

	Meilen einzulösen und für diese Rabatte zu erhalten
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & muss ein Stammkunde sein & in der Flugsuch-Maske einen Flug ausgewählt haben & sich in der Ticketreservierung befinden
<b>Nachbedingung, bei Erfolg</b>	Vorgemerkte Flugreservierung wird aktualisiert;
<b>Fehlersituationen</b>	KEINE
<b>Nachzustand im Fehlerfall</b>	KEINE
<b>Akteure</b>	Stammkunde
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer möchte einen Vorteil anhand seiner erreichten Meilen einlösen
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	1. Benutzer wählt in 1000er Blöcken seine Meilen aus 2. Ticket wird aktualisiert
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	

Milen

Nachnahme

Vorname

History von Miles

Date	Ticket	Price	Milen
2015-01-01	Wien-Linz	200	100
2015-04-05	Wien-Milan	220	110

Gesamtsumme 210

## Use-Case 10: Gebuchte Flüge bearbeiten

<b>Ziel</b>	Gebuchte Flüge bearbeiten
<b>Kurzbeschreibung</b>	Es soll einem Konsumenten möglich sein, seine bereits gebuchten Flüge zu bearbeiten, z.B. Ticket stornieren
<b>Vorbedingung</b>	Der Konsument muss eingeloggt sein & einen

	Flug zur Bearbeitung ausgewählt haben
<b>Nachbedingung, bei Erfolg</b>	Buchung wird aktualisiert;
<b>Fehlersituationen</b>	Keine Änderung mehr möglich
<b>Nachzustand im Fehlerfall</b>	Mitteilung an User;
<b>Akteure</b>	Konsument
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Benutzer möchte etwas an seinen bestehenden Buchungen verändern
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	<ol style="list-style-type: none"> <li>1. Benutzer wählt einen bereits gebuchten Flug aus</li> <li>2. Prüfung ob Flug noch geändert werden darf</li> <li>3. Änderungen durchführen</li> <li>4. Ticket wird reserviert</li> </ol>
<b>Alternativabläufe #1, #2, ..., als Folgen von Aktionen</b>	<ol style="list-style-type: none"> <li>1. Benutzer wählt einen bereits gebuchten Flug aus</li> <li>2. Prüfung ob Flug noch geändert werden darf</li> <li>3. Änderungen nicht mehr möglich</li> <li>4. Mitteilung an User</li> </ol>

Ticket stornieren

Ticket Nummer	Date	Name	Manage
011101	15-08-2015	Karl	<a href="#">x</a>
011101	15-08-2015	Monica	<a href="#">x</a>

Submit

## Use-Case 11: Flugdaten und Kundendaten abrufen

<b>Ziel</b>	Auswertungen über das Flug/Konsumenten-Verhalten durchzuführen
<b>Kurzbeschreibung</b>	Es soll einem Statistiker möglich sein, Flug- und Konsumentendaten aus dem Portal abzurufen und auszuwerten
<b>Vorbedingung</b>	Der Statistiker muss eingeloggt sein
<b>Nachbedingung, bei Erfolg</b>	Daten wurden exportiert;
<b>Fehlersituationen</b>	Keine Daten vorhanden
<b>Nachzustand im Fehlerfall</b>	Mitteilung an User;
<b>Akteure</b>	Statistiker
<b>Trigger: Auslösendes Ereignis (außer Benutzereingaben)</b>	Ein(e) StatistikerIn möchte Daten über das Flugbuchungsportal auswerten
<b>Basisablauf (Standardablauf) als Folge von Aktionen: 1., 2., 3., ...</b>	<ol style="list-style-type: none"> <li>1. StatistikerIn wählt Daten für Export aus</li> <li>2. Daten werden exportiert</li> </ol>



	3. StatistikerIn erhält Daten
Alternativabläufe #1, #2, ..., als Folgen von Aktionen	

Mobilitätsverhalten auswerten

Flug

Von  Bis

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
6	7	8	9	10	11	12

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
6	7	8	9	10	11	12

Prices

Durchschnittspreis 240 euro

Cancel Ok

## Beziehungen zw. den einzelnen Use Cases / Generalisierung Akteure

Use Case / Akteur	Beziehung
Flug eintragen	<b>Flugoptionen festlegen (include)</b> Zu jedem Flug müssen Flugoptionen eingegeben werden – ein „leerer“ Flug kann nicht existieren!
Nach Flug suchen	<b>Suchoptionen festlegen (extend)</b> Optional soll es möglich sein, Flüge über eine Suchoption zu filtern <b>Ticket reservieren (extend)</b> Optional muss es möglich sein, nach der Flugauswahl, ein Ticket für einen Flug zu reservieren
Ticket reservieren	<b>Beförderungsklasse wählen (include)</b> Bei einer Ticketreservierung muss eine Beförderungsklasse ausgewählt werden <b>Sitzplatz wählen (include)</b> Bei einer Ticketreservierung muss ein Sitzplatz gewählt werden <b>Zahlart wählen (include)</b>

	<p>Bei einer Ticketreservierung muss eine Zahlart ausgewählt werden</p> <p><b>Meilen einlösen (extend)</b></p> <p>Einen Stammkunden soll die Möglichkeit haben bei der Ticketreservierung seine Meilen einzulösen.</p>
<b>Konsument (Akteur)</b>	<p><b>Stammkunde (Akteur - Generalisierung)</b></p> <p>Stammkunde ist eine Ausprägung vom Konsumenten, der als zusätzliche Funktion Meilen einlösen kann.</p>

# **Nichtfunktionale Anforderungen**

## **1.1. Qualitätsanforderungen**

Das System soll möglichst Benutzerfreundlich aufgebaut werden. Neue User sollen sich schnell mit der Applikation zurecht finden. Die Plattform muss zuverlässig arbeiten, das heißt, dass Daten nicht verloren gehen dürfen. Um keine langen Wartezeiten zu erzeugen, soll das System möglichst effizient agieren.

## **1.2. Technische Anforderungen**

Es handelt sich um eine Web-Applikation die unter Verwendung von Java Servlets entwickelt werden soll. Die Plattform muss auf einen Tomcat-Server funktionieren.

## **1.3. Realisierungsanforderungen**

Die Installation sollte möglichst einfach funktionieren.

## **1.4. Diverses**

Durch die Verwendung von einer möglichst einfachen Datenspeicherung und keiner relationalen Datenbank, ist die Applikation nicht für Massendatenspeicherungen ausgelegt.