EasyBook - Requirements Definition

A booking system for the Västerås Flygmuseum

Version	Type	Date	Name	Description
0.1	Draft		Robert Engelmann	First draft
0.2	Update	2014-11-12	Sebastian Kunze	Added Use Cases
0.3	Update	2014-11-12	Sebastian Kunze	Added use Case Diagrams
1.0	Review	2014-11-13	Sebastian Kunze	Finalized latest version
1.1	Update	2014-11-23	Robert Engelmann	Added feedback from customer
1.2	Update & Review	2014-11-26	Sebastian Kunze	Updated non-functional requirements and reviewed feedback from customer
1.3	Update	2014-11-29	Robert Engelman	Renamed the role “user” to “customer” in textual contexts.
1.4	Update	2015-01-08	Endri Azizi & Valerio Lucantonio	Overall Review
1.5	Update	2015-01-15	Sebastian Kunze	Added section describing authorization

Table of Contents

1. Description and Definitions

1.1. General Description

We are to develop a new booking system for the Västerås Flygmuseum. The aim is to provide an easy to use, intuitive interface to book the flight simulators, which are operated by the museum. The system will unburden the staff and reduce bureaucratic effort, while at the same time provide more comfort for the museums customers and flight instructors.

The customers will be able to select their favorite flying times via a web interface. If it is outside of the normal opening hours, the coordinator will arrange an instructor and confirm the booking with the customer. The instructors will have access to all the booking data, so they are always up to date and know what is going on. The people operating the cash point can also create bookings for people who are spontaneously visiting the museum. The system will be deployed on a server in the museum and available over the internet. In this way we will ensure an undisturbed flow of information.

The following sections contain detailed information on the actors, functional requirements (in form of Use Cases) and non-functional requirements. Additionally, a list of all User Stories from the Product Backlog is attached.

1.2 Actors

Customer

A customer of the systems is a customer of the museum who wants to fly one of the museum's simulators. To fly a simulator, the customer needs to reserve it beforehand. During the opening hours of the museum, the customer simply books an available slot for a certain simulator. Outside the opening hours of the museum, the customer needs to request a simulator for a specific timespan. This should be as easy as possible and not be constrained by any registration process.

Following Use Cases can be performed by the customer:

- Book simulator
- Cancel booking

Instructor

An instructor is a member of the museum and gives instructions to assigned customers that fly a particular simulator. The system provides the instructor an overview of today's booking for all simulators. If a customer wants to fly a simulator for some more time, the instructor can edit his assigned booking for today.

Following Use Cases can be performed by the instructor:

- Log in
- View booking

- Assign booking to herself/himself
- Prolong booking

Coordinator A coordinator takes booking requests by mail or phone and at the desk and adds them to the system. They have an overview of all bookings including the personal data and are also able to edit and even to delete bookings, if required. Whenever a customer requests a slot outside the opening hour of the museum, it is the coordinator's job to confirm or deny it. He can assign an instructor who is being notified then.

Following Use Cases can be performed by the coordinator:

- Log in
- View booking
- Assign booking
- Prolong booking
- Confirm booking
- Create booking
- Delete booking
- Edit booking

Admin An administrator takes care of all administrative tasks. He manages the system’s accounts and is able to change the system’s variables.

Following Use Cases can be performed by the administrator:

- Log in
- View booking
- Assign booking
- Prolong booking
- Confirm booking
- Create booking
- Delete booking
- Edit booking
- Create account
- Disable account
- Edit account
- Change price
- Change time

1.3 Authorization

At Västerås Flygmuseum, tasks are more dynamic. A staff member might work at the cashpoint one day and give instruction to a simulator the other day. This makes it relatively hard to assign the introduced roles in section 1.2 to staff members. For this particular reason, a derived feature based authorization system was introduced. Compared to a role based authorization approach, feature based authorization systems allow to make a given feature available to any registered user. This approach, however, takes a lot more effort to manage and maintain.

The implemented derived feature based authorisation system comes with a set of predefined roles that map the desired behaviour in section 1.2. Each actor has a list of features that can be accessed. The system allows to edit this list in order to manage the dynamic tasks at the museum. Also, actors can be added, edited and removed at any time later.

2 Functional requirements

The following functional requirements describe the system's behaviour in respect to the Västerås Flygmuseum:

2.1 Use Cases

2.1.1 Customer

A customer is someone who wants to fly one of the museum's simulators.

2.1.1.1 Book simulator

Use Case	Book simulator
Actor	System, Customer, Coordinator
Precondition	/
Main path:	1. Customer selects new booking on website. 2. Customer selects a simulator. 3. Customer sees the simulator's availability. 4. Customer selects a free time slot during the opening hours of the museum. 5. Customer enters her/his name, email address and phone number. 6. Customer provides additional information. 7. Customer sees pricing information. 8. Customer sends all booking information. 9. System verifies provided information. 10. System sends notification to customer.
Alternative path (A1):	3. Customer selects time span outside the opening

	hours of the museum
Alternative path (A2):	9. System verifies provided information. a. System reminds customer that its name, email address, phone number or an optional postal address for the corresponding invoice is incorrect. b. Customer corrects name, email address, phone number or an optional postal address for the corresponding invoice accordingly. c. Customer sends all booking information again.
Alternative path (A3):	10. System send confirmation to coordinator. a. Coordinator confirms booking. System sends notification to customer.
Alternative Path (A4)	5. Customer enters her/his name, email address and phone number as well as a an optional postal address for the corresponding invoice.

2.1.1.2 Cancel booking

Use Case	Cancel booking
Actor	System, Customer
Precondition	Book simulator
Main path:	1. Customer selects *cancel booking* on website. 2. Customer enters booking ID and her/his name 3. Customer clicks *cancel* 4. System sends notification to use
Alternative path (A1):	2. Customer forgot her/his booking ID a. Customer calls museum b. Coordinator cancels booking

2.1.2 Instructor

An instructor is a trained staff member of the museum who assists a customer during his simulator flight.

2.1.2.1 Log in

Use Case	Log in
Actor	System, Instructor

Precondition	/
Main path:	1. Instructor selects *log in* on the website. 2. Instructors enters her/his username and password. 3. Instructor clicks *log in.* 4. System authenticates instructor. 5. Systems forwards instructor to the *calendar*.
Alternative path (A1):	2. Instructor forgot her/his password a. Instructor clicks *forgot password* b. Administrator resets password

2.1.2.2 View booking

Use Case	View bookings
Actor	Instructor
Precondition	● Log in ● Book simulator / Create booking
Main path:	1. Instructor clicks *bookings* on website.

2.1.2.3 Assign booking to herself/himself

Use Case	Assign booking to herself/himself
Actor	System, Instructor, Customer
Precondition	● Log in ● Book simulator / Create booking ● View bookings
Main path:	1. Instructor clicks on a highlighted unassigned booking. 2. Instructor sees date, time span, name, email address, phone number and an optional postal for the corresponding invoice address of the customer. 3. Instructor clicks *assign to me*.
Alternative path (A1):	3. Instructor clicks *assign to me*. a. Systems reminds instructor that she/he is already assigned to another booking. b. Instructor confirms her/his change. c. System removes instructor from the current booking. d. System assigns instructor to the chosen

	booking.

2.1.2.4 Prolong booking

Use Case	Prolong booking
Actor	System, Instructor
Precondition	● Book simulator ● Assign booking ● View bookings
Main path:	1. Instructor selects her/his currently ongoing booking. 2. Instructor clicks *prolong*. 3. Instructor selects end time.
Alternative path (A1)	2. Instructor clicks *prolong.* a. System reminds instructor that she/he is already assigned to an another booking. b. Instructor confirms prolonging the current booking.

2.1.3 Coordinator

A coordinator is a staff member of the museum who manages simulator bookings.

2.1.3.1 Confirm booking

Use Case	Confirm booking
Actor	System, Coordinator, Instructor, Customer
Precondition	● Log in ● Book simulator / Create booking ● View bookings
Main path:	1. Coordinator clicks on a highlighted unconfirmed booking outside the opening hours of the museum. 2. Coordinator sees date, time span, name, email address, phone number and an optional postal address for the corresponding invoice of the customer. 3. Coordinator clicks *confirm*. 4. System notifies customer.
Alternative path (A1):	3. Coordinator assigns an instructor. 4. Coordinator clicks *confirm.* 5. System notifies instructor and customer.

2.1.3.2 Create booking

Use Case	Create booking
Actor	Customer, Instructor, Coordinator
Precondition	● Log in
Main path:	1. Coordinator clicks *new booking* on website. 2. Coordinator enters name, email address, phone number and an optional postal address for the corresponding invoice of the customer. 3. Coordinator provides additional information. 4. Coordinator sees pricing information 5. Coordinator sends booking 6. System verifies information 7. System sends notification
Alternative path (A1):	5. Coordinator assigns instructor 6. Coordinator sends booking 7. System verifies information 8. System sends notification

2.1.3.3 Delete booking

Use Case	Delete booking
Actor	System, Instructor, Coordinator
Precondition	● Log in ● Book simulator / Create booking ● View booking
Main path:	1. Coordinator selects a booking. 2. Coordinator clicks *delete*. 3. System shows a pop-up that says *are you sure that you want to delete this booking*. 4. Coordinator clicks *yes*. 5. System notifies the customer
Alternative path (A1):	5. System notifies the customer and the assigned instructor

2.1.3.4 Edit booking

Use Case	Edit booking

Actor	System, Coordinator
Precondition	● Log in ● Book simulator / Create booking ● View bookings
Main path:	1. Coordinator selects a booking. 2. Coordinator changes name, email address, phone number and/or an optional postal address for the corresponding invoice. 3. Coordinator clicks *confirm.* 4. System validates provided information.
Alternative path (A1):	2. Coordinator changes date and/or time a. System notifies the customer and the assigned instructor
Alternative path (A2):	4. System verifies provided information. a. System remind coordinator that its name, email address, phone number and/or an optional postal address for the corresponding invoice is incorrect. b. Coordinator corrects name, email address, phone number and/or an optional postal address for the corresponding invoice accordingly. c. Coordinator clicks *confirm* again.

2.1.3.5 Assign booking

Use Case	Assign booking
Actor	System, Instructor, Coordinator
Precondition	● Log in ● Book simulator / Create booking ● View bookings
Main path:	1. Coordinator selects a booking. 2. Coordinator chooses an instructor from the drop-down menu. 3. Coordinator clicks *confirm*. 4. System notifies the instructor.
Alternative path (A1):	3. Coordinator clicks *confirm*. a. System reminds the coordinator that chosen

	instructor is already assigned to an another booking. b. Coordinator confirms her/his change. c. System removes instructor from the current booking. d. System assigns instructor to chosen booking.

2.1.4 Administrator

A administrator is a special staff member of the museum who manages all functionality and variables of the system.

2.1.4.1 Create account

Use Case	Create account
Actor	System, Administrator
Precondition	● Log in
Main path:	1. Administrator clicks *settings* on website. 2. Administrator clicks *accounts*. 3. Administrator clicks *create*. 4. Administrator provides name and email address. 5. Administrator chooses a role. 6. Administrator clicks *confirm*. 7. System verifies name and email address.
Alternative path (A1):	7. System verifies name and email address. a. System remind administrator that name and/or email address is incorrect. b. Administrator corrects name and/or email address accordingly. c. Administrator clicks *confirm* again.

2.1.4.3 Disable account

Use Case	Disable account
Actor	Administrator
Precondition	● Log in
Main path:	1. Administrator clicks *settings* on website. 2. Administrator clicks *accounts*. 3. Administrator selects an account. 4. Administrator clicks *disable*.

2.1.4.3 Edit account

Use Case	Edit account
Actor	System, Administrator
Precondition	• Log in
Main path:	1. Administrator clicks *settings* on website. 2. Administrator clicks *accounts*. 3. Administrator selects an account. 4. Administrator edits name and/or email address. 5. Administrator clicks *confirm*. 6. System verifies name and email address.
Alternative path (A1):	6. System verifies name and email address. a. System remind administrator that name and/or email address is incorrect. b. Administrator corrects name and/or email address accordingly. c. Administrator clicks *confirm* again.

2.1.4.4 Change price

Use Case	Change price
Actor	Administrator
Precondition	• Log in
Main path:	1. Administrator clicks *settings*. 2. Administrator clicks *simulators*. 3. Administrator selects a simulator. 4. Administrator changes price. 5. Administrator clicks *confirm.* 6. System verifies price.
Alternative path (A1):	4. Administrator changes price a. System remind administrator that price is incorrect. b. Administrator corrects price accordingly.
Alternative path (A2):	6. System verifies price. a. System remind administrator that price is incorrect. b. Administrator corrects price accordingly. c. Administrator clicks *confirm* again.

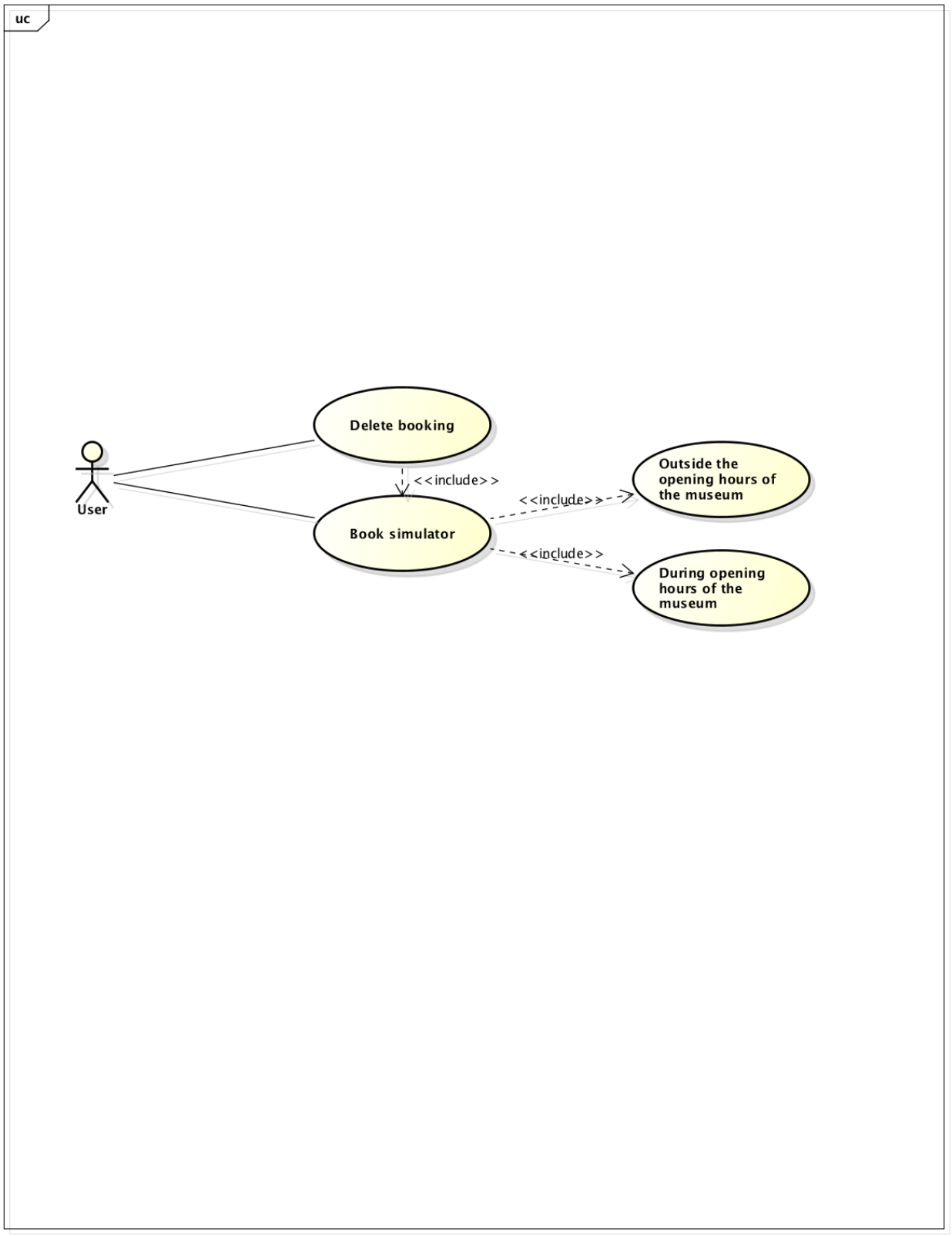
2.1.4.5 Change time

Use Case	Change time
Actor	Administrator
Precondition	• Log in
Main path:	1. Administrator clicks *settings*. 2. Administrator clicks *timeslots* 3. Administrator clicks *add new day* 4. Administrator selects simulator and defines start and end points for time slots. 5. Administrator clicks *confirm.*
Alternative path (A1):	4. Administrator clicks *copy from existing day.* a. Administrator selects existing day. b. Administrator clicks *confirm new day.*
Alternative path (A2):	5. Administrator clicks *repeat weekly.* a. Administrator clicks *confirm.*

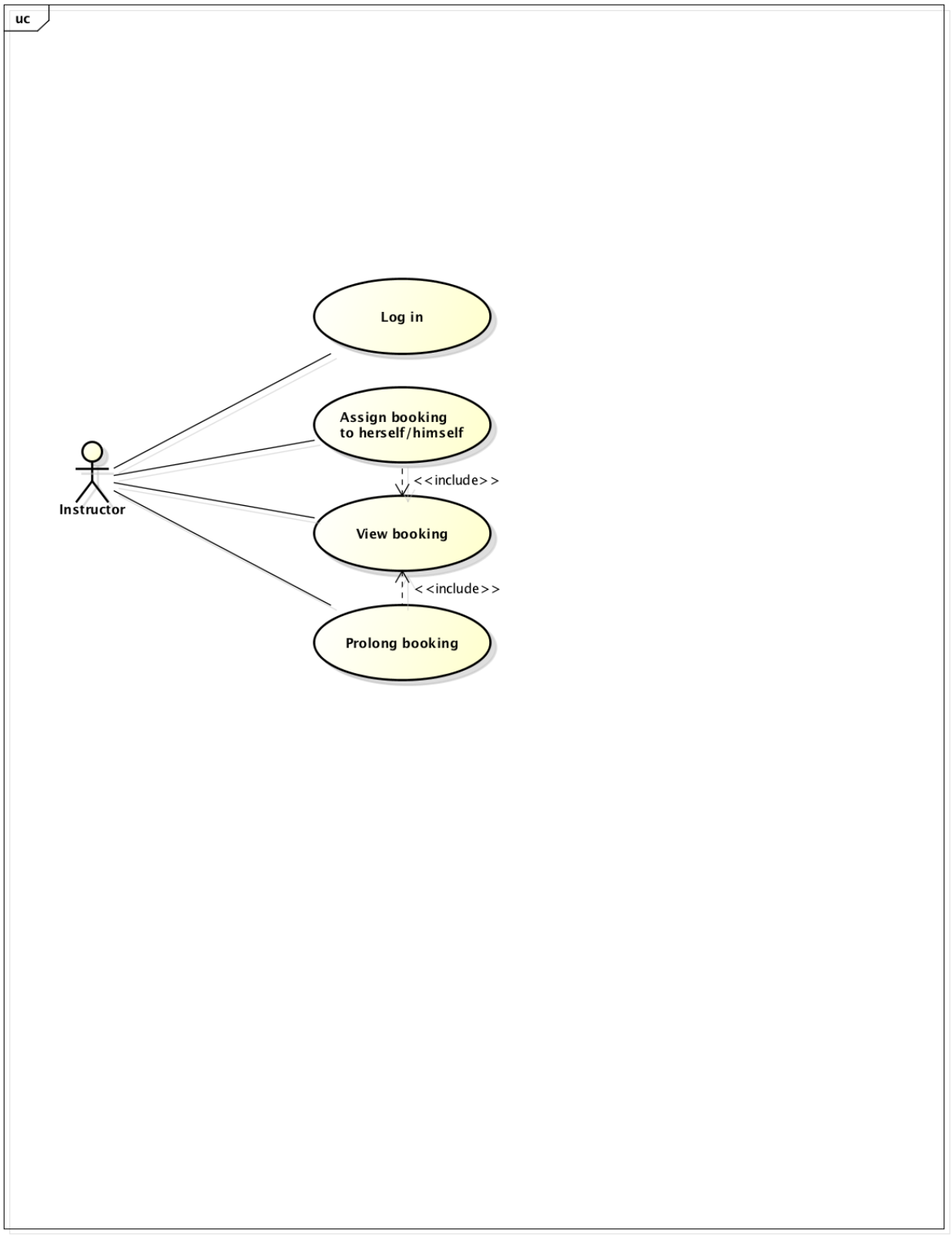
2.2 Use Case Diagrams

The following Use Case Diagrams are based on the Customer Cases in section 2.1. They represent the interaction of each actor from section 1.2 with the system in a simplified graphical representation.

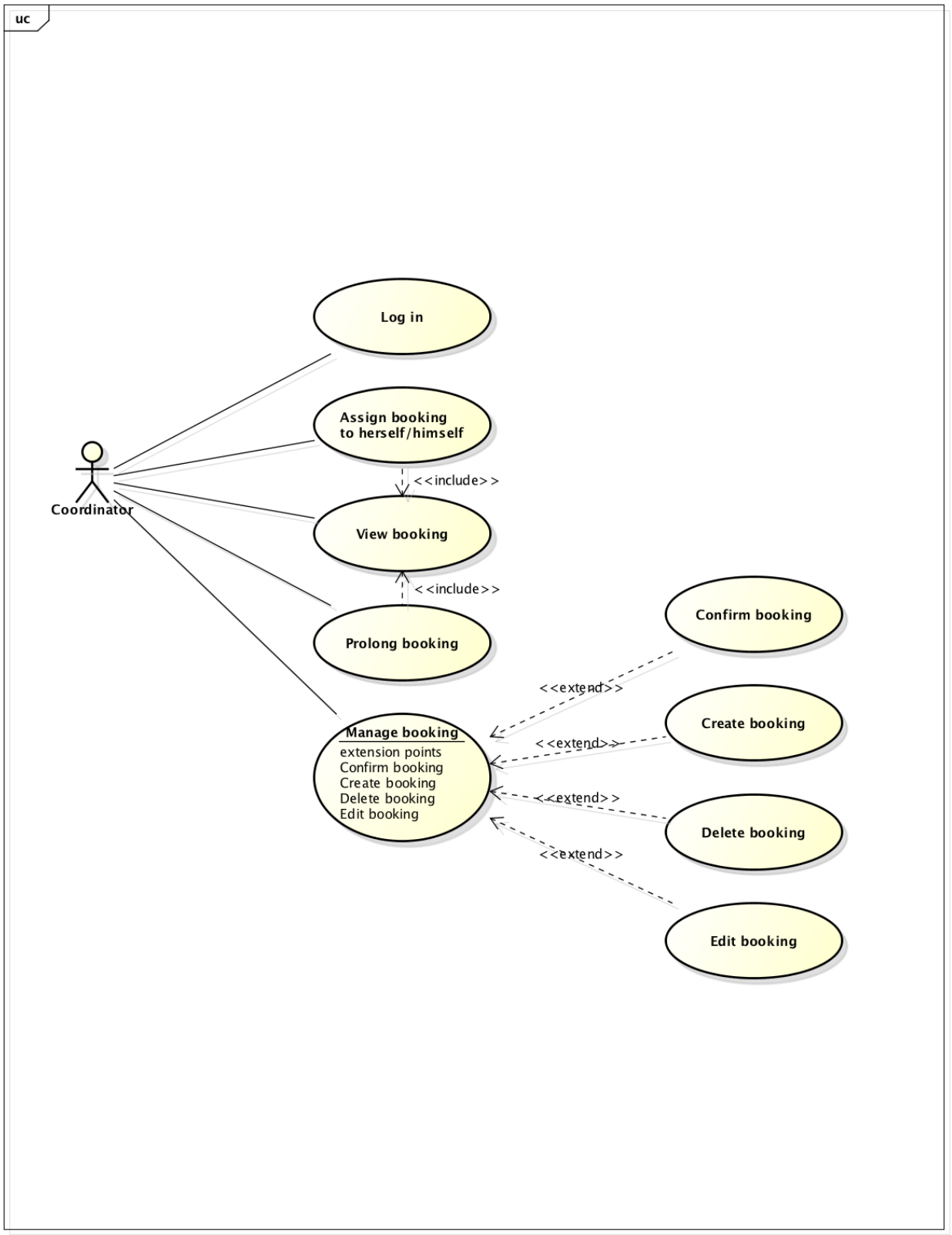
2.2.1 Customer



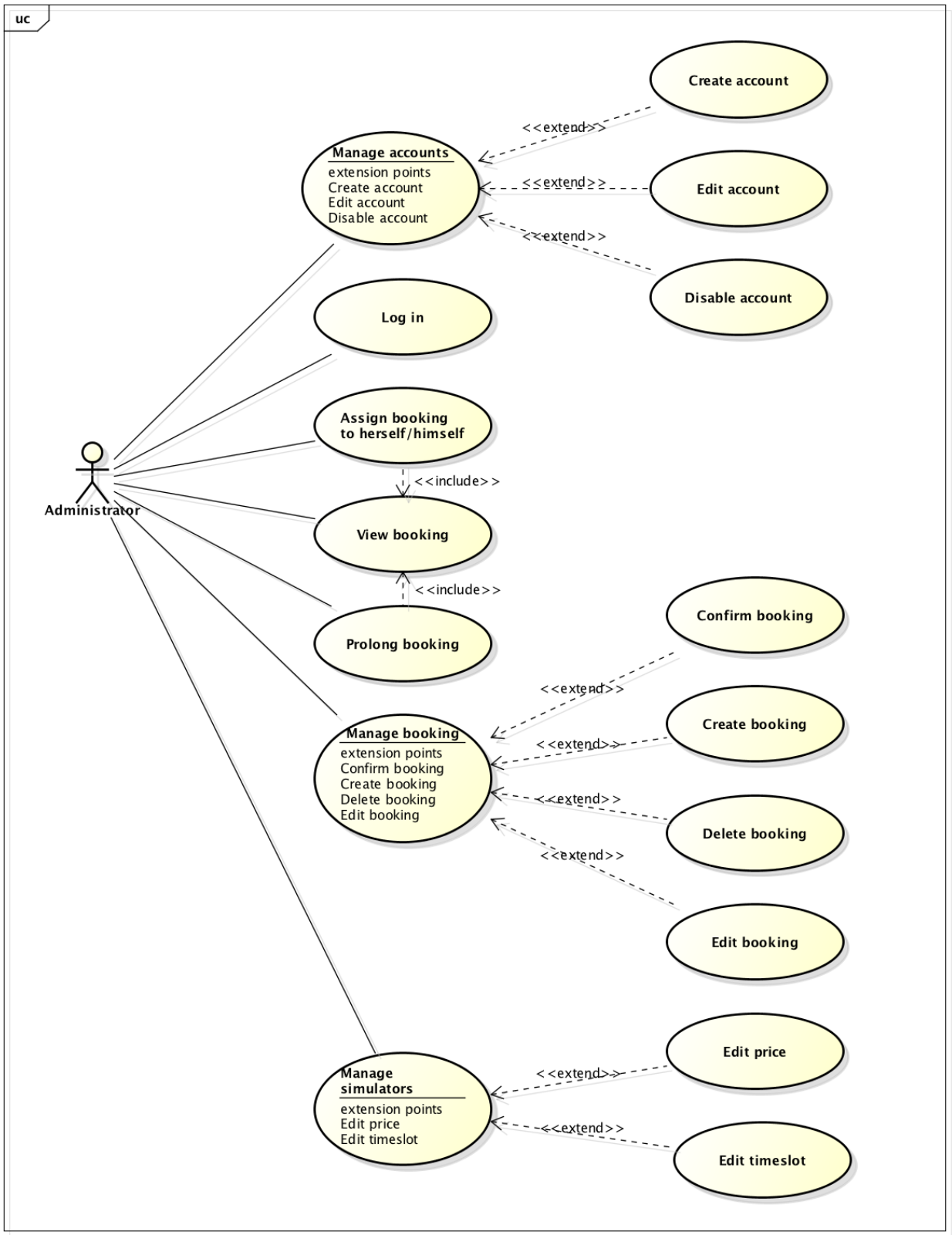
2.2.2 Instructor



2.2.3 Coordinator



2.2.4 Administrator



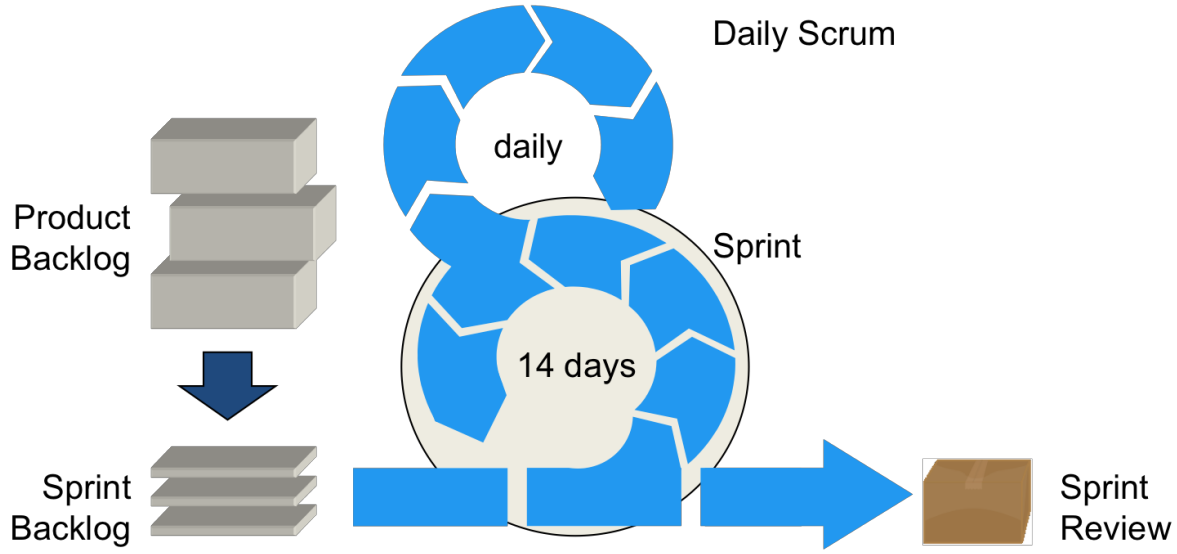
3 Non-functional requirements

Apart from the functional requirements, there are also some non-functional aspects that influence the system's overall architecture and design:

1. The system should provide an easy and simple overview for all users. The navigation structure will be intuitively. Options will only be shown if applicable for the current user. Buttons will be identifiable and visible. There will be no need to guess what a button does, because of it's descriptive name.
2. The system should observe all bookings and update itself immediately.
    - The staff member always needs to be up to date and cannot miss any important new information.
3. The system should be easily usable and accessible. It will be orientated along the following guidelines:
    a. Usability guideline: http://guidelines.usability.gov/
    b. Accessibility guideline: http://www.w3.org/WAI/eval/preliminary.html
    c. Validator: http://validator.w3.org/
    d. Checklist: http://www.usereffect.com/topic/25-point-website-usability-checklist
    - The staff member's age ranges between 15 and 80+ years. This makes it extremely necessary to an intuitive graphical user interface.
4. The system should be maintainable and well documented. This includes usage of PHPDoc, well commented code, self explaining names and modularization of the components.
    - The Västerås Flygmuseum insists to use the system for the next years and want to be able to change it, whenever new requirements come up.
5. The system should not rely on uncertain and/or vague technologies. The used technologies therefore, have to be well documented and supported. Ideally by an active online community (if open source) or supplied with credible update guarantees.
6. The customer interface of the system should be available in English and Swedish.
    - Many of the Västerås Flygmuseum's customers don't speak swedish.
7. The system should be easily usable on mobile devices. There will be no glitches, hard to hit elements (like small buttons) or a more complicated structure. The text will still be well readable and the usability will not suffer from the use of a tablet.
    - The Västerås Flygmuseum will eventually equip all instructors with tablets to ease the current workflow.
8. System will be licensed under the BSD-3 license.

4 Validation of Requirements

In order to stay in contact with the Västerås Flygmuseum, Scrum is going to be used during the development period. Therefore, we are going to have a running systems once every two weeks. This running system is going to be used as a platform to review and validate the listed requirements in section 2. Those functional requirements might be adjusted according to the Västerås Flygmuseum and influence back the current version of the running system.



5 User Stories / Appendix

The following pages contain our in Asana maintained backlog, which contains our user stories.