Requirements engineering

Use case template and interactive prototype

February 5, 2018

Introduction

This report will cover the use case template and interactive prototype for the B.a.a.S. system beginning with the use case template. This use case template builds upon the top-level use case diagram and the low-level use case diagram in the previous report. The provided feedback about that report stated that these use case diagrams were not completely correct. This is why this report provides a different (and improved) low-level use case diagram. The two diagrams are shown below.

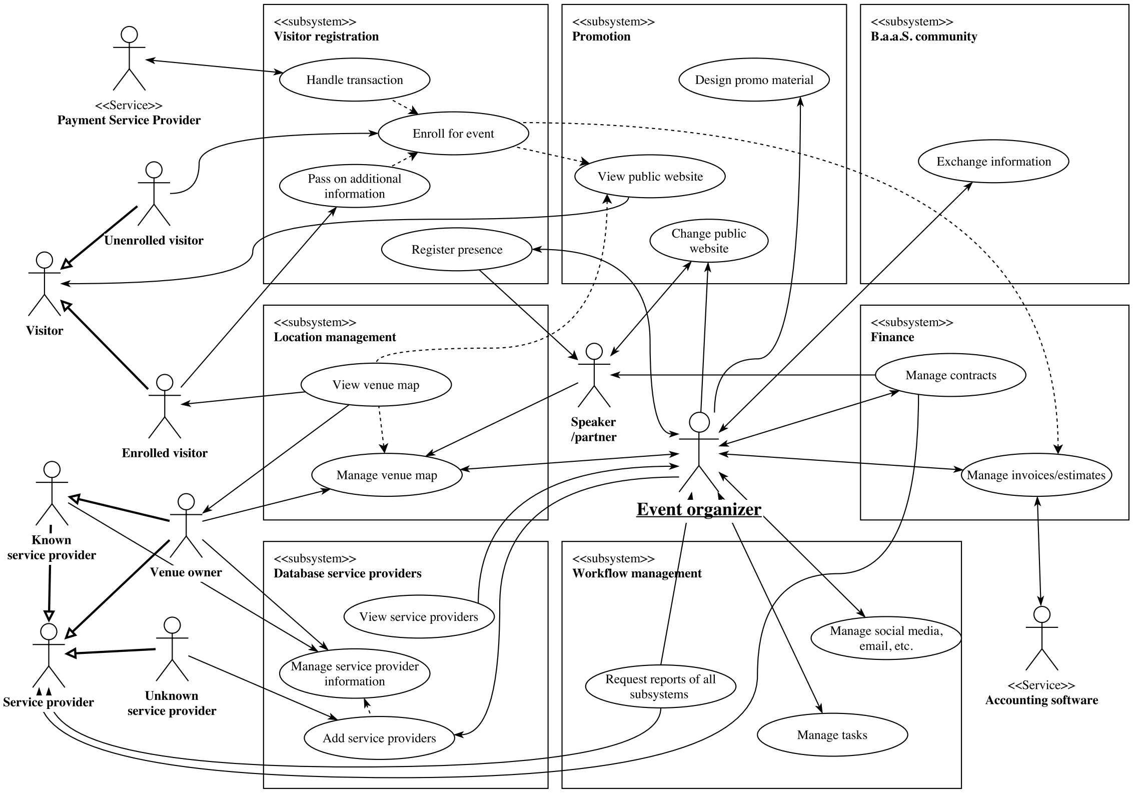


Figure 1: Top-level use case diagram

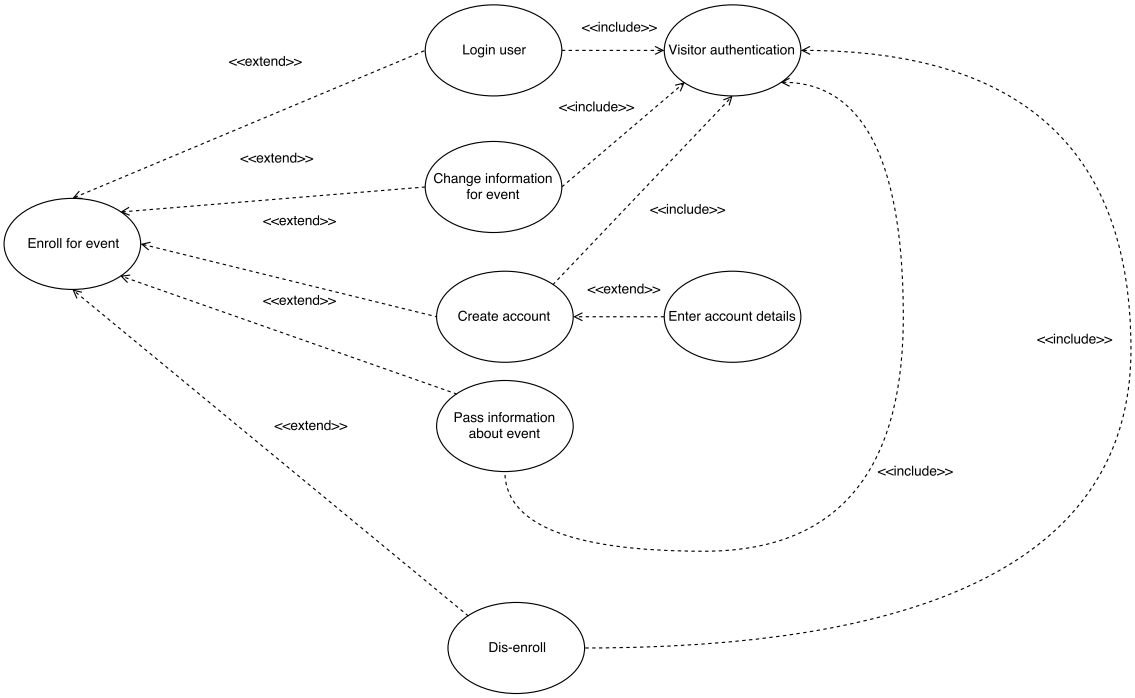


Figure 2: New low-level use case diagram

Use case template

Before creating use case templates, let's first look at the new low-level use case diagram in figure 2. This diagram specifies the actions a user should achieve when enrolling for an event. The user should, before enrolling for an event, always login or create an account. These two options are thus present in the diagram. If the user has logged in, he or she is allowed to enter information about the upcoming event such as what food he or she wants or what talks the user wants to go to. If the user entered wrong information about the event, the user should also be able to change that wrong information. This is also present in the use case diagram. The *visitor authentication* use case represents a system that handles authentication. This system is included in almost all other use cases because the user should be logged in before he or she can edit personal information.

Flows

Now that we have discussed the low-level use case diagram in figure 2 we can create a use case for this diagram.

ID and name	Enroll for event
Primary actor	Visitor of fair
Secondary actors	Authentication system
Description	This use case describes the steps taken by a normal user wanting to enroll for a fair.
Trigger	This use case starts when a visitor wants to enroll for a fair.
Preconditions	The visitor has already registered and owns an account.
Postconditions	The system has registered the user that wants to visit the fair and the user is enrolled for the fair.
Normal flow	1. The user firstly logs in to his or her account with a username and password. 2. The user enters his or her information for the upcoming event. 3. The user enrolls for the event.
Alternative flows	Alternative flow 1 1. The user firstly logs in to his or her account with a username and password. 2. The user enters wrong information for the upcoming event. 3. The user enrolls for the event. 4. The user is confronted with the wrong information and tries to change that information. 5. The user enters new information for the upcoming event. 6. The user re-enrolls for the event with the right information. Alternative flow 2 1. The user firstly logs in to his or her account with a username and password. 2. The user enters wrong information for the upcoming event. 3. The user enrolls for the event. 4. The user dis-enrolls him/herself for this same event. 5. The user enters his or her information for the upcoming event. 6. The user enrolls for the event.
Exceptions	The user may not succeed if he or she forgot the login credentials.
Priority	High
Frequency of use	This case will occur very often because each visitor of a fair will have to enroll him/herself for the fair.
Assumptions	The user succeeded in creating an account.
Other information	Enrolling for an event has to trigger a system so that the user gets a confirmation message/email with all details about the fair and the information the user entered.

This use case describes the steps a user has to take when enrolling for a fair. To make this use case more clear, it will be elaborated further. The use case requires that the user already has registered himself at the website of the fair because creating an account takes a lot of steps and can thus require a new low-level use case diagram. We need a visitor authentication system because enrolling for an event requires personal information, such as a name or age. The normal flow is pretty straight forward. The user logs into the website and enters his or her information before enrolling. The alternative flows are more interesting. The first alternative flow considers a user that enters wrong information. The user thus needs to change that information after already enrolling. This can easily be done by just entering the right information and re-enrolling. The second alternative flow considers this same user that entered wrong information and decides to dis-enroll. After this dis-enrolling he or she re-enrolls with the right information. This use case has a high priority because it will occur every time a user would want to enroll for a fair. The assumption for this use case is that the user already has created an account, this is because the account creation system can be considered a whole different use case.

Throwaway paper prototype

A throwaway paper prototype is included in appendix A. An explanation to the prototype (for testing purposes) is provided in appendix B. The prototype features the use cases present in the low-level use case diagram. Also, the normal flow and alternative flows are fully implemented as can be concluded when testing the prototype. A *Log out* button is provided at almost every screen for the second alternative flow to work. A $Change\ info$ button is provided in screen 5 to support the first alternative flow. The normal flow is also implemented with the *Enroll* button present in screen 3.

Advantages and disadvantages

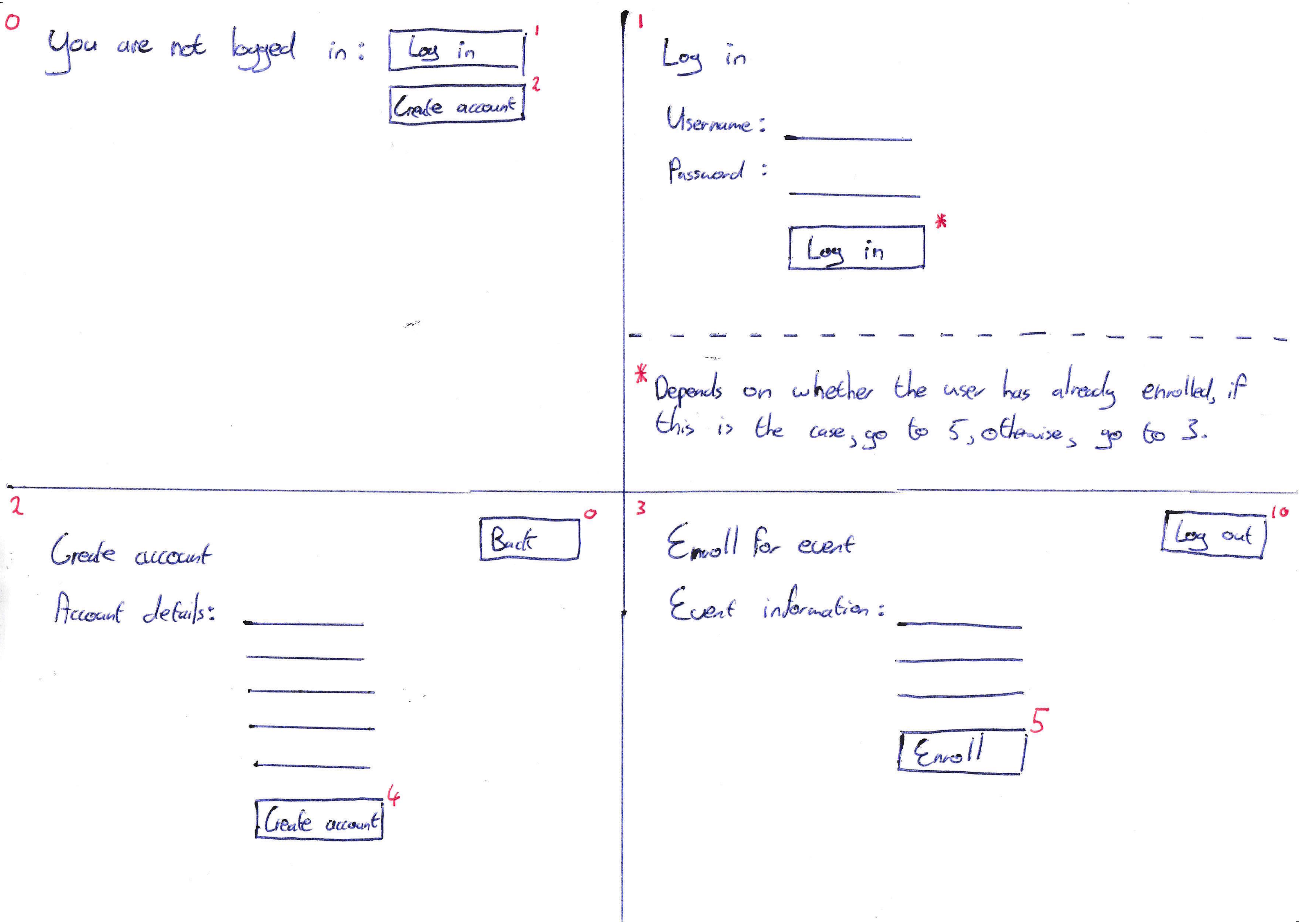
A paper prototype comes with a few advantages and disadvantages. An advantage of paper prototyping is that it is fairly easy to start with. No knowledge about any program is required. Moreover, if you are working together in a team, a paper prototype is a much better choice than an electronic one because working together on paper is much easier than working together on an electronic device. Also, paper prototyping is faster and more inexpensive than creating an electronic prototype. It is ideal for first ideas or quick sketches. However, a paper prototype also has some disadvantages. A first disadvantage is that a paper prototype looks far less professional when showing it to a tester or project owner. A paper prototype is harder to test than an electronic prototype. For example, when using Powerpoint, one could implement links so that clicking on specific buttons takes you to the corresponding next view. This report features a paper prototype because it was easier to implement the low-level use case diagram on paper than on a computer program. A couple of sketches were used before drawing to the final version of the paper prototype.

Design implications

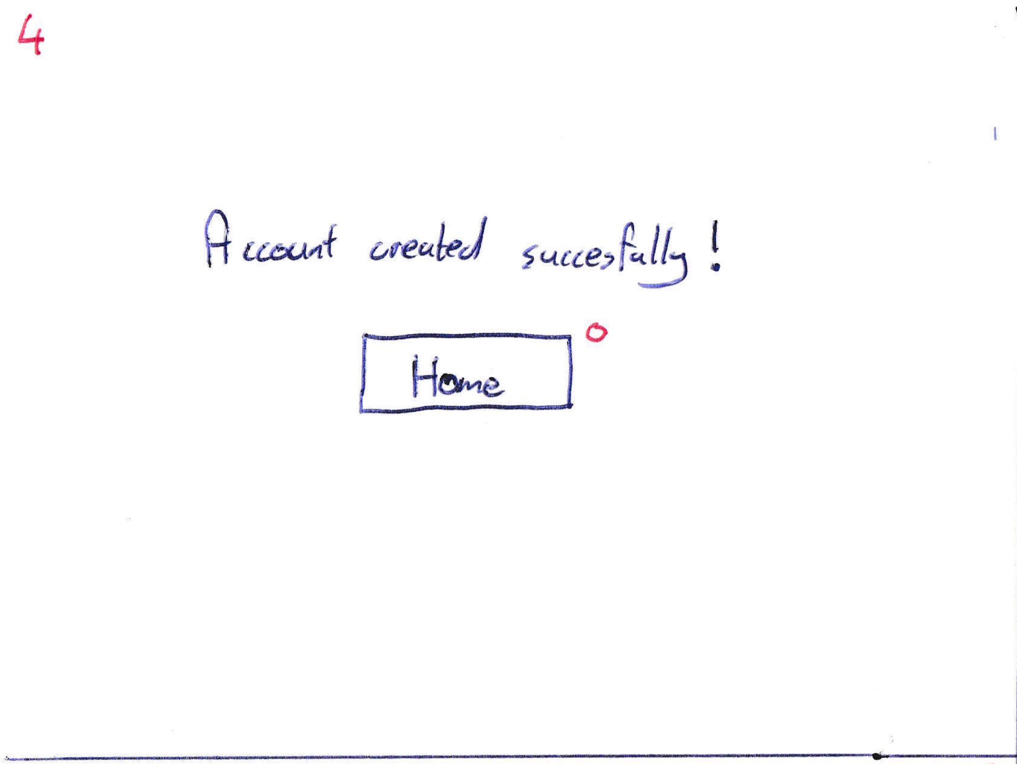
First of, creating an account is present in the low-level use case diagram. This may be a bit weird because in the use case template, we assumed that the user already owns an account. Furthermore an abstract version of the account creation button is also present in the throwaway prototype. The account creation feature is a little abstract in this low-level diagram, that is because the account creation feature can also be expanded into a low-level use case diagram. This report is only about one low-level use case diagram, the account creation feature is taken for granted and the use case abstracts from that. That is why the low-level use case diagram in this report can be considered to be a medium-level use case diagram but this is actually not the case because, except from the user authentication, the diagram can not be expanded further. That is why this report considers the use case diagram to be low-level. Another choice was made with the dis-enroll button. For enrolling to a fair, dis-enrollment is not necessary to implement. One may argue that dis-enrollment is part of another use case, but it is also implemented in this use case to make it more complete. A choice was made designing the low-level use case to also implement it in this diagram because the second alternative flow uses dis-enrolling.

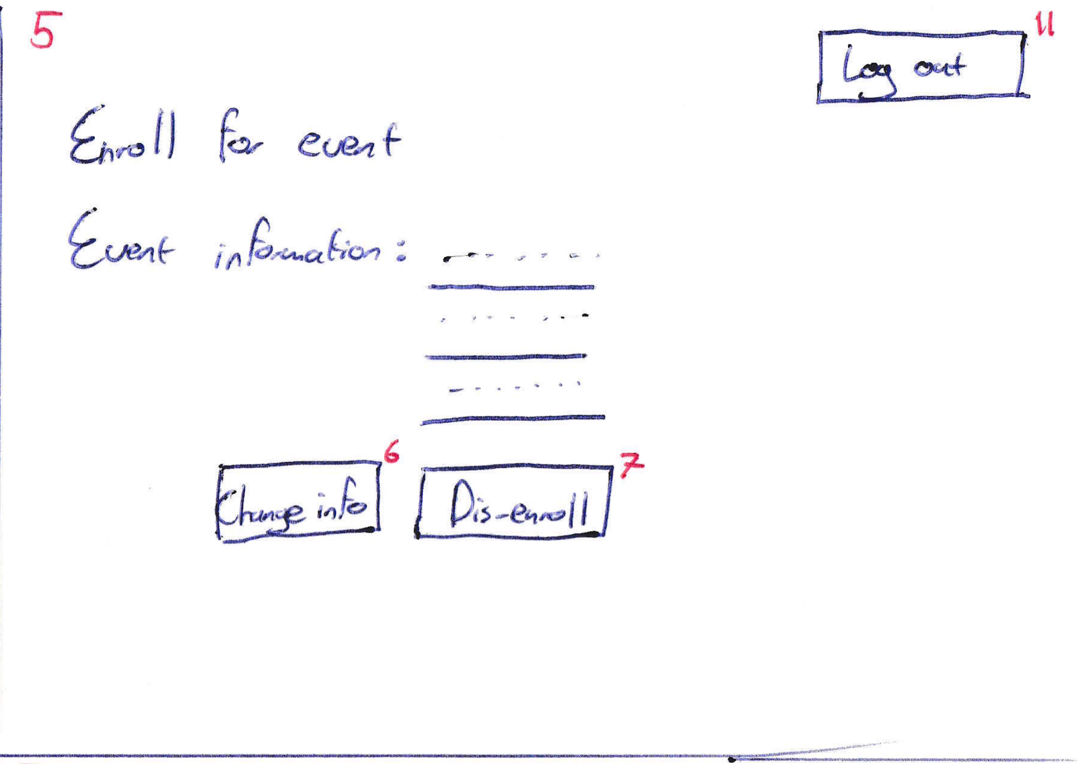
A Throwaway paper prototype

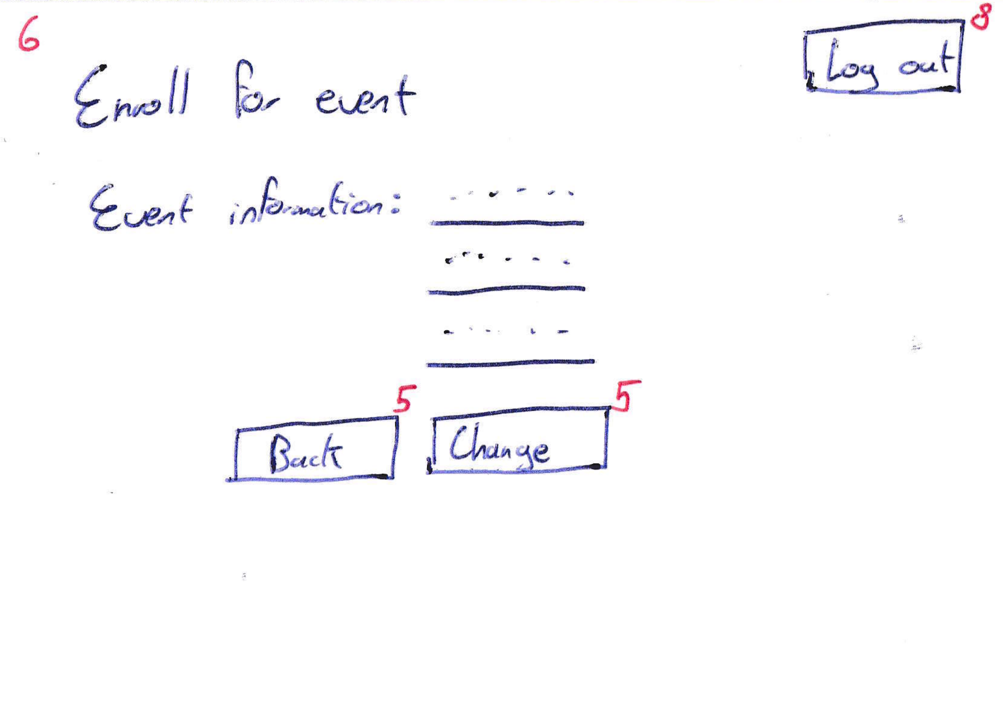
This appendix contains the throwaway paper prototype discussed on page 4.

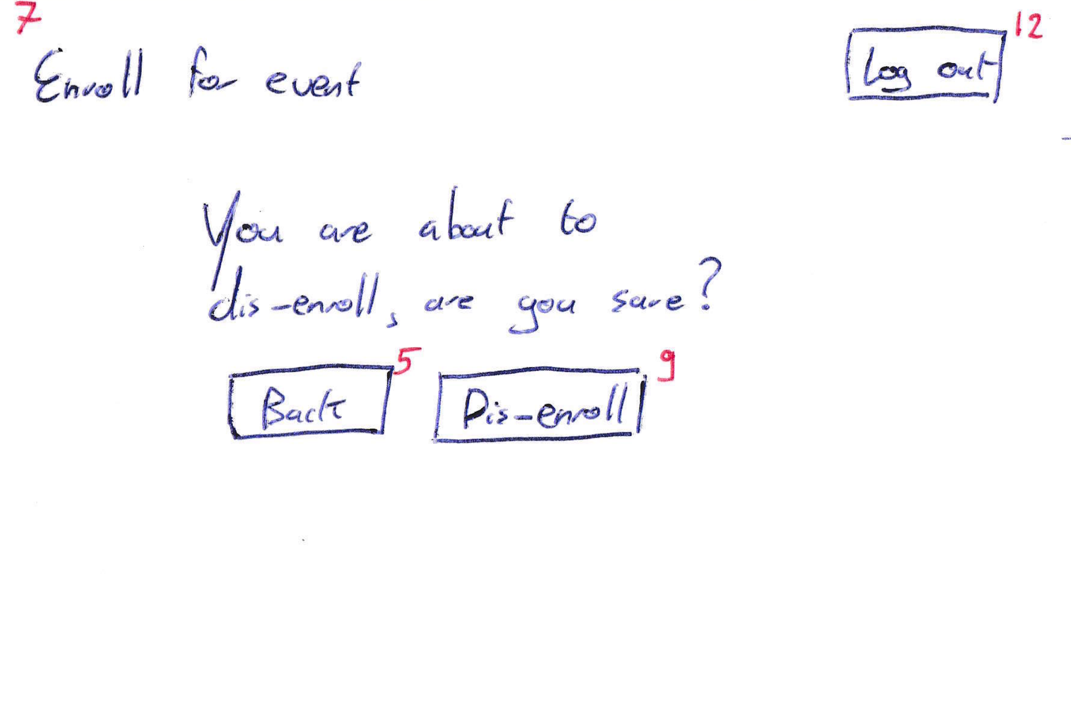


*Depends on whether the user has already enrolled, if this is the case, go to 5, otherwise, go to 3.


4
Account created succesfully!
Home
0


5
Log out
11
Enroll for event
Event information:
Change info
6
Dis-enroll
7


6
Log out
8
Enroll for event
Event information:
Back
5
Change
5


7
Enroll for event
Log out
12
You are about to dis-enroll, are you sure?
Back
5
Dis-enroll
9

8

All changes will be lost.
Are you sure you want
to log out?

Back 6 Log out 0

9

Log out 13

You have been dis-enrolled.

Home 3

10

Are you sure you want to
log out?

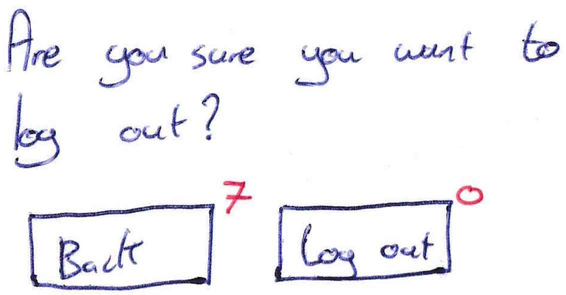
Back 3 Log out 0

11

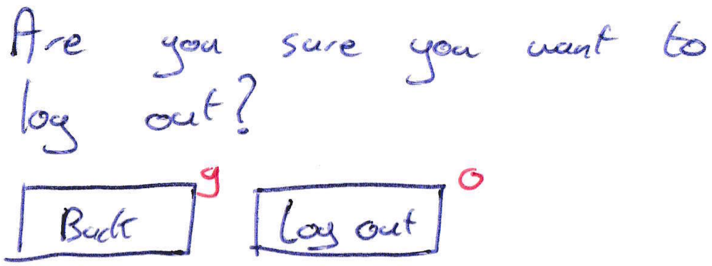
Are you sure you want to
log out?

Back 5 Log out 0

12


Are you sure you want to log out?

13


Are you sure you want to log out?

B Throwaway paper prototype manual

The throwaway paper prototype manual is specified here. The figures used in the prototype are explained first in figure 3.

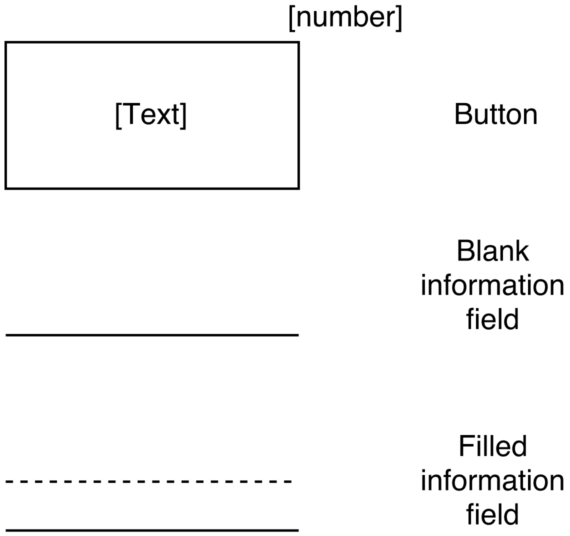


Figure 3: Legend

A button has a corresponding number[1] as seen in figure 3. This number corresponds to the view this button points to. When testing the prototype, a tester must follow the button number to the corresponding view. A blank information field represents an input field for arbitrary information (this could be a name, a password, etcetera). A filled information field is the same as a blank information field except the field itself is already filled with information. All views are represented by a quarter of a page, a view has a corresponding view number in the top-left corner.

[1] This number could also be an asterisk (*), if this is the case, the view will provide enough information to continue the testing.