Web-applications

Project Documentation DIT126

DrinkApp

Robert Stigsson robsti@student.chalmers.se	910814-4719, D
Fredrik Ek ekfr@student.chalmers.se	901109-0959, D
Martin Kastebo kastebo@student.chalmers.se	930309-3372, D
Michael Tran gustranmi@student.gu.se	931016-1576, GU

Group 12

October 24, 2014

Contents

1 The Application

We decided to make a drink application called DrinkApp. The app is intended for amusement and as help for making and managing drinks. It could be used in most areas. It could be used by bartenders as well the average guy/girl with an interest in making drinks.

The application is supposed to store drinks added by different users. Combining this with a rating system will make it possible to have several different options of the same drink, where you from the rating can know which one is probably the best choice. You are also supposed to be able to add drinks to your personal favourites, so that you easily can remember which ones you wanted to try when the weekend comes.

The main attraction should however be the drinksearch functionality where you are supposed to be able to browse for drinks by submitting either name, ingredients or both. This means that you could easily add all the drinkingredients you have at home and the application will show what drinks contain those specific ingredients. It can be anything from fruit to spices to liquor etc.

2 User roles

Our plan was to make it possible for an admin to edit and remove not only his own drinks, but everyone elses as well. We did however only prepare this functionality rather than finnishing it due to lack of time and other priorities.

3 Use Cases

3.1 Main Page

- Search for drinks by name
- Search for drinks by ingredients
- Search for drinks by name and ingredients
- Register account

3.2 Authenticated user

- Log In
- Log Out
- Change password/email under profile
- Add Drinks to the database with name, ingredients, types, steps och comment
- Watch your drinks under My Drinks
- Enter a Drinks page to share it with your friends on facebook using the url (Dont need to be logged in to see a drinks page)

- Edit your own Drinks under My Drinks
- Delete your own Drinks under My drink
- Add drinks to My Favourite Drinks by pressing the heart button next to the drink name while browsing for drinks
- Watch your favourite drinks under My Favourite Drinks
- Unfavourite your added drinks
- Rate Drinks

4 Selected approach

We selected a component- based approach using the JSF framework.

5 Tiers

Our application concists of several tiers or layers From top to bottom we have:

5.1 Web Pages (xhtml)

All web pages. The front-end of our application. The only parts that a user will interact with.

5.2 BackingBeans (Java)

All Beans that handels the functionality of the different parts of the application. Works like a minor back-end for each web page, managing all temporary data.

5.3 Controllers (Java)

All Controllers that handels communication between the application front-end and the database.

5.4 ORMs (Java)

The Object Relational Mapper where we all models are implemented in Java and translated to the underlying tier in SQL.

5.5 Database (SQL)

The underlying database where are data from the application are being stored.

5.6 Other

Aside from these we are using css and javascript for the layout on the web pages and BCrypt as encryption algorithm for passwords.

6 Code Structure

We divided our different code parts into sections looking a bit like the tiers, but with some differences.

6.1 nu.drinkapp.auth

Package containing classes involved in the authentication process.

6.2 nu.drinkapp.bb

Package containing all backing beans.

6.3 nu.drinkapp.core

Package containing all the core classes of the application such as the Models and their relations and helpclasses.

6.4 nu.drinkapp.ctrl

Package containing all controllers.

6.5 nu.drinkapp.persistence

Package containing all classes involved for persistence in the application.

6.6 nu.drinkapp.wrappers

Package containing all wrapper classes and interfaces.

7 Other

There are alot of things we wanted to do from the start that we simply didnt have time to finnish, and there are some things we tried to do but didnt succeed with. Here follows some:

7.1 Functionality we tried to do without succeeding

7.1.1 Security

We wanted to make the application protected against SQL,javascript and HTML-injections, something that we learned would be established by using only named-queries or prepared statements. This succeeded for almost all queries in the project, not for all however. We tried to ask for help during the supervision hours without success. It might be because of the use of subqueries in one of the queries or something like that. Neither did we manage to get cascade to work for entities, meaning that we had to make a native delete query that first deletes from all entities, using the one we want to delete from, as a foreign key.

This might not be a course in security, but we wanted to say that we did think of it. As an addition, because of this, we made a function to sanitize all database inputs

from the use of special characters. There are too good ways of performing injections nowadays though, so it wouldnt help versus a good hacker.

7.1.2 Tests

We tested our application on the go as soon as we implemented new functionality. But we also intended to make tests that could be run to easily verify that everything are working as intended. This is however something we spent countless hours trying to get to work. We tried to ask several supervisors for help without getting a proper answer. We simply dont have any more time to put into it, being the major reason to why our project doesnt contain any unit tests etc. Writing the actualy tests would would take us no time at all, but seeing as we couldnt run them it seems rather pointless at this stage.

7.2 Ajax problems

Something else we couldnt get to work and that the supervisors couldnt help us with was when using JSFs built-in ajax functionality the last letter in a textfield wont update with ajax. This can be seen when using find drinks. For instance when typing something in the search field and then erasing it, it wont update in the target. We do not know why and we couldnt get help fixing it, but we are aware of it.

7.3 Functionality we wanted to do, but didnt have time for

Here follows some examples of functionality we didnt have time to implement:

- Complete an admin interface so that it would be possible for admins to edit and remove all drinks in the database no matter which user added them.
- Add a table under myprofile with some fun facts such as how many drinks the actual user added, how the actual users drinks are rated, how many drinks the actual user have rated etc
- Add a guestbook where users could write and suggest updated functionality etc
- Live-feed on the home-page, that gets updated automatically when new drinks are added to the database.
- Some kind of chat, so that users can chat with each other.
- Facebook connectivity. So people could sign up on the application using their facebook id rather than registering on the page.

We also discussed alot of other things. Such as adding small infobuttons on all views containing forms etc.

We are however very happy with the application as a whole. We did spend alot of time on it and belive that the result is really good. The application can be found at drinkapp.nu:8080/DrinkApp

8 Appendix

8.1 UML

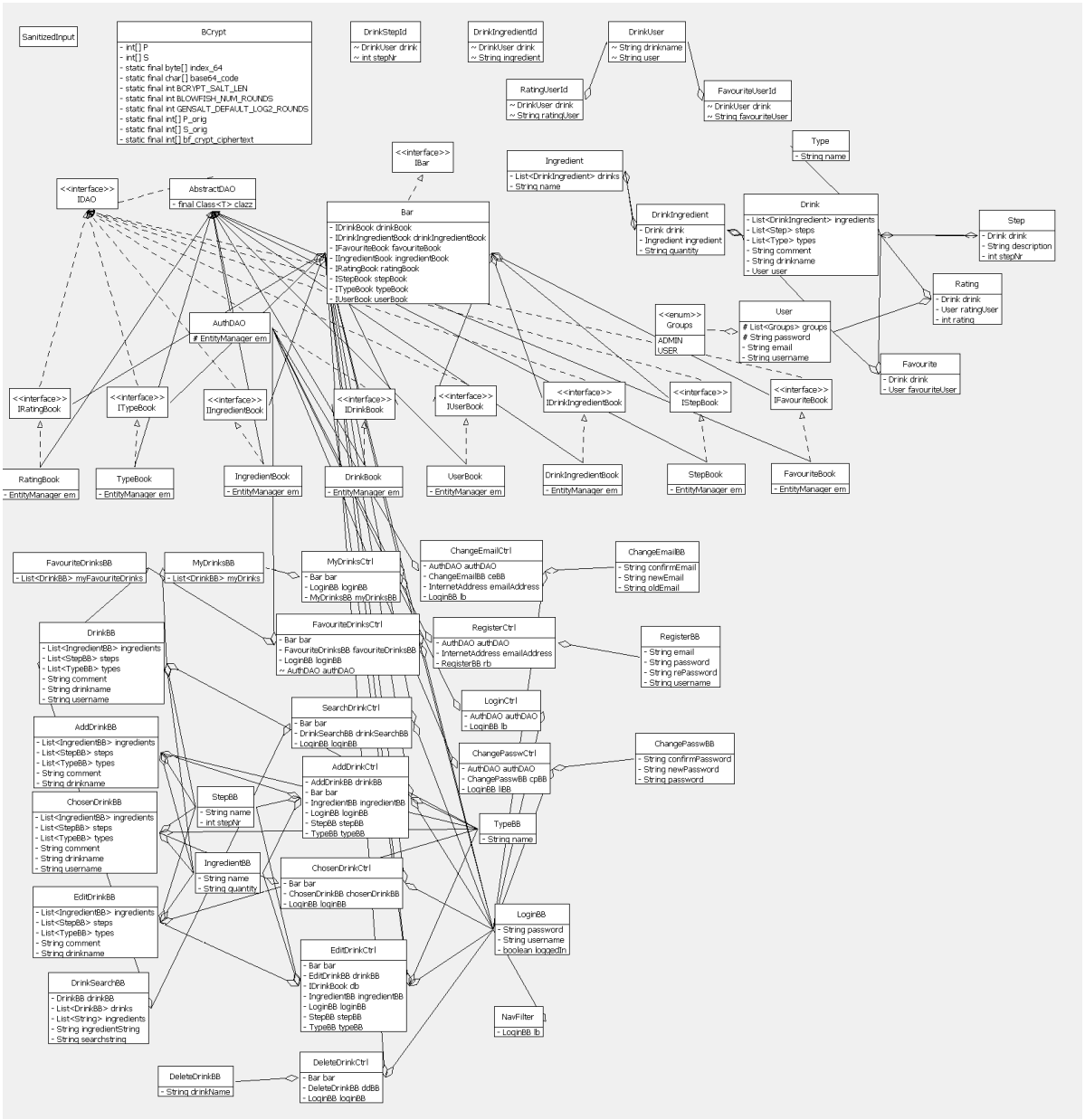

Figure 1: A nice UML-diagram of the application