CHALMERS
UNIVERSITY OF TECHNOLOGY

Forum

David Hermansson
Johan Blom
Jonathan Edfeldt
Jonathan Uhre
Jonathan Örgård
Grupp 6 - Scrubs
DAT076 Webbapplikationer
VT 2020

Introduction

In this project a website has been created for a discussion forum with the use of the Java Server Faces framework. On the forum a user can create a user account and then login to access more functions, given that the user provides the correct login information. When logged in the user can create comments and threads. The comments are located in threads which in turn belongs to a specific category, a category can only be created by an admin. Admins can also ban normal users and remove/edit content. The website stores the user credentials in a database, the password is encrypted to provide more security.

Github Repository: https://github.com/Orgardj/DAT076_Web-applikationer.git

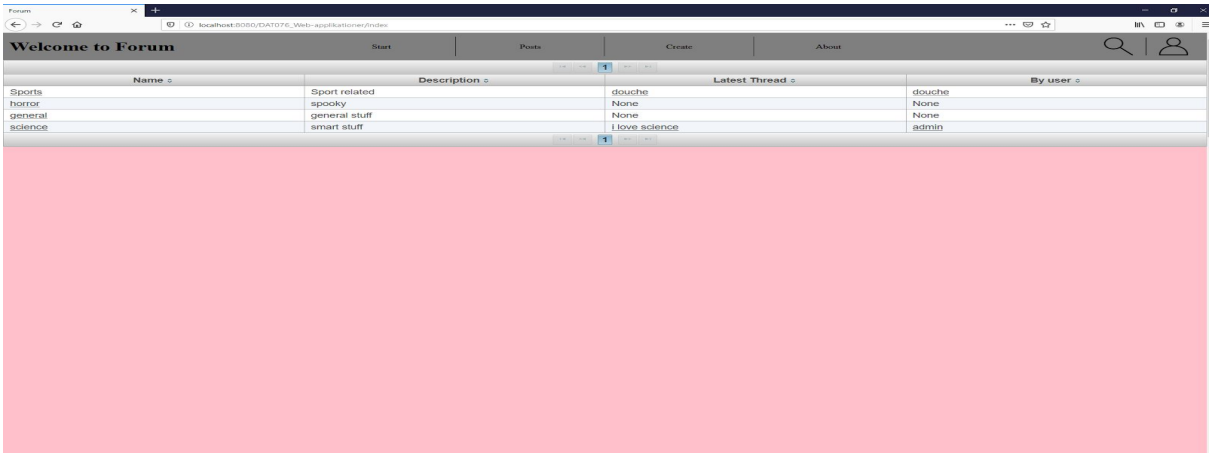
Private repository, contact orgardj@student.chalmers.se for access, Adam should have it already.

Use cases

- Admin create category
- Create thread
- Create post
- Edit own post, admin can edit any post
- Admin remove post
- Admin ban user
- Create account
- Login to account
- Remember user between sessions
- Change account settings
- Visit profile page to display user info and latest posts

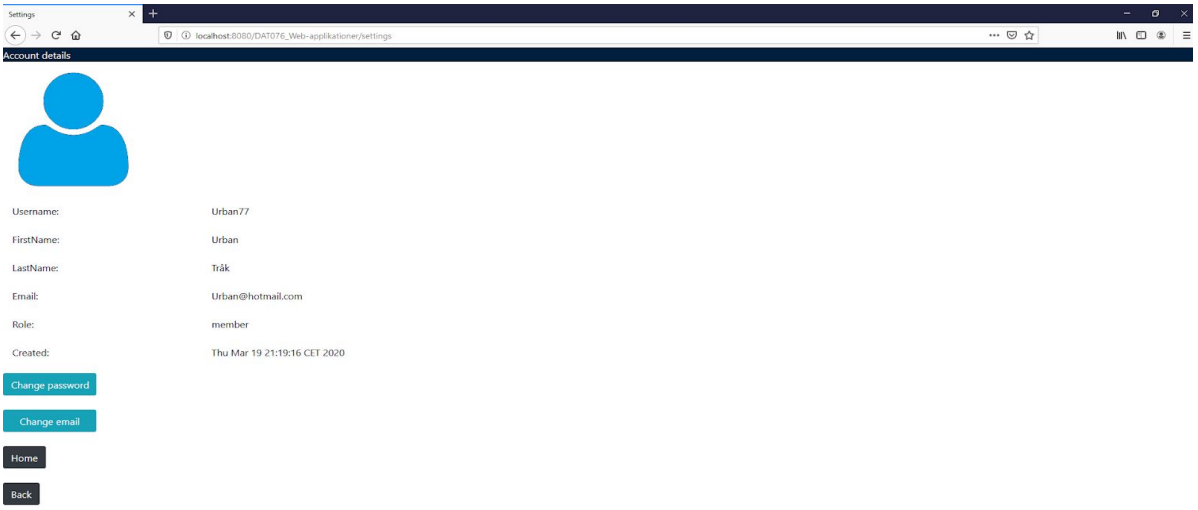
User manual

As a user when you first open the site and you don't have an account, you have the option to view all the categories,threads and posts, but you can't create anything. On the first page the categories are displayed, to view the threads belonging to that category the user can click on the category and will then go to another page where the corresponding threads are shown. To view a thread with its comments you once again just have to click on the thread and the comments will show. To be able to create comments and threads you have to login, if you don't have an account you have to create an account by registering. To create an account you click the top right button on the starting page which will show a window where you can login and register. To successfully create an account all input fields have to be entered, in which their respective requirements have to be valid, the username and email have to be unique. When the account is created the user can then log in and when logging in; the user has the option to remember the account so that the user is still logged in even when closing the page. If the user is an admin he can also create new categories on the starting page.

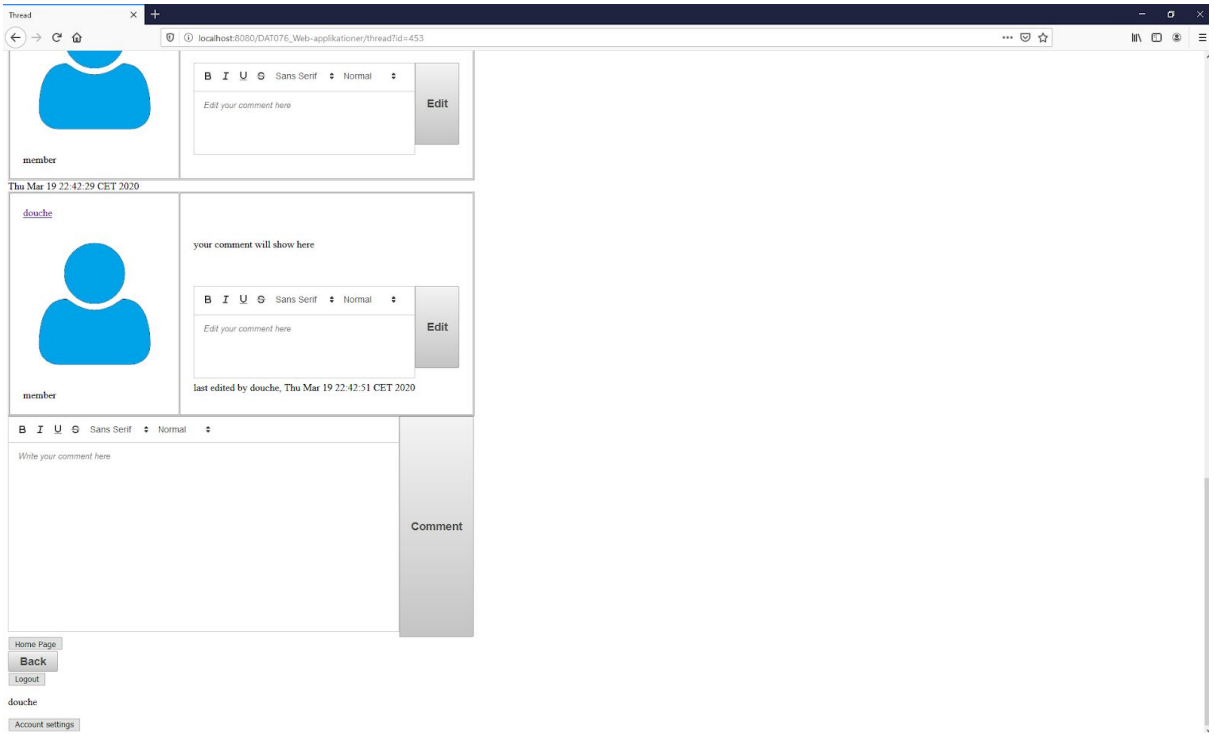


(Starting page with all the categories, option to login and register in top right)

If the user is logged in, he can change his password and email and also view the account information by clicking the “account settings” button which you only can see if you are logged in. To change email or password, just click the respective button and the user can then input their updated information.



(Account settings page)



(Thread page with all the corresponding comments)

In the thread page the user can see all the comments belonging to the thread and if logged in, have the option to write their own comments and edit them. An admin can also edit other users posts and remove them, the admin can even ban a user if he desires.

Design

Frameworks

- Java Server Faces
- Arquillian
- Junit

Libraries

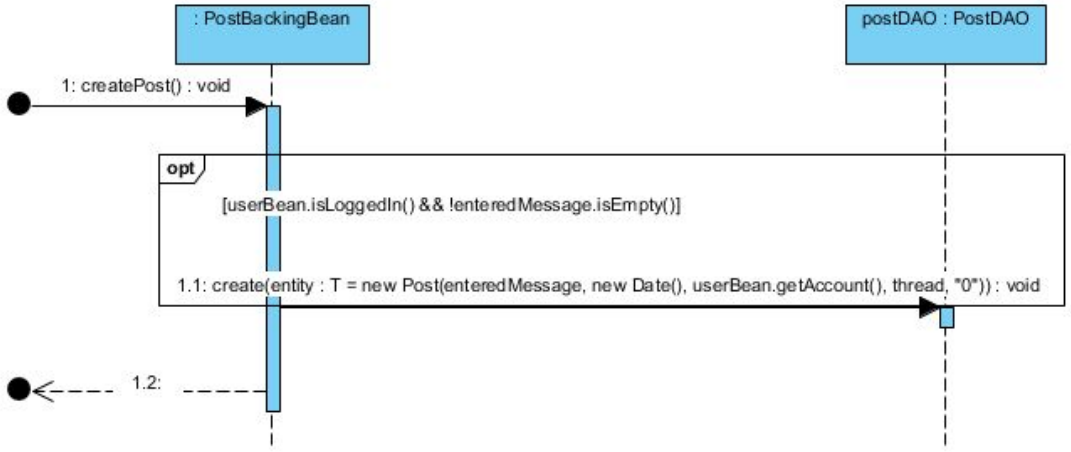
- PrimeFaces
- OmniFaces
- Bootstrap
- Apache commons lang
- Lombok
- Querydsl
- jQeury
-

Other Technologies

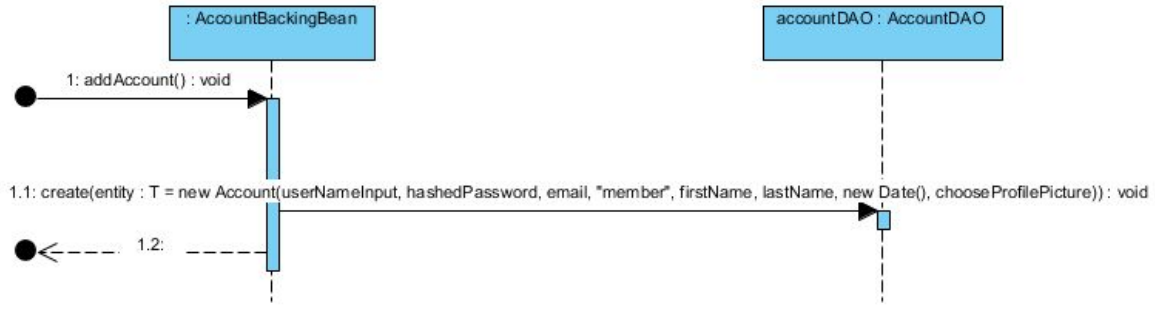
- Payara
- Java EE
- Javax
- Maven
- Java DB

Application Flow

To create a post, the user presses a single button along with some inputted text. the button calls newPost() in postController, which in turn calls createPost() in a backingBean. The backingBean performs some checking and creates a new Post entity by invoking the data access object for Post.



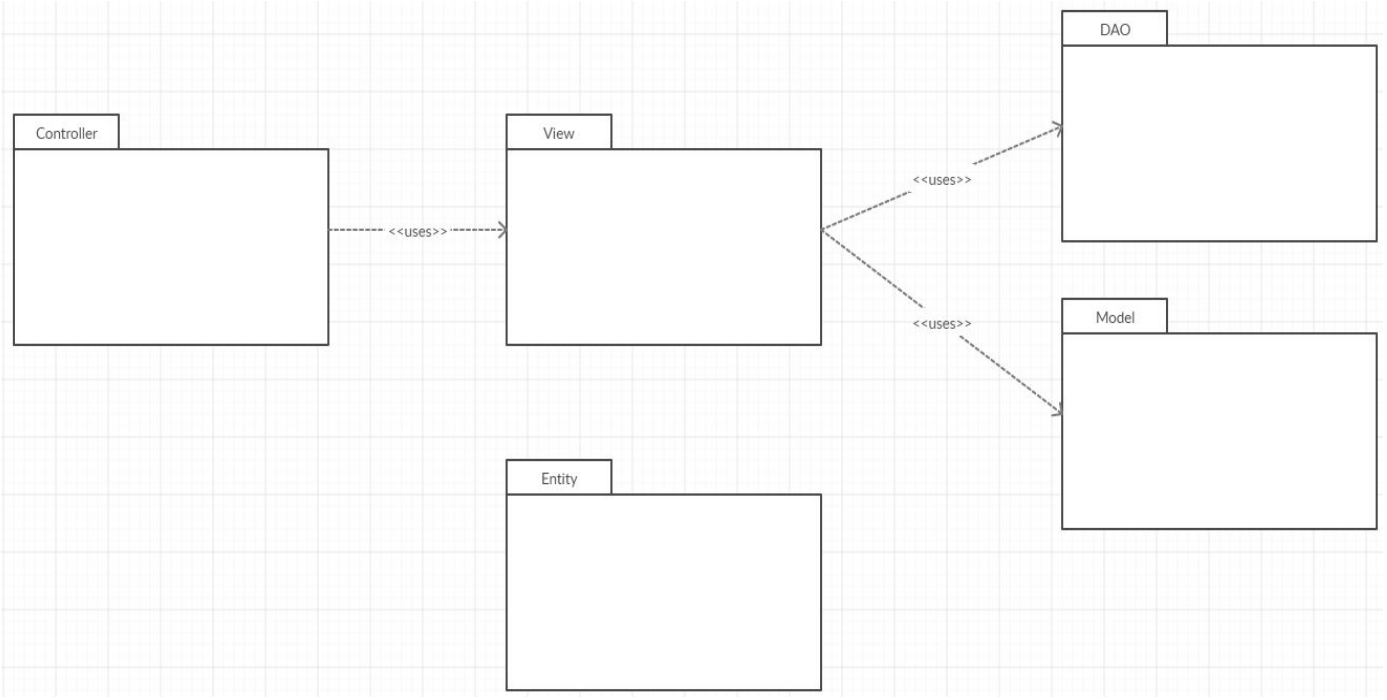
To create an account, the user must fill out a form and press a button. The button calls addAccount() in a backingBean which creates a new Account entity by invoking a create method in its corresponding data access object. The checks for correct input are handled by jsf, and are therefore taken care of before backingBeans get to do anything.



Most commandButtons in our jsf files invokes a method in one of the controllers, which in turn calls a method in a relevant backingBean. After the backingBean has updated the backend accordingly, the

controller can update the active view to present whatever changes has been made. These steps are what the majority of our use cases boil down to, no matter how you interpret them.

Package diagram

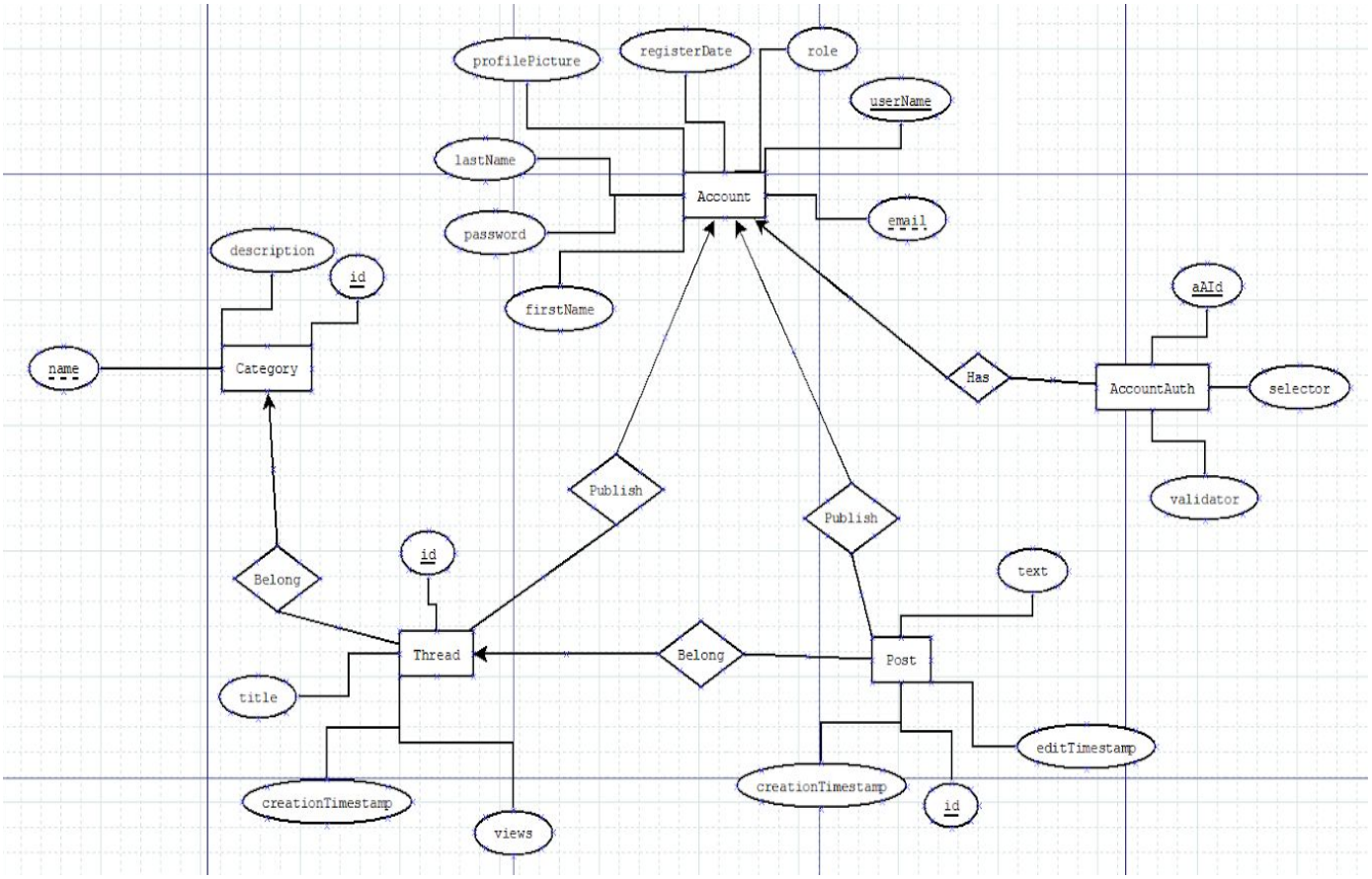


Code coverage report

lab3-1.0-SNAPSHOT

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.view		0%		0%	343	343	220	220	143	143	5	5
model.entity		24%		8%	245	282	44	125	79	116	0	10
controller		0%		0%	20	20	44	44	15	15	5	5
model.dao		73%		33%	7	35	20	97	4	32	0	6
model		0%		0%	6	6	6	6	5	5	1	1
Total	4,775 of 5,631	15%	721 of 750	3%	621	686	334	492	246	311	11	27

Model diagram



Responsibilities

We used a Trello board to keep track of what everyone was doing. Trello link: https://trello.com/invite/b/GVBg75BU/4418507a800ad4c06ea5088755438d64/dat076web-applikationer

Responsibilities by member:

- Jonathan Edfeldt
    - Frontend for the index page
    - Refined login/register component
    - Frontend, assembling the navigation between components and pages
- David Hermansson
    - User profile picture
    - Edit posts
    - Some more DAO backend tests
    - Back button
    - Search for post/thread

- Jonathan Uhre
    - Prime faces text editor
    - Create new thread
    - Edit posts toggle visibility and default value
    - Multiple minor fixes
- Johan Blom
    - Account settings page
    - Create account
    - Encryption for passwords
    - Login page and login logic
    - Register page and register logic
- Jonathan Örgård
    - Latests thread/post
    - Remember user between sessions
    - Admin ban non admins
    - Admin remove category/thread/post
    - View counter for threads
    - Create category
    - Keep user logged in for the session
    - Public account page
    - Refresh components with websockets
- Everyone (lab 4)
    - Most of the DAO backend tests
    - ER diagram for the database
    - Entities for each database table
    - DAOs for each entity
    - Frontend views to display categories/threads/posts