Receptsidan - Group 15

Adis Mahmutovic - 951127-0010
David Berg Marklund - 920807-1671
Linus Nilsson - 940413-6294
Max Hansson - 930505-9199

March 2018

1 Links

Live website - http://www.dcc.nu/
GitHub repo - http://github.com/maxh-/DAT076-project/

2 Application Flow

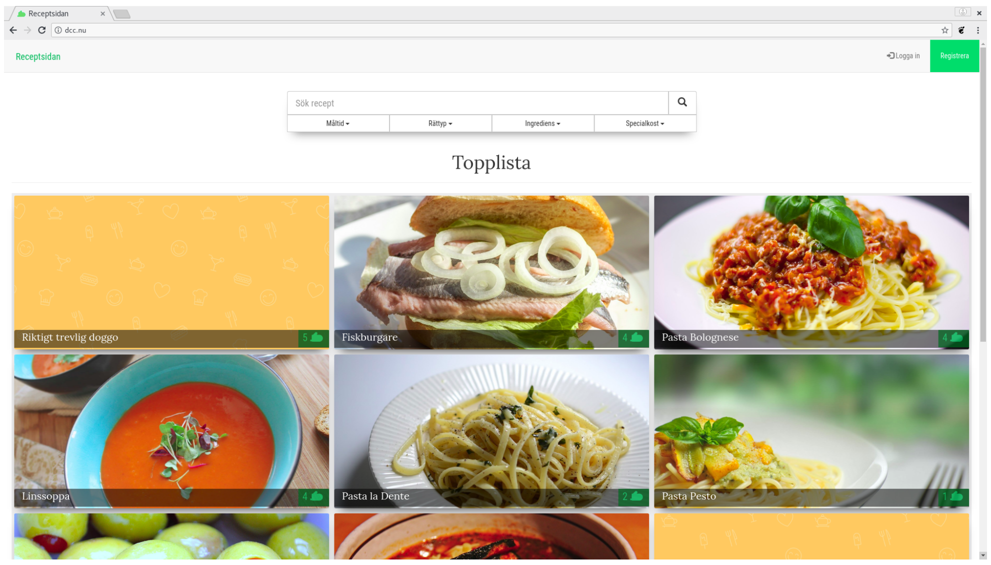

Figure 1: Screenshot of the homepage

Once the user visits the page, the user is faced by recipes based on popularity. If nothing appeals the user, he or she can filter recipes by either selecting one or more tags or actually searching by text. If the user finds a search filter that good, he or she can then share the URL to one or more of his or her friends.

Once the user finds a recipe that he or she wants to cook, the user can click and visit the recipe to get a better look at it and its instructions and ingredients. If the user then proceeds to the cooking stage, he or she can then enter cooking mode to get an easier look at the instructions and simply navigating between them.

If the user finds the recipe well tasting, he or she can then add it saved recipes or simply press the "ovenmitt-like"-button. If the user decides to both like and save it, the recipe basically gets two likes. The recipe then gets bumped up for other users looking for popular recipes in the browse page.

If the user saves the recipe, it gets saved and can later be found among saved recipes.

Also, if the user likes the recipe, he or she can visit the recipe writer's public profile page to maybe get a glimpse of other recipes the writer has published and possibly recipes for another day.

If the user tries to like or save a recipe while not logged in, he or she gets ridirected to the login page.

If the user creates a recipe he or she can't wait to share with other people, the user can then publish it by adding a new recipe to the page.

If that the user made a mistake, such as forgetting an ingredient or misspelling, he or she can then edit the recipe from the private profile page.

If the user has been logged out for a while and forgot his or her password, the user can click the "Glömt lösenord"-link in the login form to retrieve a reset password link where the user can select a new password.

3 Use cases

- Authentication
- Add new recipes
- Private profile page
- Modify existing recipes
- Rate recipes/comment on recipes
- Browse page with suggestions
- Filtered search
- Cooking mode

- Public profile page
- Favorites

4 Application design and used techniques

The app is built in Node.js.

4.1 Front-end

The UI, aka the "view" part of the application is written using React components with Twitter Bootstrap UI framework. The UI state is managed in the individual components, as well as Mobx state manager stores where more complex state management is needed. (i.e. for handling full recipes.) The UI is entirely decoupled from the backend and communicates through a REST-style API. For authentication we used the JSON Web Token standard [link]. For the comments system we used Disqus service.

Bookmarkability is ensured by coupling the browser URL with the application state wherever applicable, for example in the **start page** where you can link to the page with filters pre-applied.

Libraries used:

- React
- Mobx
- Bootstrap
- Disqus
- jsonwebtoken

4.2 Back-end

The backend is a RESTish API where each endpoint returns a JSON. The backend is of a model-controller design. The model is defined by the ORM Sequelize which hooks up to a SQL database, in our case a Mysql database. This can be seen in servers/models. The ER diagram for the model is described in figure 2. We use Node.js with express do define the routes and endpoints. Express is defined in bin/www and server.js in the root folder. In server/routes all the routes are defined. The authentication library used is called Passport and it is defined in server/config/passport/, we chose to use a token based authentication approach since it works well with REST APIs and React. Each route that needs validation has a validation schema defined in server/validation-schemas, the validation library used is express-validator. Middlewares are defined in server/middlewares where you can find authentication middleware. Controllers are in server/controllers and does what you would expect a controller to do.

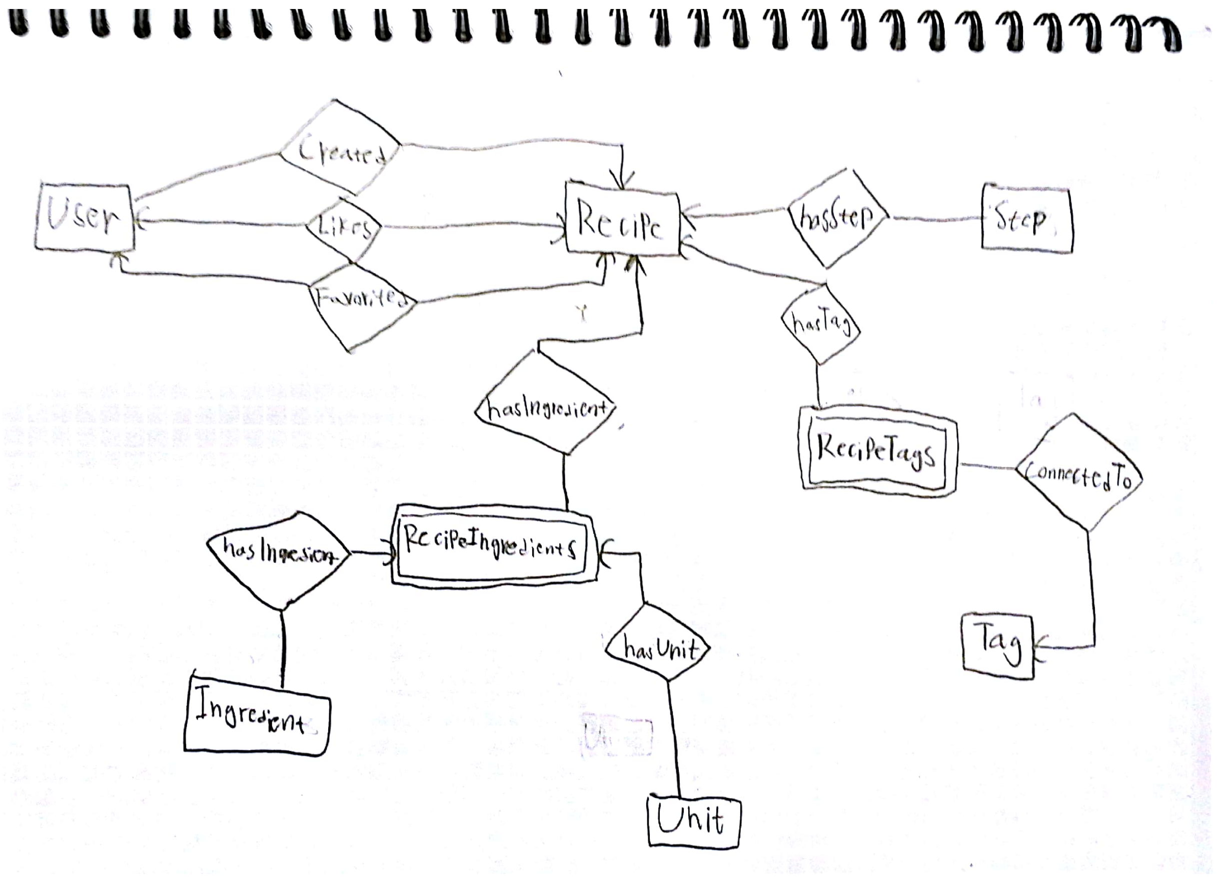

Figure 2: ER-chart of database

Main libraries used:

- Express
- Sequelize
- Express-validator
- Passport
- jwt-simple
- bcrypt-nodejs

5 Responsibilities

Max Hansson had programming lead responsibilities. He was also frontend lead developer. David Berg marklund was backend lead developer. Linus Nilsson was project manager as well as frontend developer and Adis Mahmutovic was also front end developer.

We all decided to use SCRUM as a workflow.