Applied object-oriented programming (EEN060/EEN065)

Tillämpad objektorienterad programmering

Final Project

Project group

Group number: 1

Linnéa Solbakken Sjöberg & Anya Toftefors (Program: TIDSL)

Project idea

Project title

Recipe sharing system

Project description

The idea of the project is to create a recipe page where users can share and review each other's recipes. You don't have to be logged in to use the page but if you want to make a recipe post or comment other people's recipe you have to register an user account and be logged in to it.

Use case 1: List (mandatory)

Title:	Home page

Description:	A page where the 10 last recipe posts are listed. For each post there is the name of the author, the publication date, the title of the recipe and a grade from visitors that have tried the recipe.
Condition:	User does not need to be logged in to access this use case
Screen sketch: (optional)	

Use case 2: Search (mandatory)

Title:	Search result
Description:	This use case allows users to search for recipes. The user can also add ingredients that they want the recipe to include. The results will be showed in a new screen where the recipes that matches the search term best are displayed in order. The recipes will be shown the same way as in the home page.
Condition:	User does not need to be logged in to access this use case
Screen sketch: (optional)	

Use case 3: Detail (mandatory)

Title:	Detailed recipe

Description:	This page shows the same information about the recipes as before. However, you also get the completed recipe with ingredients and step by step instructions. At the bottom of the page you can also see the comments.
Condition:	User does not need to be logged in to access this use case
Screen sketch: (optional)	

Use case 4: Create

Implemented?[1]	Yes
Title:	Posting a recipe
Description:	The page used to post a recipe is composed of a form with four text fields. One where you put the title of the recipe, one where you list all the ingredients that you need, one where you write the step by steps and one extra field where you can write tips, comments etc.
Condition:	User needs to be logged in to access this use case
Screen sketch: (optional)	

Use case 5: Update

Implemented?[1]	Yes
Title:	Updating a recipe Commenting a recipe
Description:	Only the user that has uploaded the recipe can update it. When you update the recipe, you return to the same page as when you post it. All the previous information will still be there, but you can delete it if you want. To fulfill the update of the page the user can save it. The original recipe will be updated with the date of the update. The visitors of the page can add comments and grade the recipe from one to five in a field.
Condition:	User needs to be logged in to access this use case
Screen sketch: (optional)	

Use case 6: Delete

Implemented?[1]	Yes
Title:	Deleting a recipe Deleting a comment

Description:	The only person that can delete a recipe is the creator of it. The user have to go to the recipe´s detail screen where there will be a delete button that is followed by a confirmation button. When deleting a comment on a post you must return to the recipe's detail screen. On the comment there will be a button that you can press on to delete. The only people that can delete a comment is the owner of the recipe and the person that has commented.
Condition:	User needs to be logged in to access this use case
Screen sketch: (optional)	

Project Proposal – Database Description

Here the students should provide a list containing the tables/classes expected to be included in the project. For each table, should provide a description, the list of relationships (if any). There also should be a list of columns and their respective data type and any specifics of the columns (whether they are required, foreign keys, length limit, etc.) Comments can be added to the tables/columns if needed.

***Note**: The names of the tables always have capitalized names (e.g., ThisIsASuitableName) and are in singular form.*

Table:	**User**	

Description:	Contains the user information and enables the login. Each registered user becomes a row in this table.			
Relationships:	· Recipe/ one-to-many / a user can publish several recipes; each recipe can be reviewed by everyone but only changed by a single user. · Comment + grade / one-to-many / one user can have zero, one or many comments, while a comment and a grade has one user.			
Column name	**Data type**	**Constraints**	**Default**	**Comment**
id	Integer	primary key, not null	auto increment	
name	String(40)	not null		
email	String(120)	unique, not null		
password	String(60)	not null		Hashed password
Profile	integer	Not null	1	1: normal user 2: admin user

Table:	**Recipe**	

Description:	Contains information related to each recipe posted by the users. Each recipe becomes a row in this table. To be able to create a post, a user needs to be registered and logged in.			
Relationships:	· User / many-to-one / one recipe has one user, while a user has zero, one or many recipes · Comment / one-to-many / one recipe can have zero, one or many comments, while a comment Is associated with one recipe. · Grade / one-to-many / one recipe can have zero, one or many grades, while a grade is associated with one recipe.			
Column name	**Data type**	**Constraints**	**Default**	**Comment**
Id	Integer	primary key, not null	Auto increment	
title	string(100)	Not null		
date_posted	DateTime	Not null		
Content	Text	Not null		
User_id	Integer	Foreign key (User), not null		
User	Relationship	User, backreef('recipes')		

Table:	**Comment**			
Description:	Contains information related to each comment that a user makes about a post. Each comment posted in the system will become a row in this table. Every comment has a user and a post associated with it. To clarify it is associated with the user who wrote the comment and the post the comment is about. To post a comment, the user must be registered and logged in.			
Relationships:	· Post / many-to-one / one comment is about one post, while a post can have zero, one or many comments · User / many-to-one / one comment is written by one user, while a user can make zero, one or many comments			
Column name	**Data type**	**Constraints**	**Default**	**Comment**
Id	Integer	Primary key, not null	Auto increment	
Content	Text	Not null		
Date_posted	Datetime	Not null	Utcnow	
User_Id	Integer	Foreign key (user), not null		
User	Relationship	User, backreef('comments')		
Recipe_id	Integer	Foreign key (user), not null		

Recipe	Relationship	Post, backreef(‘comments’)		

Table:	**Grade**			
Description:	Contains information related to each comment that a user makes about a recipe. Each grade posted in the system will become a row in this table with the comment it is posted with. To grade a recipe, the user must be registered and logged in.			
Relationships:	· Comment / one-to-one / one grade is about one comment, and a comment is about one grade			
Column name	**Data type**	**Constraints**	**Default**	**Comment**
Id	Integer	Primary key, not null	Auto increment	
Content	integer	Limit 1 to 5		
User_id	Integer	Foreign key (user), not null		
User	Relationship	User, backreef(‘grades’)		
Recipe_id	Integer	Foreign key (post), not null		

Recipe	relationship	Recipe, backreef(‘grades’)		

Database execution of the final project

Follow the instructions in the assignment page.

Final Project

Follow the instructions in the assignment page.

[ ] I allow my final project to be used as example in the next instances of this course.

[ ] While sharing our project with other students, I want the names of the students to not be associated with the project (anonymized project).

---

[1] As an optional use case, please mark whether or not your group is planning to implement this use case. Please update it in the final version of this document to reflect the final submitted version of your project.