Matteus Laurent, Johan Levinsson, Oscar Petersson, Erik Peyronsson

MatLabb - Designspecifikation

Högskoleingenjörsutbildning i datateknik, 180 hp

Designspecifikation - 6 oktober 2015
Programmeringsprojekt, HT15
TDDI02, Linköpings universitet

Handledare:
Johan Frimodig
Institutionen för datavetenskap

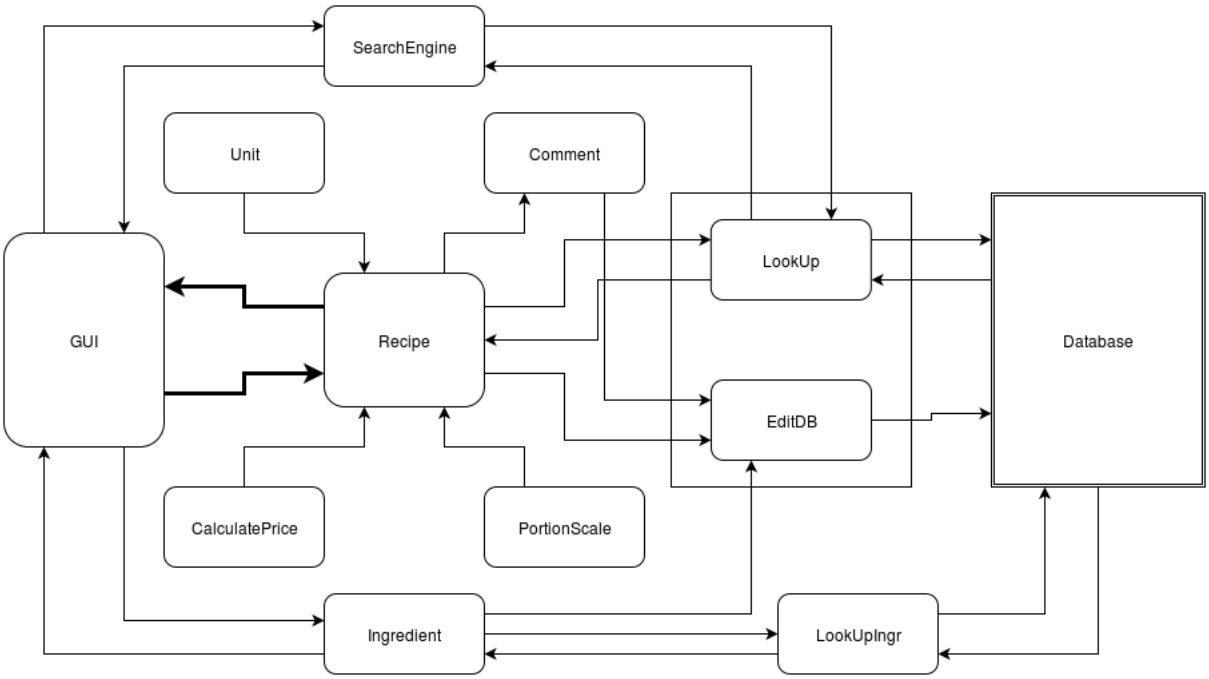
Innehåll

1. Inledning

Alla har vi någon gång stått framför vårat kylskåp och funderat över vad man kan hitta på för middag med det kylskåpsinnehåll vi konfronteras med. Projektet MatLabb är en interaktiv receptdatabas med grafiskt användargränssnitt. Dess syfte är att hjälpa användaren att organisera recept, söka recept baserat på tillgängliga ingredienser, samt att underlätta portions- och enhetsomvandling. Recepten kommer även att innehålla information om exempelvis näringsinnehåll och pris.

MatLabbs målbild presenteras närmare i dokumentet "MatLabb – Kravspecifikation", medan detta dokument närmare presenterar skalet och vad som finns under detsamma.

2. Arkitektur



Figur 2.1: Översikt över första lagrets moduler.

MatLabb har tre centrala moduler:

1. En databas som lagrar all receptrelaterad information.
2. En receptmodul (`Recipe`) som genom hjälpmoduler hämtar, skriver och behandlar databasens information.
3. Ett användargränssnitt (`GUI`) som hjälper användaren att på ett intuitivt och välbekant sätt interagerar med receptmodulen, och i förlängningen databasen.

   I detta kapitel ges en överblick över funktionaliteten hos modulerna i första "lagret" och hur dessa interagerar.

2.1 Databas

Databasen utgörs av en MySQL-databas där recept, ingredienser och relaterad data lagras. Exakt data som lagras framgår i kravspecifikationen, samt i kapitel 3 - *Detaljerad teknisk specifikation.*

2.2 Recipe

`Recipe` utgör det primära navet mellan databasen och användargränssnittet (`GUI`). Här behandlas aktuellt recept enligt de instruktioner som mottas via GUI:t. Modulen interagerar direkt med samtliga moduler förutom `SearchEngine`, `Ingredient` och `LookUpIngr`.

2.3 GUI

`GUI` är en abstraktion av det grafiska användargränssnittet och i förlängningen användaren. "Modulen" kommunicerar med tre olika grenar av MatLabb:

- `SearchEngine` - för sökning av recept
- `Recipe` - för interaktion med recept
- `Ingredient` - för interaktion med enskilda ingredienser.

2.4 SearchEngine

`SearchEngine` är MatLabbs sökmodul för recept. Den sköter direkta receptnamnssökningar genom `LookUp`.

2.5 Unit

`Unit` är en modul vars syfte är att konvertera enheter. Den ska kunna hantera prefixbaserade enheter (ex. deciliter $\leftrightarrow$ liter) och bör kunna hantera "köksmått" (ex. matsked $\leftrightarrow$ tesked). Modulen ska inte kunna hantera omvandling mellan volymenheter och viktenheter då detta förutsätter känd densitet för ingredienserna i fråga, något som sällan är tillgängligt och av föga intresse.

`Unit` används utav `Recipe` i samspel med `PortionScale`, samt i samspel med `CalculatePrice`.

2.6 Comment

`Comment` är en modul vars syfte är att hantera receptkommentarer. Modulen används utav `Recipe` för att vidarebefodra relevant information till `EditDB`.

2.7 CalculatePrice

`CalculatePrice` är en modul som med hjälp av prisdata beräknar portionspriset för aktuellt recept. Modulen används utav `Recipe` i samspel med `PortionScale`.

2.8 PortionScale

`PortionScale` är en modul som behandlar inmatad portionsskalning. Modulen används utav `Recipe` i samspel med `Unit` och `CalculatePrice`.

2.9 Ingredient

`Ingredient` kontrollerar ingredienshantering. Den hämtar information från databasen via `LookUpIngr` och redigerar databasposterna via `EditDB`.

2.10 LookUp, LookUpIngr

Dessa två moduler är närbesläktade och kan möjligtvis slås ihop till en. Deras syfte är att navigera databasen och leverera hämtad information till den modul som behöver den (`SearchEngine`, `Recipe` eller `Ingredient`).

2.11 EditDB

`EditDB` styr skrivning till databasen. Den används utav `Recipe`, `Comment` och `Ingredient`.

3. Detaljerad teknisk specifikation

Programmet MatLabb har i stort tre delsystem:

* GUI, Det grafiska användargränssnittet * Shell, eller, det inre skalet som håller centrala objekt och variabler, t.ex. aktivt recept och portionskalning. * Lookup, som tillhandahåller relevanta verktyg för kommunikation med databasen.

GUI ansvarar för att skriva ut data från det inre systemet på skärmen i grafisk tappning. Användaren styr även systemet genom att interagera med menyer, knappar och strängfält snarare än att ge skriftliga hänvisningar till kommandoprompten. Information presenteras enligt konceptbilderna i (referens till sektion med bilder). Det grafiska gränssnittet implementeras med hjälp av biblioteket Qt och kommer dels designas i klienten Qt Creator.

Shell innehåller objekt av klasserna Recipe, Ingredient och Lookup som datamedlemmar, där de två förstnämnda reflekterar det aktuella receptet och ingrediensen som vårt program interagerar med. Utåt tillhandahåller Shell publikt endast funktioner som kan tänkas motsvara alla möjliga handlingar från användaren. Denna grupp av funktioner inkluderar, men är ej begränsade till, följande:

* addRecipe(string) // Konstruerar ett nytt tomt Recipe-objektet för datamedlemmen currentRecipe_. Ett defaultargument av datatypen string existerar för att potentiellt tilldela ett namn. * editRecipe() // Kallar på currentRecipe_.editRecipe() för att kunna ändra på dess datamedlemmar. * importTxt(string) // Importering från textfil. Konstruerar ett nytt Recipe-objekt och försöker fylla i dess datamedlemmar enligt en standardmodell. * exportTxt() // Kallar på currentRecipe_.exportTxt(string) för att exportera till .txt. Kan modifieras för att först hämta ett annat recept från databasen för exportering. * addIngredient(string) * editIngredient(string)

* matchRecipe(string) // Levererar en sträng till Lookup-objektet för att slå i databasen för exakt matchning. * matchIngredient(string) * searchRecipe(cont¡SearchTerm*¿) // Levererar söktermer i en godtycklig container till Lookup. recipeSearchResults_ tilldelas det resultat som Lookup ger.

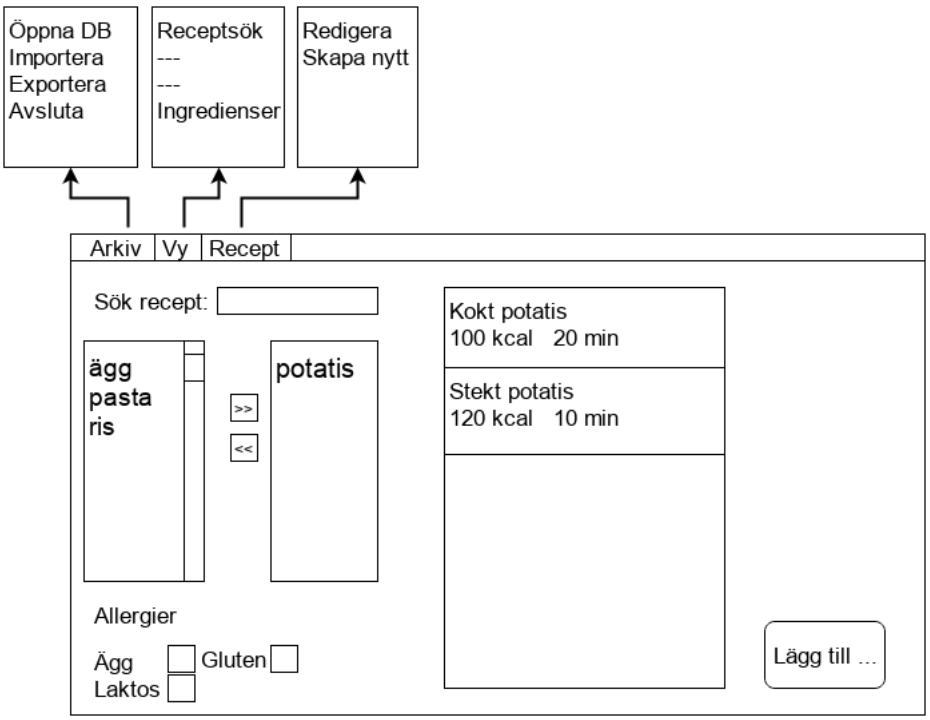
* get-funktioner som används av GUI:t och returnerar relevant data.

Objekt av klasserna Recipe och Ingredient kommer endast att existera i stabilt tillstånd som datamedlemmar i klassen Shell. Nya objekt skapas antingen i samband med t.ex. funktionen addRecipe(string) eller byggs upp och returneras av Lookup som resultat av en matchning i databasen. Klasserna Recept och Ingredient innehåller främst fullständig data om ett specifikt recept/ingrediens, men även funktioner för implementeringen av Shell's "användarfunktioner", t.ex. åtkomst och redigering.

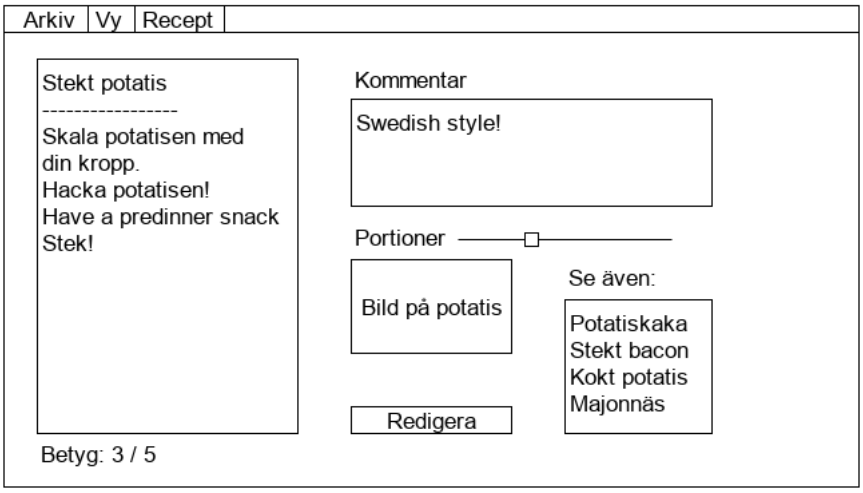
Endast ett objekt av klassen Lookup existerar i programmet och då som datamedlem av Shell. Lookup ska inte hålla koll på några värden, utan fyller istället uppgiften att avgränsa funktionalitet och samla funktioner som har att göra med databaskommunikation på ett ställe.

// Lookupfunktioner?

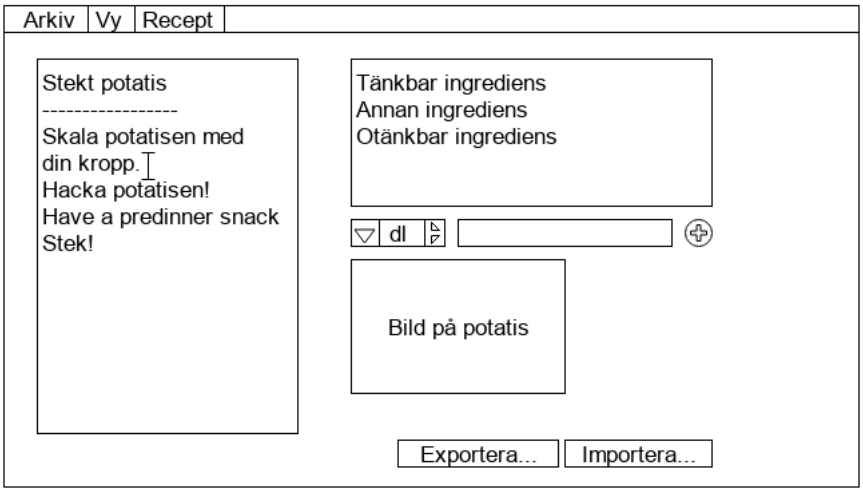
4. Design av användargränssnitt



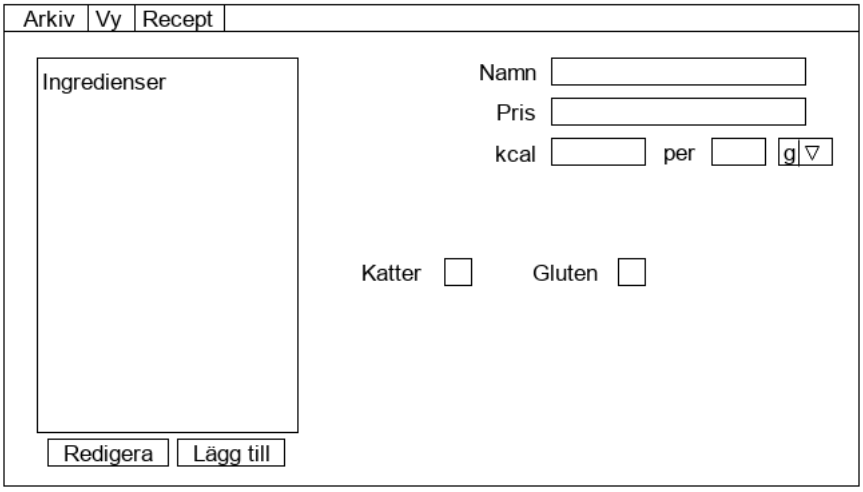
Figur 4.1: Sökfönster



Figur 4.2: Receptfönster



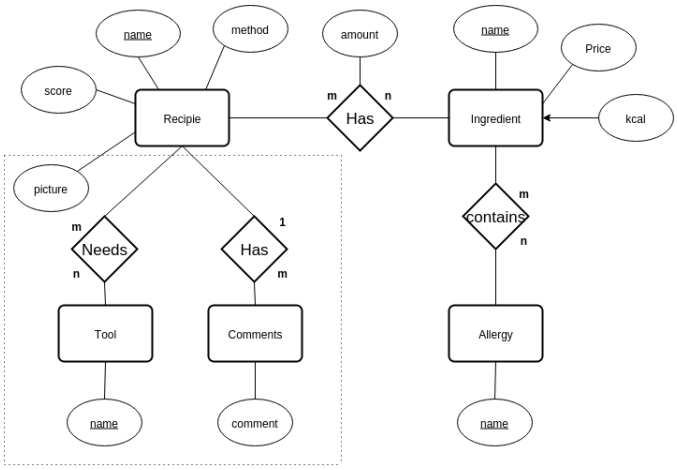
Figur 4.3: Redigering av recept



Figur 4.4: Redigering av ingredienser

5. Design av databas

Då all receptinforamtion behöver sparas mellan körningar av programmet behöver den lagras externy. I det här programmet kommer en databas användas med hjälp av MySQL. Databasen kan överskådas i EER-diagramet i 5.1. Den del som ligger inom de streckade området hör till funktionalitet som endast kommer implementeras i mån av tid.



Figur 5.1: Entity-Relationship diagram

Databasen kommer bestå av fem entiteter `Recipe` som innehåller information unik för ett enskilt recept. `Ingredient` som är en lista över de olika ingredienser som databasen innehåller, `Allergy` som är en lista över allergier, `Tool` som är en lista över allergier samt `Comment` som är en lista över kommentarer till varje recept.

Entiteten `Recipe` är den entiteten som lagrar namn, beskrivning, bild tillagningsmetod, då recept skall ha unika namn för att särskilja dem åt är det receptets namn som agerar primärnyckel.

`Recipe` har en m-n relation till `Ingredient` vilket ger oss en lista på ingredienser till vajre recept. Genom att ha attributen `kcal` och `price` på entiteten `Ingredients` istället för `Recipe` behöver unik information om portionspris och närings-inehåll till varje recept inte sparas utan kan räknas ut beroende på vilka ingredienser som ingår. Genom att tillföra attributet `amount` behöver varje ingrediens endast lagras en gång per recept och enhetsomvandling och portionsskalning kommer vara möjlig.

För att hålla ordning på de vanligaste matallergierna (samt kött mejeri och fisk för veganer/veganer) finns entiteten `Allergy`. Den har även en 1-n relation till `Comment` vilket resulterar i att alla recept får en lista med komentarer skrivna av användaren. samt en m-n relation till `Tool` som ger en lista över de redskaps som behövs.

Genom att utforma databasen enligt sagda modell kommer programmet kunna utföra sökningar olika sökningar och filtreringar på ingredienser som ingår, inte ingår, eventuella allergener etc.