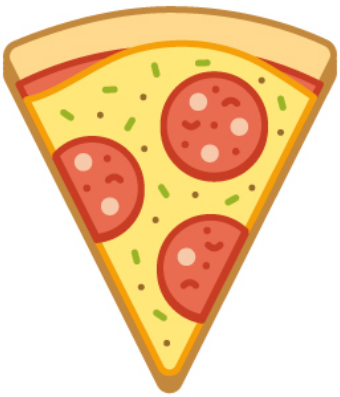
Mozzarella - webeC Project

Marius Küng (marius.kueng@students.fhnw.ch)

24.03.2015

Contents

1 Introduction



1.1 Motivation

Mozzarella is a collaborative shopping-/what's in your fridge list app. Mozzarella aims to help the user keep track of what groceries he has

in his fridge, which of these he should consume and what he needs to buy soon. Because of shared lists it's able to keep track of an entire household.

The idea originates from the problem that a family has several frigdes, freezers and kitchen cabinets with loads of stuff in it. It's super easy to loose track of what's already at home and should be consumed next. Otherwise you throw away way to much groceries.

With this you can easily save money and help the environment by not throwing away food.

1.2 Features (v1)

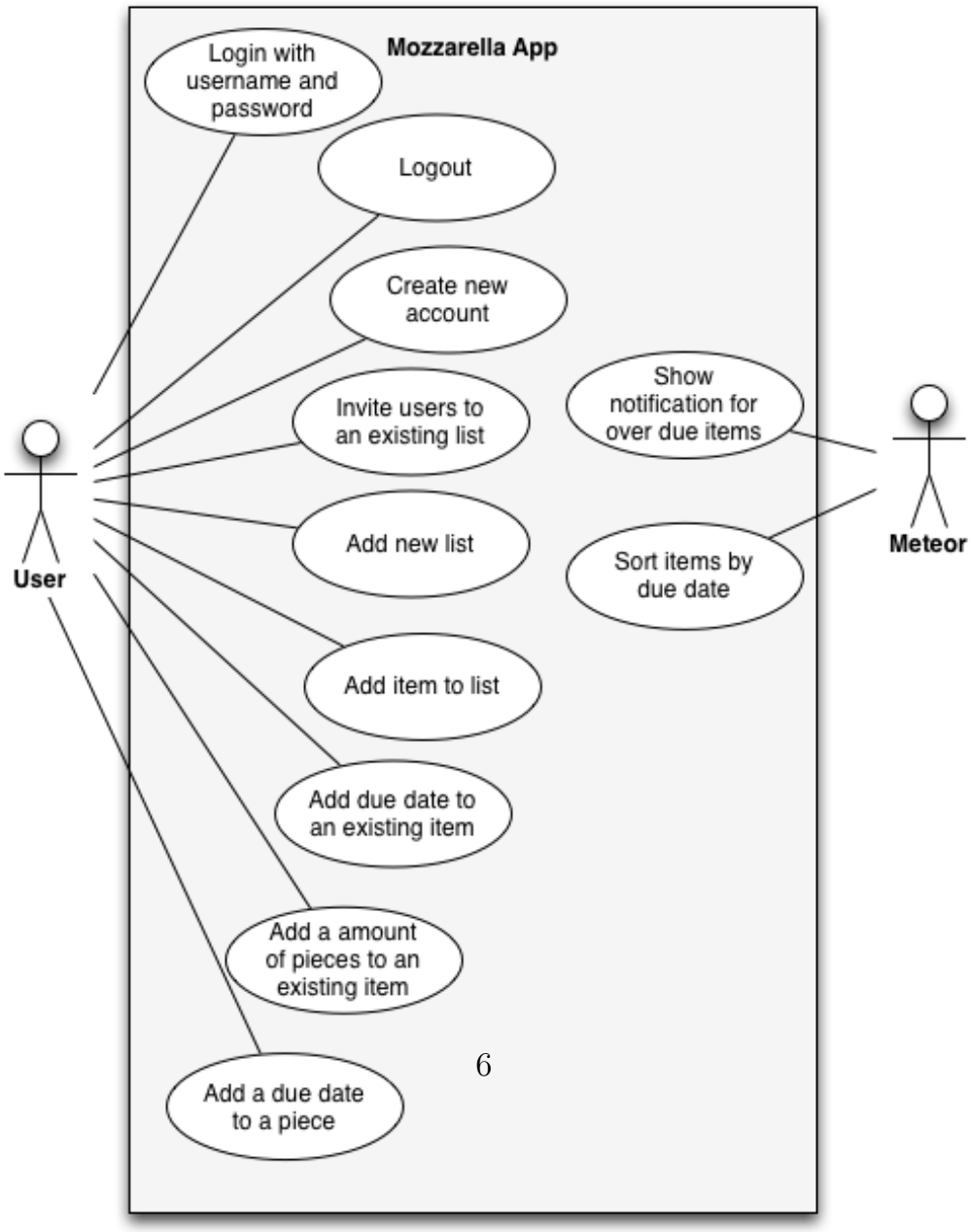
In the first version the following features will be implemented:

- **Shared lists** with unlimited users. Invite everyone you want. For example a you can manage lists for ***at home***, ***weekend bbq***, or the ***birthday party***.
- **Item entries** in every lists. Each item has a title, a piece counter ***(how many apples do I have?)***, and an overview of all pieces ***(1 & 3 pound pack of ground meat)***
- **Due-date notifications** which inform you easily which items you should consume next.

1.3 v2

2 Requirements

2.1 Usecase Overview



2.1.1 Usecase: Login with username and password

Use Case	Login with username and password
Description	Allows user to login with his username + password
Actors	User
Preconditions	Not logged in
Basic Flow	User fills in username + password, clicks login
Alt. Flow	None
Postconditions	*User is logged in (has session token)*
Notes	-

2.1.2 Usecase: Logout

Use Case	Logout
Description	Logout user from running current session.
Actors	User
Preconditions	Logged in
Basic Flow	User clicks logout button.
Alt. Flow	-
Postconditions	*User is logged out (session token removed)*
Notes	-

2.1.3 Usecase: Create new account

Use Case	Create new account
Description	A user creates a new account.
Actors	User
Preconditions	User doesn't have an account yet.
Basic Flow	User clicks "create new account" button.
Alt. Flow	User tries to login but no existing account was found.

Use Case	Create new account
Postconditions	UC: Login
Notes	-

2.1.4 Usecase: Add new list

Use Case	Add new list
Description	A user creates a new list.
Actors	User
Preconditions	UC: Login
Basic Flow	User clicks “add new list” button.
Alt. Flow	-
Postconditions	List is now added and visible to the user
Notes	-

2.1.5 Usecase: Invite user to existing list

Use Case	Invite user to existing list
Description	User shares a list with other users.
Actors	User
Preconditions	The invited user exists (username).
Basic Flow	User clicks the “Share list” button and enters the username of a user.
Alt. Flow	-
Postconditions	List is now shared.
Notes	-

2.1.6 Usecase: Add item to list

Use Case	Add item to list
Description	A user adds a new item to an existing list.
Actors	User
Preconditions	UC: Login, UC: Add new list
Basic Flow	User fills in the “new item” input field an submits it.
Alt. Flow	User adds a **due date** and a **piece amount** to the new item.
Postconditions	Item is now added to the existing list and visible to the user. Add amo
Notes	-

2.1.7 Usecase: Add due date to an existing item

Use Case | Add due date to an existing item ——————|——————— **Description** | A user adds a due date to an existing item. **Actors** | User **Preconditions** | UC: Login, UC: Add new item **Basic Flow** | User hovers over an item and clicks the “add due date (clock)” button **Alt. Flow** | User double clicks the current due date and changes it. Due date can be null. **Postconditions** | - **Notes** | -

2.1.8 Usecase: Add amount of pieces to an existing item

Use Case	Add amount of pieces to an existing item
Description	A user adds an amount of pieces to an existing item.
Actors	User
Preconditions	UC: Login, UC: Add new item
Basic Flow	User clicks on the arrows on the amount element.
Alt. Flow	If no amount is given, 1 is the default amount.
Postconditions	-
Notes	If the user adds a new item, a piece is automatically added with a refe

2.1.9 Usecase: Add due date to a piece

Use Case	Add due date to a piece
Description	A user adds a due date to a piece of an item.
Actors	User
Preconditions	UC: Login, UC: Add new item
Basic Flow	User opens item view and changes due date UC: Add due date to an e
Alt. Flow	-
Postconditions	-
Notes	-

2.1.10 Usecase: Show notifications for over due items

Use Case	Show notifications for over due items
Description	The app shows the user which items are over due.
Actors	Meteor (Backend)
Preconditions	UC: Add new item
Basic Flow	Meteor checks if any items of a user are over due and adds them to a s
Alt. Flow	Meteor shows the number of over due items in the over due list title.
Postconditions	-
Notes	-

2.1.11 Usecase: Sort items by due date

Use Case | Sort items by due date ——————|————— **Description** | The app sorts all items in each list by its due date **Actors** | Meteor (Backend) **Preconditions** | UC: Add new item **Basic Flow** | Old items are on top, new items below. **Alt. Flow** | - **Postconditions** | - **Notes** | - # Software Architecture

2.2 Conceptual Data Model

- Entities
- Relationships
- Attributes

2.3 Operations

- Actor
- Operations