**Universität Bern**

Introduction to Software Engineering

MensaUniBE

Software Requirement Specification

v1.0

Last modified on 11.11.2013

**Customer**  
Bledar Aga

**Technical Tutor**  
Andrei Chis

**Autors**  
Raul Bolaños, Nicolas Kessler, Theodor Truffer, Lukas Zahnd

Table of Contents

[Table of Contents 1](#_Toc368778285)

[Changelog 2](#_Toc368778286)

[Introduction 3](#_Toc368778287)

[Use Cases 3](#_Toc368778288)

[#1 View mensa list 3](#_Toc368778289)

[#2 View mensa details 4](#_Toc368778290)

[#3 Show upcoming menus 4](#_Toc368778291)

[#4 View Map 4](#_Toc368778292)

[#5 Find mensa 5](#_Toc368778293)

[#6 Set favourite mensa 6](#_Toc368778294)

[#7 Add Friends 6](#_Toc368778295)

[#8 Delete Friends 7](#_Toc368778296)

[#9 Notify Friends 8](#_Toc368778297)

[#10 Join friends for lunch 9](#_Toc368778298)

[#11 Rate a menu 9](#_Toc368778299)

[#12 Menu’s rating 10](#_Toc368778300)

[#13 Favourite food 10](#_Toc368778301)

[#14 Set default language 11](#_Toc368778302)

[#15 Switch Language 11](#_Toc368778303)

[Use Case Diagram 13](#_Toc368778304)

[Specific requirements 14](#_Toc368778305)

[Functional requirements: 14](#_Toc368778306)

[Non-functional requirements: 14](#_Toc368778307)

# Changelog

|  |  |
| --- | --- |
| **Date** | **Changes** |
|  |  |
| 11 Nov. 2013 | * Project Restructuring / Cleanup * Added menu rating dialog * Added pull to refresh functionality to fetch recent data |
| 05 Nov. 2013 | * Implemented first tests |
| 30 Oct. 2013 | * Refactored Model to make use of improved REST API * Refactored WebService for better exception handling |
| 29 Oct. 2013 | * Refactored database and model |
| 27 Oct. 2013 | * Fixed crashing map fragment |
| 26 Oct. 2013 | * Added notification button in menu bar * Added system settings for username and startscreen configuration * Implemented device/user identification and storage in API * Implemented calculation of nearest mensa * Improved offline storage of data |
| 25 Oct. 2013 | * Implemented ViewPager (Swiping tabs) for the MensaDetail view * Implemented the MensaMap functionality and location services * Implemented basic offline capability via database * Refactored project to use support library for older devices * Project Cleanup * Implemented own REST API for more power and flexibility |
| 23 Oct. 2013 | * Added favorite mensa functionality to UI * Added handling of back button of the device * Added custom list adapter to customize the look of the list items * Project Cleanup |
| 22 Oct. 2013 | * Implemented Favorite Mensa functionality * Implemented MensaDetail view with Menus |
| 18 Oct. 2013 | * Project cleanup |
| 16 Oct. 2013 | * Project refactoring and reorganisation * Refactored WebService to an asynchronous task * Connect Application Data Model to UI and display real data * Refactored Application Data Model |
| 15 Oct. 2013 | * Implemented Application Data Model |
| 14 Oct. 2013 | * Added WebService & JSON Parsing functionality |
| 09 Oct. 2013 | * Created prototype screenshots and interactive wireframe |
| 08 Oct. 2013 | * Updated prototype |
| 05 Oct. 2013 | * Updated use cases according to the last meeting with customer * Corrected use case diagram * New use cases: “Delete Friend”, “Switch language” * Updated use cases to fit navigation prototype |
| 04 Oct. 2013 | * Created first prototype eclipse project |
| 02 Oct. 2013 | * Initial version of the document |

# Introduction

The application gives students, employees or any interested person information about the mensas of the University of Berne. Among these information should be the locations of all the mensas, how to go there from the current position and, of course, the daily menu and the ones of the following days with all its prices. The app will include basic social media functionality, such as inviting friends for lunch or rating the menus.

Stakeholders in this project are Mr [Bledar Aga](http://scg.unibe.ch/wiki/students/bledaraga), The university of Bern and The mensas in Bern.

The application can be run on Android based devices.

This application is designed to help students find the appropriate mensa to eat.

# Use Cases

(Index by importance)

## View mensa list

Actors: The app-user.

Description: Get a list of all mensas on the campus.

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The user is able to see all available mensas on the campus

Basic Flow:

1. The user starts the application.
2. The user taps on “Mensas” in the menu and the application shows the list of all available mensas.

Notes:

1. The list displays the name of the mensas as clickable items in alphabetical order.
2. The list should feature a search box to give the user the ability to search for mensas. The searching process should be done in real-time i.e. after entering one letter the list should already show mensas that start with that letter.

## View mensa details

Actors: The app-user.

Description: Get detailed information about a certain mensa such as schedule, daily menu, menus of the next days, prices, address.

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The user is able to see the following information of a certain mensa:
   1. Schedule
   2. Menu of the day
   3. All menus of the current week (beginning on Saturday)
   4. Prices of these menus
   5. The address of this mensa

Basic Flow:

1. The user starts the application.
2. The user taps on “Mensas” in the menu and the application shows the list of all available mensas.
3. The user taps on the mensa he wants information about and the application shows all the details about that mensa.

## Show upcoming menus

Actors: The app-user

Description: Lets the user browse all upcoming menus of the week (beginning on Saturday), including the price of the menus and where they will be served.

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The user sees all upcoming menus for the next days.

Basic Flow:

1. The user starts the application.
2. The user taps on “Upcoming Menus” in the menu.
3. A list will be displayed with all menus of the week, sorted by the day when they will be served.

Notes:

1. If the user did not define any favourite mensas the application will automatically show the upcoming menus on the home screen

## View Map

Actors: The app-user.

Description: Show a map with all the mensas, also find and show the closest mensa from your current position.

Pre-condition:

1. Active Internet connection.
2. Active GPS on the device.

Post-condition:

1. The user should be provided with a map on which the closest mensa and the directions from the current position are marked.

Basic Flow:

1. The user starts the application.
2. The user taps on “Closest mensa” in the menu and the application shows a map with a mark for all mensas and directions to the closest mensa from the users position.

Alternative Flow:

2a. No active GPS on the device

1. The user will be prompted that there is no GPS connection and that this feature is only available with GPS. The message should read as follows:

“You do not have GPS activated on your device”.

Note that this feature only works with active GPS!”

Notes:

1. After opening the map, the mensas should all be clickable to calculate the directions for other mensas

## Find mensa

Actors: The app-user.

Description: Shows a map with all the mensas and indicates the way to a chosen mensa.

Pre-condition:

1. Active Internet connection.
2. Active GPS on the device.

Post-condition:

1. The user should be provided with a map on which the chosen mensa and the directions from the current position are marked.

Basic Flow:

1. The user starts the application.
2. The user taps on “Mensas” in the menu and the application shows the list of all available mensas.
3. The user taps the button to find a mensa which is displayed for every mensa in the list
4. The application shows the map and the shortest way to this mensa from the users current position.

Alternative Flow:

4a. No active GPS on the device

1. The user will be prompted that there is no GPS connection and that this feature is only available with GPS. The message should read as follows:

“You do not have GPS activated on your device. Note that this feature only works with active GPS!”

## Set favourite mensa

Actors: The app-user.

Description: Choose a favourite mensa, for which you can quickly see the current menu and the menus for the next days on the applications start screen.

Pre-condition:

1. No special conditions must hold.

Post-condition:

1. The user has a correctly set favourite mensa.

Basic Flow:

1. The user starts the application.
2. The user taps on “Mensas” in the menu and the application shows the list of all available mensas.
3. The user taps the star button which is displayed for every mensa in the list
4. Once this preference is set, the user gets all important information on the start-screen after launching the application

Notes:

1. The user should be able to have more than one favourite mensa.

## Add Friends

Actors: The app-user [A] and app-user [B]

Description: Add other users of this application to your friend list.

Pre-condition:

1. Active Internet connection.
2. The two users are not already friends.

Post-condition:

1. The two users are now set as their respective friends.

Basic Flow:

1. The user [A] starts the application
2. The user [A] taps on “Friends” in the menu and the application shows a list of all added friends
3. The user [A] taps on the button “add friend” in the action bar, enters a friend’s email address to add him to the list
4. The second [B] user will receive a push notification about the request
5. The second user [B] opens his application
6. The second user [B] taps on “Friends” in the menu
7. The second user [B] taps on “Friend Requests” and the application displays a list of all the unanswered requests
8. The second user [B] taps on one of the requests and can now either accept the request or ignore it.
9. If the request was accepted the first user [A] appears now in the second users [B] friend list and vice versa.

Alternative Flow:

3a. User [A] entered an invalid address (entered text was not an email address)

1. The user should be prompted that he entered an invalid address. The message should read as follows:

“You did not enter a correct email address!”

1. The user presses “Ok” and is returned to the add friend form.

3b. User [A] entered an email address that could not be resolved to any other app user

1. The user should be prompted that the other user could not be found. The message should read as follows:

“<Entered email address> could not be found. Try again!”

1. The user presses “Ok” and is returned to the add friend form.

6b. Alternatively the user [B] taps on the notifications button in the action bar on the app’s home, to open the request.

Notes:

1. The app should provide some sort of user name system instead of email addresses.

## Delete Friends

Actors: The app-user and one of his friends.

Description: Deletes one or more friends from the users friends list

Pre-condition:

1. Active internet condition
2. The two users are friends

Post-condition:

1. The selected friends are removed from the users friends list

Basic Flow:

1. The user starts the application
2. The user navigates to friends in the menu and the application shows a list of all added friends
3. The user taps and holds on a friend to select him. The user can also select multiple friends
4. The user taps on delete friend on the action bar
5. The user presses “Yes” in the upcoming message box. The message should read as follows:

“Really delete <x> friends?”

Alternative Flow:

5a. The user presses “No”

1. The selected friends will not be deleted and the app returns to the friends list

Note:

1. What should happen to the deleted friend? Will his friends list also be affected by this?

## Notify Friends

Actors: The app-user and his friends.

Description: Let all your friends, or specific ones, know where you go for lunch, so they can join you.

Pre-condition:

1. Active internet connection
2. The user must already have friends; otherwise this feature will not be available.

Post-condition:

1. All the chosen friends of the app-user will get a push notification where they can accept the invitation.

Basic Flow:

1. The user starts the application
2. The user taps on “Friends” in the menu and the application shows a list of all added friends
3. The user taps and holds on a friend to select him. The user can also select multiple friends
4. The user taps on the button “invite” in the action bar
5. The user chooses the mensa from a dropdown menu where he plans to go
6. The user chooses the time when he wants to eat
7. The user enters a message that will be sent to all invited friends and taps “send invitation”
8. The user gets a notification as soon as his friend joins or declines the event

## Join friends for lunch

Actors: The app-user and the friend who invited him.

Description: If a friend lets you know where he goes for lunch, you can join him.

Pre-condition:

1. Active Internet connection.
2. The user was invited by one of his friends.

Post-condition:

1. The person who sent the invitation will receive the answer of the user by push notifications

Trigger: The user receives an invitation for lunch from one of his friends

Basic Flow:

1. The user gets an invitation by push notification
2. The user starts the application
3. An icon in the action bar indicates that there are new notifications. The user taps this button.
4. The list with notifications opens. The user taps on the new invite. The details of the invite will be displayed.
5. The user taps either on “accept” or “decline”
6. The application sends a notification with the answer back to that friend

Alternative Flow:

3a. Alternatively the user navigates to “Friends” and then opens the tab “Invites” to open the invitation.

## Rate a menu

Actor: The app-user.

Description: Grade a menu you’ve eaten from zero to five.

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The new vote on the menu will be added to the average voting and is always displayed with that menu.

Basic Flow:

1. The user starts the application.
2. Either the user taps on a menu of his favourite mensa on the start-screen or taps on “Mensas”, taps on the mensa where he had lunch or taps on “Upcoming Menus”, chooses the menu he had.
3. The user taps on “Rate” and can now rate the menu from zero to five.

Notes:

1. Since a user may only vote once and the same menu could be served again at another day, the app must have a way to determine if two menus are the same and treat them as such.

## Menu’s rating

Actors: The app-user.

Description: See how other people rated a specific menu.

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The user knows the rating of a menu.
2. The rating of the menu should not be affected by this.

Basic Flow:

1. The user starts the application.
2. Either the user taps on a menu of his favourite mensa on the start-screen or taps on “Mensas”, taps on the mensa where he had lunch or taps on “Upcoming Menus”, chooses the menu he had.
3. The user can see next to the menu how many people have rated this menu and the average grade.

## Favourite food

Actors: The app-user

Description: Get notified when your favourite mensa serves your favourite food

Pre-condition:

1. No special conditions must hold other than an active Internet connection.

Post-condition:

1. The user gets a push notification as soon a menu matches the criteria.

Basic Flow:

1. The user starts the application
2. The user taps on “Settings”
3. The user taps on “Food criteria”
4. The user can choose from a list of ingredients his preferred ones
5. Once this preferences are set, the user gets a push notification when his favourite mensa serves a menu containing at least one of the user’s favourite ingredients, also indicating how many criteria matched

Notes:

1. The user should be able to set up ONE criteria consisting of one or more ingredients.

## Set default language

Actors: The app-user.

Description: Change the default language in which the menus should be displayed.

Pre-condition:

1. No special conditions must hold.

Post-condition:

1. The menus will be displayed in the given language

Basic Flow:

1. The user starts the application
2. The user taps on “Settings”
3. The user taps on “Menu Language”
4. The user can choose between “Deutsch (Standard)” or “English”

## Switch Language

Actors: The app-user.

Description: Switch the language of a menu from English to German or vice versa

Pre-condition:

1. No special conditions must hold other than an active internet connection.

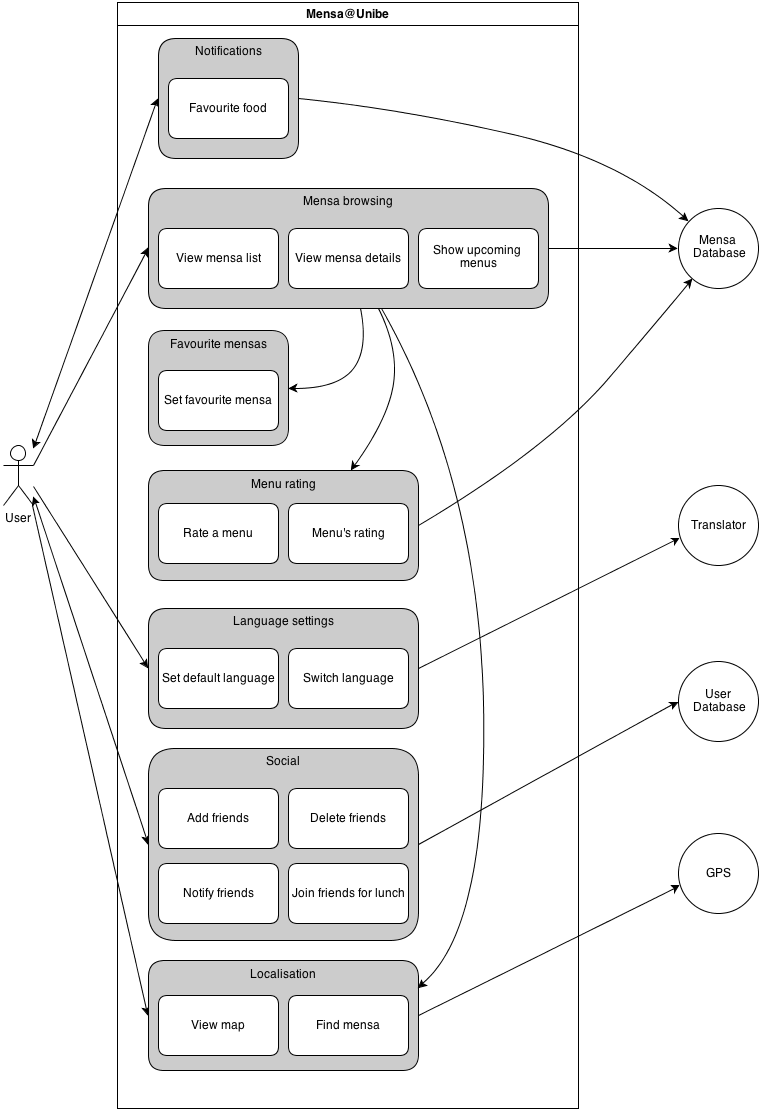
Post-condition:

1. The menu is displayed in German if it was in English or vice versa

Basic Flow:

1. The user starts the application
2. Either the user taps on a menu of his favourite mensa on the start-screen or taps on “List”, taps on the mensa where he had lunch or taps on “Upcoming Menus”, chooses the menu he had.
3. The user taps on the button displaying a German flag or a union jack.

## Use Case Diagram



# Specific requirements

## Functional requirements:

1. List showing all the mensas.
2. Possibility to retrieve the information of mensas from a server.
3. Possibility to retrieve menus.
4. Calculate the closest mensa from current position.
5. Show a mensa on the map.
6. Show the directions to a mensa on the map.
7. Settings for the favourite mensa.
8. Database with the users.
9. Local friends list.
10. Feature to add friends and accept friend requests.
11. Feature to invite friend for lunch and accept invitations.
12. Feature to rate menus.
13. Database to store menu ratings.
14. Settings for food criteria.
15. Feature checking upcoming menus for food criteria.
16. Setting for preferred menu language.
17. Translator for menus.

## Non-functional requirements:

1. The application should always be up-to-date: Every week, the menus of each mensa should be updated. Also the schedule, prizes, addresses and employees of the mensas should be updated if they changed.

2. The whole application should be clearly arranged and easy to understand, so that everyone can use it intuitively. Furthermore the design should be attractive.

3. It should be dependable. If you invite friends, it should be certain that the friends get the invitation and if your favourite menu is being served in a mensa, you should be informed.

4. The performance of the application should be good. It should be fast and you should get quick responses to your requests, so that it’s comfortable to use.

5. The application should be compatible with both newer and older versions of Android. Also, users with a newer version should be able to interact with users with an older version of Android.

6. Privacy should be granted. E.g. if you rate a menu, nobody should be able to see how you rated that menu, only how many people have rated and what the average rating is.