**Project:** Mensa@Unibe

**Customer:** Bledar Aga

**Developers:** Raul Bolanos, Manuela Häfliger, Nicolas Kessler, Theodor Truffer

**Last Update:** Tuesday, 01 October 2013

**Software Requirement Specification**

1. **Introduction**

The application gives students, employees or any interested person informations about the mensas of the University of Berne. Among these informations should be the locations of all the mensas, how to get 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 are anyone who has an interest in the project. Project stakeholders are individuals and organizations that are actively involved in the project, or whose interests may be affected as a result of project execution or project completion. They may also exert influence over the project’s objectives and outcomes. The project management team must identify the stakeholders, determine their requirements and expectations, and, to the extent possible, manage their influence in relation to the requirements to ensure a successful project.

<http://en.wikipedia.org/wiki/Project_stakeholder>

( danach löschen)

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 will be running on Android based devices.

The goal of this application is to help students find the right mensa to have lunch.

1. **Use Cases**

(index by importance)

**#1**

**Name: 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 “List” in the action bar 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**

**Name: 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, employees, 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. menu of the next days
   4. prices of these menus
   5. employees of this mensa
   6. the address of this mensa

Basic Flow:

1. The user starts the application
2. The user taps on “List” in the action bar 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

Notes:

1. What days are included for “the menu of the next days”

**#3**

**Name: Show closest mensa**

Actors: The app-user

Description: 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 action bar and the application shows a map with a mark for the closest mensa from the users position and directions to that mensa.

Alternative Flow:

2a. No active GPS on the device

1. The user will be promted 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. This use case is a shortcut for use case #4 where automatically the nearest mensa will be displayed

**#4**

**Name: Find mensa**

Actors: The app-user

Description: Shows the way on the map 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 choosen mensa and the directions from the current position are marked.

Basic Flow:

1. The user starts the application
2. The user taps on “List” in the action bar and the application shows the list of all available mensas
3. The user chooses the mensa he wants to go and the application shows information about this mensa
4. The user clicks a button to find this mensa and 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 promted 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!”

**#5**

**Name: 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 “Settings”
3. The user taps on “Favourite mensa”
4. The user can choose his favourite mensa from a list of available mensas
5. Once this preference is set, the user gets all important information on the start-screen after launching the application

Notes:

1. If the user already had an favourite mensa the old one should simply be overwritten

**#6**

**Name: Add Friends**

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

Description: Add other users of this application to your friendlist.

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 action bar and the application shows a list of all added friends
3. The user [A] taps on the button “add friend”, 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 action bar
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 sould 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.

Notes:

1. Should the friends be listed with their email address or should a user name system be provided?

**#7**

**Name: 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 action bar and the application shows a list of all added friends
3. The user taps on the button “invite”, chooses all or individual friends and the mensa he plans to go to
4. The list of friends now provides checkboxes to chose which friends should be invited. Then the user taps “next”
5. The user chooses the mensa where he plans to go and taps “next”
6. The user chooses the time when he wants to eat and taps “send invite”
7. the user gets a notification as soon as his friend joins or declines the event

Alternative Flow:

3a. The user has not any friends yet

1. The button “invite” is not clickable and no invites possible

**#8**

**Name: 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 invite will receive the answer of the user by push notifications

Trigger: The user receives an invite 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. Above the daily menu of the user’s favourite mensa appears the friend’s invitation with two buttons “join” or “decline”
4. The application sends a notification with the answer back to that friend

**#9**

**Name: 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 “List”, 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. How to determine if a menu was already served and rated in the past? Is this needed for this app?

**#10**

**Name: 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 “List”, 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

**#11**

**Name: 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. How many criteria should a user be able to set?
2. What is the exact nature of the criteria? (general: vegetarian, meat, fish. Or more detailed: rice, lamb, pork, potatoes, carrots etc.)

**#12**

**Name: Show upcoming menus**

Actors: The app-user

Description: Lets the user browse all upcoming menus of the week, 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 action bar
3. A list will be displayed with all menus of the week, sorted by the day when they will be served.

Notes:

1. For how many days should the menus be displayed? Only for the active week or more?

**#13**

**Name: Set language for menus**

Actors: The app-user

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

Pre-condition:

1. No special conditions must hold.

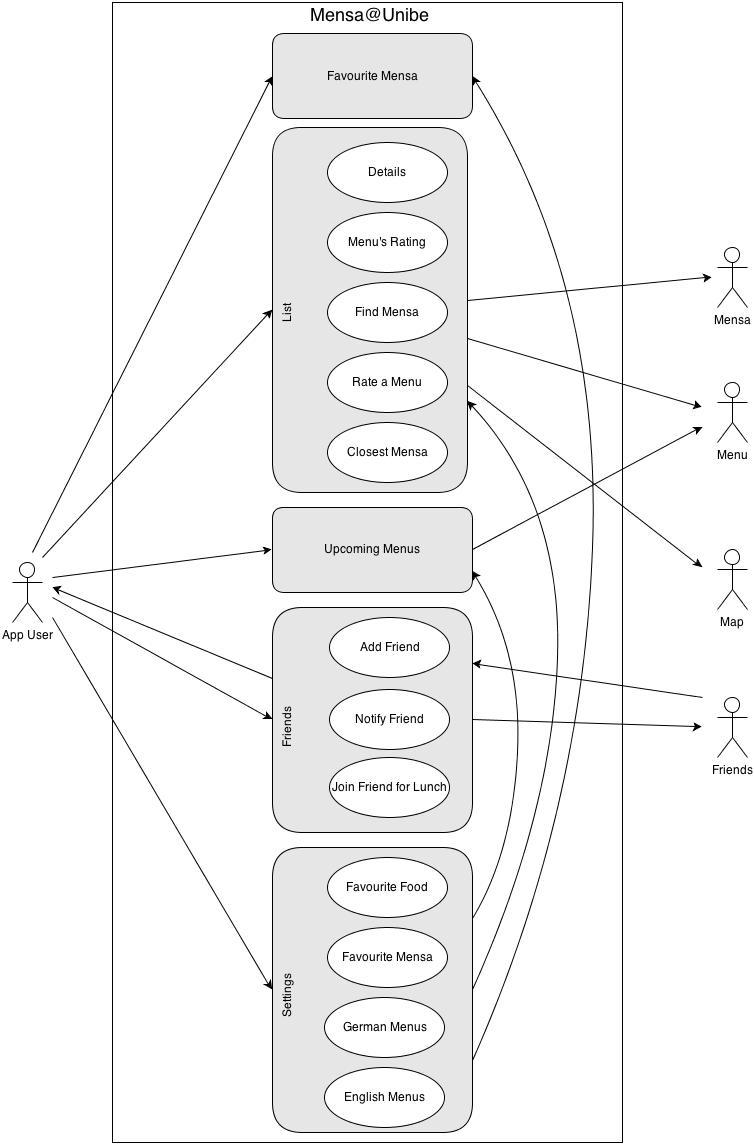
Post-condition:

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 (Standart)” or “English”

**Use Case Diagram**



1. **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 all the users
9. Local friends list
10. Feature to add friends and accept friend requests
11. Feature to invite friend for lunch and accept invites
12. Feature to rate menus
13. Database to store menu ratings
14. Settings for food criterias
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, the prizes, the address and the employees of the mensas should be updated if 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.