**Project Requirements**

**Project Name: Tutor Negotiating**

**Team:** ese2015-team7

**Customer:** Niklaus

Revision History

| **Version** | **Date** | **Revision Description** |
| --- | --- | --- |
| .01 |  |  |
| .02 |  |  |
| .. |  |  |
| 1.0 |  |  |
|  |  |  |
|  |  |  |

Date: October 5, 2015

Table of Contents

[1 Introduction 3](#__RefHeading__616_1576712830)

[1 Purpose 3](#__RefHeading__618_1576712830)

[2 Stakeholders 3](#__RefHeading__620_1576712830)

[3 Definitions 3](#__RefHeading__622_1576712830)

[4 System overview 3](#__RefHeading__624_1576712830)

[5 References 3](#__RefHeading__626_1576712830)

[2 Overall description 4](#__RefHeading__628_1576712830)

[A. Diagram 4](#__RefHeading__630_1576712830)

[B. Use cases 5](#__RefHeading__632_1576712830)

[3 Specific requirements 7](#__RefHeading__634_1576712830)

[1 Functional requirements 7](#__RefHeading__636_1576712830)

[2 Non-functional requirements 7](#__RefHeading__638_1576712830)

# Introduction

## Purpose

The goal of this project is to build a web portal targeted to university students who are preparing for a specific exam. The portal/platform connects students with tutors so they get well prepared for exams and improve their knowledge as well as understanding for specific lectures.

## Stakeholders

* Students
* Tutors
* Customer Niklaus
* ese2015-team7

## Definitions

## System overview

The system provides services to sign up and login. When signing up, there's no difference between signing up as a student or tutor. Both user types have to fill in basic information like name, password, email and user name. It should also be possible to extend basic- with optional information like profile picture, short biography, grades, current semester and available time slots.

Once logged in, a selection based search can be done the find matching tutors. There are two types of searching: basic search and specific search. For basic search, the user first selects a university, then a subject and finally a lecture. This will list all tutors offering their help regarding the selections. The specific search mode extends basic search with grades and available time slots selections.

If a student finds a tutor he wants to get in touch with, he contacts the tutor by sending a request. The tutor then can accept and pays a fee for the contact information (like an e-mail-address or a phone number). Payment can be a monthly fee (that starts from the first engagement) or just a single fee for every contact engagement. There is no other communication possible (like a built-in communication system).

Students who got in touch with tutors should have the possibility to rate them. Due to this rating system, students can choose more reliable between several tutors regarding their rating.

## References

Here are some links to existing portals (too generic ones) our customer mentioned.

* [https://tutor24.ch](https://tutor24.ch/)
* [http://www.nachhilfe-vermittlung.ch](http://www.nachhilfe-vermittlung.ch/)
* [http://owltutors.ch](http://owltutors.ch/)

# Overall description

# Diagram

# Use cases

List of Use cases:

On TRELLO!!! Added when finished!

1. **Create Profile (Student/Tutor)**
   1. **Actors**

Customer (Student/Tutor)

* 1. **Description**

As a Customer I want to create a new profile/account to use your services

* 1. **Trigger**

Click on the sign-up button

* 1. **Pre-conditions**

1. Customer has a valid email-address
2. Customer gives a not yet used Username
   1. **Post-conditions**
3. A new profile/account is created in the database with the given information
4. Validation email is sent to the given email-address
   1. **Main scenario**
5. Customer clicks on the sign-up button
6. Customer enters his information
7. System checks validity of email-address
8. System checks validity of Username
9. System sends a validation email to the given email-address
10. A new profile/account is created in the database with the customer's data
    1. **Alternative scenarios**
       1. Given email-address mail not valid
          1. System will prompt customer to enter a valid email-address
          2. User gives a valid email-address
          3. Use case resumes on step 4
       2. Given username already in use
          1. System will prompt customer to enter a different username, stating that the given username is already in use
          2. User gives a valid username
          3. Use case resumes on step 5
    2. **Special requirements**
    3. **Notes**
11. What is the maximum amount of characters used for username?
12. Are there any special conditions on the username, e.g. at least one number, or at least one capital letter?
13. **Search and Contact Tutor** 
    1. **Actors**

Customer (Student)

* 1. **Description**

As a Customer I want to find a tutor in a specific course at a specific university in a specific subject.

* 1. **Trigger**

Customer clicks on the search box.

* 1. **Pre-conditions**

1. Customer has a validated account.

2. Customer must be signed up.

* 1. **Post-conditions**

1. Customer receives Mail with confirmation of his request.

2. Tutor gets a request from the Student

* 1. **Main scenario**

1. The Customer chooses his university

2. The customer chooses his subject

3. The customer chooses his course

4. The customer clicks the search button

5. System prompts customer to choose one tutor out of a list of all possible tutors

6. User chooses a tutor and presses “new request”

7. System opens blackbox for request message

8. The customer enters his request message

9. the customer presses “send request”.

* 1. **Alternative scenarios**
  2. Customer chooses either one, tow or none of university, subject and course
  3. Customer presses Search button
  4. System prompts request of missing information.
  5. Scenario continues at step 3.6.1
  6. **Special requirements**

Customer should not be able to search classes he teaches as well.

* 1. **Notes**

1. **Login**
   1. **Actors**

Customer (Student/Tutor)

* 1. **Description**

As a Customer I want to log in to the page.

* 1. **Trigger**

Customer clicks on login.

* 1. **Pre-conditions**

1. Customer has an account.

* 1. **Post-conditions**

1. Customer is the Home-Page.

* 1. **Main scenario**
     + 1. System requests User – Information
       2. Customer enters user-name/e-mail
       3. Customer enters user-password (\*\*\*\*\*\*)
       4. Customer presses login button
       5. System verifies valid user-account
       6. System loads Personal-Home-Page
  2. **Alternative scenarios**
  3. System requests User-Information
  4. Customer enters e-Mail/Username and no password or reversed
  5. Customer presses login button
  6. System requests valid account-information
  7. Continue at step 4.6.1

2.1 System requests User-Information

2.2 User enters (wrong) e-mail/Username

2.3 User enters (wrong) password

2.4 User presses login button

2.5 System denies access and requests valid account-info

3.1 System requests User – Information

3.2 Customer enters user-name/e-mail

3.3 Customer enters user-password (\*\*\*\*\*\*)

3.4 Customer presses login button

3.5 System checks account and requests validation

3.6 Customer enters validation-code

3.7 System validates account

3.8 System loads Welcome-Page

* 1. **Special requirements**
  2. **Notes**

1. **Tutor handles Student request**
   1. **Actors**

Customer (Student) - Student

Customer (Tutor) - Tutor

* 1. **Description**

A Tutor receives a request from a student and accepts or rejects the request.

* 1. **Trigger**

Student send request to tutor.

* 1. **Pre-conditions**

1. Both Customers have a validated account.

2. Tutor must be signed up

* 1. **Post-conditions**

1. Tutor must pay fee to see contact information

2. Student gets informed that tutor accepted him with contact info

3. Tutor gets contact information after valid payment

* 1. **Main scenario**

Tutor gets request on Home-Page.

Tutor presses on the new request.

System loads the new request with message an anonym info

Tutor either presses “accept”

System initiates payment

System deletes requests and notifies Student

* 1. **Alternative scenarios**

1. Tutor gets request on Home-Page.

2. Tutor presses on the new request.

3. System loads the new request with message an anonym info

4. Tutor either presses “reject”

5. System deletes request and notifies Student by email

* 1. **Special requirements**
  2. **Notes**

***EXAMPLE***

1. **Withdraw Cash** (Enter a short name for the Use Case using an active verb phrase)
   1. **Actors**

Customer

[An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor that will be initiating this use case (primary) and any other actors who will participate in completing the use case (secondary).]

* 1. **Description**

As a customer I want to withdraw money from my account.

[Provide a brief description of the reason for and outcome of this use case.

Format: As a [user role] I want to [goal] (so I can [reason])]

* 1. **Trigger**

Customer inserts ATM card.

[Identify the event that initiates the use case. This could be an external business event or system event that causes the use case to begin, or it could be the first step in the normal flow.]

* 1. **Pre-conditions**
     1. Customer has active deposit account with ATM privileges
     2. Customer has an activated ATM card

[List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each pre-condition.]

* 1. **Post-conditions**

1. Customer receives cash
2. Customer account balance is reduced by the amount of the withdrawal and transaction fees

[Describe the state of the system at the conclusion of the use case execution. Should include both *minimal guarantees* (what must happen even if the actor’s goal is not achieved) and the *success guarantees* (what happens when the actor’s goal is achieved. Number each post-condition.]

* 1. **Main Scenario**

1. Customer inserts ATM card
2. Customer enters PIN
3. System prompts customer to enter language performance English or Spanish
4. System validates if customer is in the bank network
5. System prompts user to select transaction type
6. Customer selects Withdrawal From Checking
7. System prompts user to enter withdrawal amount
8. …
9. System ejects ATM card

[Provide a detailed description of the user actions and system responses that will take place during execution of the use case under **normal, expected** conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and description.]

* 1. **Alternative Scenarios**

4a. Customer is not in the bank network

1. System will prompt customer to accept network fee
2. Customer accepts
3. Use Case resumes on step 5

4b. Customer is not in the bank network

1. System will prompt customer to accept network fee
2. Customer declines
3. Transaction is terminated
4. Use Case resumes on step 9 of normal flow

[Document branches from the main flow to handle special conditions (also known as extensions). For each alternative flow reference the branching step number of the normal flow and the condition which must be true in order for this extension to be executed.]

* 1. **Special Requirements**

User validation (step 4) cannot take more then 30 seconds.

[Identify any additional requirements, such as nonfunctional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.]

* 1. **Notes**
     1. What is the maximum size of the PIN that a use can have?

[List any additional comments about this use case or any remaining open issues or TBDs (To Be Determined) that must be resolved.]



# Specific requirements

*(define all the* [*functionalities*](http://en.wikipedia.org/wiki/Functional_requirement) *that your application needs to fulfil the scenarios described in section 2)*

## Functional requirements

1.) The web portal allows to all the users to create a profile (sign up) by just indicating the **name**, an **e-mail-address**, a **password** and a **username**. The only public viewable information of this (basic) profile is the username.

2.) If a user is once sign up, the web portal allows him to login with his **username** or **e-mail-address** and his **password**. If the login-data are correct and he clicks on the login button, he will be automatically gets the view of his profile.

3.) The users can **extend his profile** with optional information like profile **picture**, short **biography**, **grades**, current **semester** and available **time slots**.

4.) A student can start a **basic search** for a tutor where he can chose with the first dropdown-field the **university**, then with the second one the **subject** and on the third one the **lecture**. Once chosen the student can click on the “search” button and the web portal will check on the databases for all the **matching tutors**.

5.) With a **specific search function** the student can although search for minimum **grades** and open **time slots**.

6.) To **protect** the **privacy** of the tutors there are only this two search options (basic and specific search). A user can’t search for real names or for an e-mail-address.

7.) As a **search result** the student gets a list of all the **matching tutors**, with:

- the **usernames** (directly linked to the tutors profiles),

- the grads of the tutors (if indicated) and

- for every tutor a “**new request**”-button

or he gets an **error message**, “that no tutor is available for the lecture he searched”.

8.) If at least one tutor is found, the student can click on the “**new request”**-button behind every tutor on the listand our portal sends automatically a **request-message** to the chosen tutor.

9.) The tutor gets this request-message on his own profile and can then click the “**accept**”- or the “**reject**”-button.

10.) If the tutor pays the monthly fee he can click the “accept”-button without any other costs, if he pays for every contact separately he got to pay new the few. As a result the tutor gets the contact information of the students or he gets the confirmation that he didn’t accept the request of this student.

## Non-functional requirements

(external, performance, etc.)

*Product requirements:*

1.) The users got to have **access** on the web portal **from different platforms**. (Windows, Mac, Smartphones, Tablets,…)

2.) The **data** are always **on the server**, the users can only access them with a server request for the web portal.

*Company requirements:*

3.) The development of the web portal will be organized and executed on the platforms **Trello** and **GitHub**. The developers and the clients got to have access to these sources at any time.

4.) The progress of the development process and the state of the requirements will be checked every week in **a scrum standup meeting** or in a more specific meeting. This protects us form doing unnecessary work and guarantees that the developers and the customers are on the same page.

*Safety requirements*

5.) The developers and the clients have to **protect the data** of the users and both are doing this to the best of their knowledge and belief.