Course Information Management System

Group 2

David Chang
Linda Chowdhury
Oscar Fitinghoff
Patrik Parberg
Tomas Hansson

Preface

Expected Readership of the Requirements Document

This document is expected to be read by the clients (universities), project supervisor and project team members, where the project team members will be responsible for system architecture and development.

Version History

Version	**Comments (reason for/summary of changes)**	**Date**	**Author(s)**
1.0	First version.	2008-01-02	David Chang, Linda Chowdhury, Oscar Fitinghoff, Patrik Parberg, Tomas Hansson

Contents

Introduction

Who Are the Users and What Problem Does the System Solve for Them?

The system will mainly solve two problems – the problem of getting easy access to and an overview of course information for students and the problem of easily creating and updating a course website for teachers with limited skills in web design.

Course information about schedule, lectures, results and course descriptions for university courses today are scattered over several different course websites, making it a time consuming task to check for course news or get an overview of what's currently happening in all the courses. We would like to solve this problem by creating a web application that runs course websites, that enables student to get easy access to information from all the different courses he or she is registered for and course leaders an effective and easy way to maintain the course website.

The main users will be university students, who are high school graduates, able to read English and able operate websites without any difficulty. Except for this we expect the students to be a fairly diverse group of people. The students are expected to use a range of different operating systems using the Mozilla Firefox web browser. For the students the system will mainly solve the problem of getting an easy overview of the courses the students are currently registered for, and providing a single interface for all course websites helping quick navigation. Other than this the system will also provide necessary functionality for the students, such as schedule and general course information.

Other users will be course leaders and course assistants at universities. These two groups of users are assumed to be able to read and understand English. They will not always be as used to navigating websites as the students normally are. Generally they are also older than the students. Also these two groups are expected to be diverse, for example using different operating systems. The problem for these two groups is that the creation and management of a course website require skills in web design for a good result, and that updating the website can be a time consuming and inconvenient task. The system will provide functions for creating course websites, relieving the need for web design skills, and simplify the process of updating the website. The course leaders and assistants are not expected to have any experience or skills in web design.

Other than the above mentioned user groups the system will be used by a system administrator that takes care of administration of user accounts (adding, removing and assigning privileges) and courses (adding, removing and editing).

The Main Uses of the System

For the students the system will provide functions for getting a quick overview of the courses they're currently taking, for example by functions for displaying course news from all the courses the student is attending, Really Simple Syndication (RSS) feeds for all course news, a compilation of all courses' deadlines and schedules and by offering a uniform layout of the course websites. The following is a usage narrative for a student:

> Simon is a 20 year old Economics major who is on his lunch break after a morning of lectures. After finishing his meal he takes up his laptop to check the latest news from the web via the university's wireless connection. While checking his normal RSS feeds in Mozilla Firefox he discovers that news has been posted for one of the courses he is registered for. This afternoon's lecture has been cancelled. Simon checks his compiled schedule for all the courses he is registered for and finds that he can attend a lecture in another course instead of the one that has been cancelled.
>
> In the news he also discovers that the course leader in his Economics course has entered the results for the latest home assignment. Simon goes to the front page of the system, logs in, checks his latest results and discovers a pleasing grade A for the assignment. Thereafter he logs out from the system to end the session.

The main usage for the course leaders and assistants is to administrate the course website. The course leader can also easily add functionality common to all (or some) course websites, for example functionality for viewing and exporting schedule.

> Eric is 53 years old and a new member of the Digital Electronics institution. He has just been assigned course leader of his first course in Digital Design. He has planned the course in detail and is about to create the course website from his computer in his office.
>
> He logs in to the system and easily creates a new website in a couple of easy and intuitive steps using a guide. The guide queries him for what functionality he would like to have on the website and helps him add content. After finishing the guide the system sets up a website with the functions Eric specified. The course website for Digital Design now contains functions for publishing course news and a calendar for the deadlines in the course, as well as general course information.
>
> The course news will be automatically displayed on the registered students' personal page as well as in their RSS feeds, enabling Eric to communicate quickly with many of his students. The course's schedule will be included in the compiled schedule for the students. Eric thereafter logs out from the system to end the session.

The Context/Environment in Which the System is to be Used

By course leaders and assistants the system will be used mostly at a stationary computer connected to the Internet, for example in the course leader's office, at home or in one of the computer labs. The course leaders and assistants will be spending more time in the system than the average student.

For students the usage will be more spontaneous. They'll be using the system from laptops connected to the Internet via the universities' wireless connections or from laptops or stationary computers at home.

The course leaders, course assistants and students are expected to use the system via Mozilla Firefox 2.0 or later versions, running on any operating system that supports this web browser.

The Scope of the System

Topic	In	Out
User authorization	X	
Create course website	X	
Add, display, edit and remove course description	X	
Add, display, edit and remove course news	X	
Add, display, edit and remove schedule	X	
Import schedule from iCalendar format for teachers	X	
Export schedule to iCalendar format	X	
Laboratory registration		X
Exam registration		X
Result registration	X	
Display results	X	
RSS feed for news	X	
Registration for course	X	
Course evaluation		X
Uploading of files for teachers	X	
Uploading of files for students' hand-ins		X
Overviews of deadlines for courses	X	
Access to schedule via Wireless Application Protocol (WAP)		X
Course website statistics		X
Search functionality for courses	X	
Search functionality for content on course websites		X
Forum		X
Course selection		X
Integration with Kungliga Tekniska Högskolan's (KTH) systems		X
Different designs and layouts for course websites		X
Support for different languages for the interface		X

Main Factors When Designing and Building the System

One of the problems the system aims to solve is to provide an easy and fast user interface to quickly get an overview of course news and schedule. To develop a user interface that provides this information easily will be one of the main challenges for this project.

Building an administration system that can be easily used even by a person without any knowledge in web design will also be a significant problem. We will have to take extra care when designing this part of the system.

The users shouldn't be able to view all data in the system, so we have to implement an authentication system to restrict users. The system also has to keep track of different users' privileges.

For result registration the system has to support different grade systems and point systems. The major grade systems should be supported, as well as most customizations that may be used in individual courses. Supporting all customizations might prove to be very complex, and some calculations may have to be done externally from the system.

One problem for the result registration would also be to create a user interface that enables the teachers to effectively assign results. To accomplish this, the user interface should be designed in such a way, that the teacher quickly can navigate through fields only by using the keyboard. The system should allow for batch processing for students to limit the number of user actions (for example allowing the user to select all users for a certain grade and then assigning the grade to them all at once). Page loading has to be at a minimum (Asyncronous JavaScript and XML (AJAX) should be used as much as possible for this) and page loads that cannot be avoided need to be quick.

Technologies and Risks

For the website we plan on using JavaServer Pages (JSP) backed with MySQL as a database. For the web pages we will use Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS). We also plan on using the JavaScript library Scriptaculous for AJAX features and user interface programming. For exportation and importation of schedules we'll be using the iCalendar format.

For easy access to course news headlines we will be using RSS, which is a rather simple technology that we don't find any major risks associated to. Exporting data to the formats iCalendar are also considered to be relatively simple and risk-free.

The main risk lies in the lack of experience using JSP within the group. All the group members have experience using Java in general, but none have previously worked with JSP. The group has also found that the number of examples for JSP on the web is not as many as for example that for PHP: Hypertext Preprocessor (PHP) and Active Server Pages .NET (ASP.NET). To limit this risk we plan on studying JSP early on in the process.

The group feels confident using databases in general and MySQL in particular and is not expecting any major problems surrounding that area. Even so, all group members have been assigned to refresh their knowledge in MySQL.

The group lacks a person with solid skills in design, and creating a good looking user interface might prove to be a challenge. To overcome this risk we will try to create a simple user interface, and separate it clearly from the logic implementation so that it may be developed externally later should the need arise.

User interface programming in web browsers is not very well standardized and might be a cause for problems, especially since the group members are not very experienced in that area. Using the Scriptaculous library will certainly simplify the user interface programming.

Glossary

A

Asyncronous JavaScript and XML (AJAX)
Refers to several web development techniques that enable the client and server to communicate in an asynchronous manner, making it possible for parts of the web pages to be dynamically updated without having to reload the entire web page.

Apache Tomcat
A web container that provides an environment for Java code to run in cooperation with a web server.

Active Server Pages .NET (ASP.NET)
A server-side scripting language developed by Microsoft that provides similar functionality to that provided by JSP.

C

Client-server architecture
An architectural model for distributed systems where the system functionality is offered as a set of services provided by a server. These are accessed by client computers that make use of the services.

Cascading Style Sheets (CSS)
A formatting language that enables web developer to separate styling from the layout of the web pages, as well as enabling styling options not available otherwise.

Central Processing Unit (CPU)
The unit in the computer that manage and control fetching, decoding and execution of instructions, communication with memory units etc.

D

Database
A computer database is a structured collection of data that is stored in a computer system so that a computer program or person using a query language can consult it to answer queries.

Downtime
Refers to a period of time or a percentage of a timespan that a system is unavailable or offline.

Dynamic web pages
A web page that can change according to visitors' selections or interactions, which is achieved via server-based web programming languages such as ASP.NET or PHP.

E

Entity-relationship model (ER model)
An Entity-relationship model is a relational schema database modeling method used to model a system and its requirements, where an entity represents a discrete object and a relationship captures how two or more entities are related to one another.

H

Hypertext Markup Language (HTML)
A authoring and editing language used to create web pages on the World Wide Web.

Hypertext Transfer Protocol (HTTP)
A protocol for communication that's used between web servers and web browsers.

I

iCalendar
A standard for group calendaring and scheduling, which enables calendaring data to be sent via e-mail or the Web that is automatically entered into the recipient's schedule.

Information Page
A page belonging to the course website where the course leader can choose to present any textual information he or she wishes.

J

JavaServer Pages (JSP)
Enables software developers to create dynamic web pages that run on the web server. The web pages can for example load and process data from a database server, as well as receive user input from end-users.

M

Mozilla Firefox
Mozilla Firefox is a web browser

MySQL
A widely used open source database relation management system.

O

Object oriented
Generally used to describe a system that deals primarily with different types of objects, and where the actions you can take depend on what type of object you are manipulating.

P

Personal Page
Gathers information and presents it on one page, enabling the user to quickly get an overview of the information and providing functions related to the information.

PHP Hypertext Preprocessor (PHP)
A server-side, HTML embedded scripting language used to create dynamic Web pages.

Plug-in
A hardware or software module that adds a specific feature or service to a larger system. The idea is that the new component simply *plugs in* to the existing system.

R

Really Simple Syndication (RSS) feeds
Enable the users to easily stay updated with news on many websites simultaneously using special software or integrated functions in web browsers such as Mozilla FireFox or Internet Explorer 7.

S

Scriptaculous
A JavaScript library that is run on the client. The library is designed to be compatible with most common web browsers and provides functions for AJAX and visual effects.

Single core Central Processing Unit (CPU)
A CPU with one core, as opposed to CPU's with two cores.

T

Thin-client
A thin-client is a client computer or client software in client-server architecture networks which depends primarily on the central server for processing activities, and mainly focuses on conveying input and output between the user and the remote server.

Three-tier architecture
An architecture that is a variant of client-server architecture, which use multiple servers.

W

WAP
An international standard for wireless communications, which is frequently used by mobile phones to access the Internet.

Web browser
A software application used to locate and display web pages.

User Requirements Definition

Functional Requirements

1. Authentication System

1.1. Course leaders, course assistant, students and system administrators shall be able to identify themselves by logging in and authenticating themselves.

Rationale: Parts of the system shall be unavailable to unauthorized users.
Test: A user shall not be authenticated when entering a username that doesn't exist, or when entering a password that doesn't match the entered username. As a base for this test use case UC1 (Authenticate to the System) will be used.

2. Personal Pages

2.1. Authenticated course leaders, course assistants, students and system administrators shall have a personal page where course news, scheduled activities, results and deadlines relevant to them are available.

Rationale: The personal page gathers information and presents it on one page, enabling the user to quickly get an overview of the information and providing functions related to the information.
Test: It shall be possible for an authenticated user to view course news, scheduled activities, results and deadlines on his or her personal page. As a base for this test use case UC1 (Authenticate to the System) will be used.

2.2. The schedule, deadlines and course news are available to anyone visiting a user's personal page without logging in.

Rationale: The user wants to be able to quickly access information on the personal pages, and therefore wants to skip the hassle of logging in.
Test: It shall be possible for any user (logged in or not logged in) to view an overview of deadlines, scheduled activities and course news on the personal page for any user. As a base for this test use cases UC3 (View Overview of Course News), UC21 (View Scheduled Activity from Compiled Schedule) and UC27 (View Overview of Deadlines) will be used.

2.3. Students shall be able to get an overview of course news for all the courses they are registered for.

Rationale: Providing an overview removes the need for students to continuously visit all the different course websites to read the latest course news.
Test: It shall be possible for any user (logged in or not logged in) to view an overview of course news for a specific user. As a base for the test use case UC3 (View Overview of Course News) will be used.

3. Creation Guide for Course Website

3.1. There shall be a creation guide for course leaders that will guide them through the process of creating a course website. The guide queries the user for information for a course website – namely a course description, course schedule, information pages and deadlines – and then creates the website.

Rationale: The course website shall be possible to create without knowledge of web design.
Test: When authenticated as a course leader there shall be a guide available to guide the user when creating a new course website, allowing the user to add a course description, course schedule, information pages and deadlines to the course website. As a base for the test use case UC4 (Create Course Website) will be used.

4. Course Description

4.1. Course leaders shall be able to add and edit the course description for courses they are responsible for.

Rationale: The course description need to be up-to-date, and the course leader is the one who is best aware of what the course description should be.
Test: When authenticated as a course leader it shall be possible to add and edit the course description. As a base for the test use cases UC4 (Create Course Website) and UC5 (Edit Existing Course Description) will be used.

4.2. Any user shall be able to view the course description.

Rationale: The course description provides useful information to people involved in the course.
Test: It shall be possible for any user (logged in or not logged in) to view the course description. As a base for this test use case UC6 (View Course Description) will be used.

5. Course News

5.1. Course leaders shall be able to add, edit and remove course news for courses they are responsible for.

Rationale: Course leaders are interested in giving accurate and up-to-date information regarding the course.
Test: When authenticated as a course leader it shall be possible to add, edit and remove course news for the courses that the user has been assigned course leader for. As a base for the test use cases UC7 (Add Course News), UC8 (Edit Existing Course News) and UC9 (Remove Existing Course News) will be used.

5.2. Any user shall be able to view course news for specific courses.

Rationale: Course news provides current information to people involved in the course.
Test: It shall be possible for any user (logged in or not logged in) to view course news on the course websites. As a base for this test use case UC10 (View Course News) will be used.

6. Information Pages

6.1. Course leaders shall be able to add, edit and remove information pages for courses they are responsible for. An information page is a page belonging to the course website where the course leader can choose to present any textual information he or she wishes.

Rationale: Course leaders want to be able to publish content that is necessary for students throughout the duration of the course.
Test: When authenticated as a course leader it shall be possible to add, edit and remove information pages for the courses that the user has been assigned course leader for. As a base for the test use cases UC11 (Add Information Page), UC12 (Edit Existing Information Page) and UC13 (Remove Existing Information Page) will be used.

6.2. Any user shall be able to view information pages for courses.

Rationale: There is a need for pages with information that is necessary throughout the duration of a course. Information pages fulfil this need.
Test: It shall be possible for any user (logged in or not logged in) to view information pages. As a base for this test use case UC14 (View Information Page) will be used.

7. Schedule

7.1. Course leaders shall be able to add a schedule by importing a file in iCalendar format and remove an existing schedule for courses they are responsible for.

Rationale: The course leader might prefer to use a desktop calendar application when creating a schedule. Many popular desktop calendar applications can export calendar entries to the iCalendar format, and for such cases it would be convenient to provide functionality for importing schedules in the iCalendar format.
Test: An iCalendar file exported from the Google Calendar application shall be imported into a course website. Start and end times for activities as well as descriptions shall be checked for consistency between the calendar exported and the schedule that exists in the system after the iCalendar file was imported. As a base for this test use cases UC15 (Import Course Schedule) and UC16 (Remove Existing Course Schedule) will be used.

7.2. Course leaders shall be able to add, edit and remove scheduled activities for courses they are responsible for.

Rationale: Schedules can and will change and therefore the schedule need to be kept up-to-date.
Test: When authenticated as a course leader it shall be possible to add, edit and remove scheduled activities. As a base for the test use cases UC17 (Add Scheduled Activity), UC18 (Edit Existing Scheduled Activity) and UC19 (Remove Existing Scheduled Activity) will be used.

7.3. Any user shall be able to view the schedule for a course and details for a scheduled activity.

Rationale: The schedule provides useful information about the activities in the course to people involved in the course.
Test: It shall be possible for any user (logged in or not logged in) to view the course schedule. As a base for the test use case UC20 (View Scheduled Activity from Course Schedule) will be used.

7.4. It shall be possible to view a compiled schedule for a student, containing activities for all the courses that the student is registered for, and details for a scheduled activity.

Rationale: The student doesn't want to have to look at the different course schedules individually to get an overview of his or her schedule. The system will solve this problem by providing a compiled schedule for the student's courses.
Test: It shall be possible for any user (logged in or not logged in) to view the compiled schedule for another student. As a base for the test use case UC21 (View Scheduled Activity from Compiled Schedule) will be used.

7.5. Course leaders, course assistants and students shall be able to export the schedule to iCalendar format.

Rationale: An Internet connection is not always available to the user; therefore some users prefer to have a local copy of the schedule available.
Test: It shall be possible for any user (logged in or not logged in) to export the schedule to iCalendar format. As a base for the test use case UC22 (Export Schedule in iCalendar Format) will be used.

8. Deadlines

8.1. Course leaders shall be able to add, edit and remove deadlines for activities in courses that they are responsible for.

Rationale: Deadlines are useful to assist the course leader in distributing the course work evenly over the course and communicating with the students what work is needed.
Test: When authenticated as a course leader it shall be possible to add, edit and remove deadlines. As a base for the test use cases UC23 (Add Deadline), UC24 (Edit Existing Deadline) and UC25 (Remove Existing Deadline) will be used.

8.2. Any user shall be able to view descriptions for deadlines.

Rationale: The deadlines provide useful information about the workload in the course to the people involved in the course.
Test: It shall be possible for any user (logged in or not logged in) to view a description for a deadline in a course. As a base for this test use case UC26 (View Deadlines) will be used.

8.3. Students shall be able to get an overview of deadlines for the courses they are registered for.

Rationale: An overview of all the relevant deadlines will help the students to prioritize and plan their work.

Test: It shall be possible for any user (logged in or not logged in) to view an overview of deadlines for a specific user. As a base for the test use case UC27 (View Overview of Deadlines) will be used.

9. Uploaded Files

9.1. Course leaders shall be able to upload, edit and remove files for courses they are responsible for.

Rationale: Course leaders are interested in providing files containing relevant information for the course.
Test: When authenticated as a course leader it shall be possible to upload, edit and remove files for the courses that the user has been assigned course leader for. As a base for the test use cases UC28 (Upload File), UC29 (Edit Existing File) and UC30 (Remove Existing File) will be used.

9.2. Course leaders, course assistants and students who are registered for the course shall be able to download uploaded files.

Rationale: In order to keep the bandwidth (and possibly printing) costs down only users to whom the files are relevant can download them.
Test: When authenticated as a course leader, course assistant or student it shall be possible to view the files for the courses that the user has been assigned to or registered for. As a base for the test use case UC31 (Download File) will be used.

10. Course Assignments

10.1. Course leaders shall be able to add, edit and remove course assignments for courses they are responsible for.

Rationale: Results for students are assigned per course assignment. Assignments need to be added, edited and removed to reflect the graded assignments in the course so that results can be registered.
Test: When authenticated as a course leader it shall be possible to add, edit and remove course assignments for the courses that the user has been assigned course leader for. As a base for the test use cases UC32 (Add Course Assignment), UC33 (Edit Existing Course Assignment) and UC34 (Remove Existing Course Assignment) will be used.

11. Results

11.1. Course leaders and course assistant shall be able to register results (including modifying previously registered results) for courses they are responsible for.

Rationale: Results need to be registered in the system so students can view their results.
Test: When authenticated as a course leader or course assistant it shall be possible to register results for students for the courses that the user is responsible for. As a base for the test use case UC35 (Register Results) will be used.

11.2. Students shall be able to view results for courses they are registered for.

Rationale: Students want to view results to see how they are doing in the course.

Test: When authenticated a user shall be able to view his or her own results. As a base for the test use case UC36 (View Results) will be used.

12. Course Registration

12.1. Course leaders shall be able to view registered students for the courses they are responsible for.

Rationale: This allows the course leader to tell whether a student is registered or not.
Test: When authenticated as a course leader it shall be possible to view registered students for the courses that the user has been assigned course leader for. As a base for this test use case UC37 (View Registered Students) will be used.

12.2. Course leaders are required to accept students' application after they have applied for a course, before the students are registered for the specified course.

Rationale: Only students accepted by the course leader should get registered to the course.
Test: When authenticated as a course leader it shall be possible to accept student's applications to join the course for the courses that the user has been assigned course leader for. An accepted student shall be registered for the course. As a base for the test use case UC38 (Confirm Application to Get Registered for Course) will be used.

12.3. Students shall be able to apply for courses (after which the course leader has to accept them before they are registered for the course).

Rationale: Students need to apply to get registered for the course so that course assistants and course leader can enter results, and to enable the course leader to allocate appropriate resources depending on how many students are taking the course. By applying for the course, the students also confirm that they want to take the course.
Test: When authenticated as a student it shall be possible to apply for a course. As a base for the test use case UC39 (Apply for Course) will be used.

12.4. Course leaders shall be able to unregister registered students.

Rationale: Students that drop out or are registered by mistake should not stay registered for the course.
Test: When authenticated as a course leader it shall be possible to unregister registered students for the courses that the user has been assigned course leader for. As a base for the test use case UC40 (Unregister Registered Student) will be used.

13. Miscellaneous

13.1. System administrators shall be able to add and remove course leaders', course assistants' and system administrators' privileges. All users have student privileges (they can for example apply for a course).

Rationale: Only certain users should have privileges to use functions restricted to course leaders and course assistants. Only system administrators are entrusted with the authority to add and remove these privileges.
Test: When authenticated as a system administrator it shall be possible to add and remove user privileges. As a base for the test use case UC44 (Edit Existing User Privileges) will be used.

13.2. Course leaders shall be able to add and remove course assistants' privileges for courses they are responsible for.

Rationale: Course leaders want to be able to add and remove privileges for course assistants in their courses without having to involve the system administrator, as adding the course assistants themselves is likely to be less time consuming than contacting the system administrator. Furthermore, in some courses new course assistants need to be added with short notice, and enabling the course leader to add privileges for this speeds up the process.
Test: When authenticated as a course leader it shall be possible to add and remove privileges for course assistants in the courses that the course leader is responsible for. As a base for the test use cases UC45 (Add Course Assistant) and UC46 (Remove Existing Course Assistant) will be used.

13.3. System administrators shall be able to add and edit courses.

Rationale: A course needs to be added before a course leader can be assigned and before the course leader can create a website.
Test: When authenticated as a system administrator it shall be possible to create a course. As a base for the test use cases UC47 (Add Course) and UC48 (Edit Existing Course) will be used.

13.4. System administrators shall be able to add and remove user accounts, and edit user account's password.

Rationale: User accounts need to be added and removed to reflect the changing group of people that need to use the system.
Test: When authenticated as a system administrator it shall be possible to add and remove user accounts, and edit their passwords. As a base for the test use cases UC41 (Add User Account), UC42 (Edit User Account Password) and UC43 (Remove User Account) will be used.

Non-functional Requirements

14. Product Requirements

14.1. Personal Page Availability

Description: The personal pages shall be available.
Measure: The personal page shall be requested once a minute during 24 hours. The personal page shall belong to a user registered to four courses with ten course news each and a total for all four courses of 10 scheduled activities for the current week.
Goal: 1380 responses (60 missed requests).

14.2. Personal Page Performance

Description: How much time it takes for the personal pages to be generated (not including the time required to transfer the data to the client).
Measure: The average time it takes for a personal page to be generated over 100 requests when there are 10 simultaneous requests for a personal page. The personal page shall belong to a user registered to four courses with ten course news each and a total for all four courses of 10 scheduled activities for the current week.
Goal: 5 seconds

14.3. Personal Page Scalability

Description: The ability to cope with concurrent requests for the personal page of a student.
Measure: The factor by which page generation time increase when the number of concurrent requests increases from 10 to 100. The personal page shall belong to a user registered to four courses with ten course news each and a total for all four courses of 10 scheduled activities for the current week.
Goal: 3

14.4. Language

Description: The language used in the system shall be English.
Measure: -
Goal: -

15. External Requirements

15.1. Security

Description: If a user without privileges for adding, editing or removing course news for a course in which they are registered attempts to do so, the system shall inform the user that he or she does not have the authority to perform the action in question.
Measure: When authenticated as a user without privileges to add, edit or remove course news the user tries to add, edit or remove course news. The scope of this test will only be to access course news administration for one course website, as either logged in or logged out.
Goal: The user shall not be able to access the course news administration.

15.2. Privacy

Description: If a user tries to view another student's results, the system shall inform the user that he or she does not have the authority to perform the action in question.
Measure: Trying to access another student's result page directly (being either logged in or not). The scope of this test will only be to access one other student's page with results, as either logged in or logged out.
Goal: The user shall not be able to view the other student's results.

Use Cases

The following extension applies to all use cases, but is only presented here to reduce clutter.

*a. At any time, the system fails.
1. The user tries again.
   1b. The system is not available.
      1. The user tries again another time or contacts the system administrator.

Use Case UC1: Authenticate to the System

Primary Actor: Course leader, course assistant, student and system administrator.
Stakeholders and Interests:

- *Course leaders:* Want to make sure that only course leaders can add, edit and remove content on the course websites.
- *Course assistants:* Want to be able to register results for the courses they are assigned to.
- *Students:* Want to be certain that they can view their results, and that other students can not view their results.
- *System administrators:* Want to make sure that unauthorized users can't change or access certain information in the system.
- *University:* Has interests in much the same as the course leaders – making sure that unauthorized persons can't add inappropriate content.

Trigger: The user selects to log in.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to the system.
Minimal Guarantee: The user is either logged in or logged out.
Success Guarantee: The user is authenticated and given his or her user privileges.

Main Success Scenario

1 The system requests the user to input:
   - Username.
   - Password.

2. The user enters the username and password.
3. The system validates the user's login details.
4. The system creates a session with the user and stores data necessary for identification.

Extensions

3a. The username entered doesn't exist or the password entered doesn't match the username.
   1. The system notifies the user of the problem and allows the user to correct the problem.
   2. The user corrects the problem and resubmits the username and password.
   3. Continue from step 3.

Frequency of Occurrence: A couple of times per week and user.

Use Case UC2: View Personal Page

Primary Actor: Course leader, course assistant, student and system administrator.
Stakeholders and Interests:

- *Course leaders:* Want to have quick access to the administrative functions associated with the courses they are responsible for.
- *Course assistants:* Want to have quick access to the grading functions for the courses they are assigned as assistants for.
- *Students:* Want an overview of course related information such as news and schedule.
- *System administrators:* Want to have quick access to administrative functions for the system.

Trigger: The user navigates to the personal page they want to view.

Preconditions: The user is using a computer with a working Internet connection and has started the web browser Mozilla Firefox.
Minimal Guarantee: No information in the system has been modified.
Success Guarantee: The system displays the user's personal page.

Main Success Scenario

1. The system validates if the user is authenticated as the owner of the personal page.
2. The user is authenticated as the owner of the personal page.
3. The system displays the personal page, including restricted functions that require the user to be authenticated.

Extensions

2a. The user is not authenticated as the owner of the personal page.
  1. The system displays the personal page, only displaying public information.

Frequency of Occurrence: Daily on average.

Use Case UC3: View Overview of Course News

Primary Actor: Student.
Stakeholders and Interests:
- *Course leaders:* Want the students to be able stay up-to-date with news in the course.
- *Course assistants:* Want the students to be able stay up-to-date with news in the course.
- *Students:* Want to stay up-to-date with news in the course.
- *University:* Wants students to be able to stay up-to-date with activities in the course, minimizing the number of misunderstandings.

Trigger: The user selects to view course news.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox.
Minimal Guarantee: The course news has not been changed.
Success Guarantee: The system displays a list of course news from the courses the student is registered for.

Main Success Scenario

1. The system displays a list of course news for the courses that the user is registered for.

Frequency of Occurrence: Daily or almost daily for most students.

Use Case UC4: Create Course Website

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want a way to create a course website containing course description and the ability for students to apply for the course.
- *Course assistants:* Want a course website that has a course description and the ability for students to apply for the course.
- *Students:* Want a course website that has a course description and the ability for them to apply for the course.

- *University:* Wants a course website that satisfies the students' needs.

Trigger: The user selects to create a course website for a course.
Preconditions: The user is <u>authenticated</u> (UC1) and <u>assigned as the course leader</u> (UC44) for the course in question.
Minimal Guarantee: The course website is either fully saved or not saved at all.
Success Guarantee: A course website is created with course description and the ability for students to apply for the course, and a confirmation has been displayed. The course website is maintainable through the system.

Main Success Scenario

1. The system starts the creation guide for the course website.
2. The system requests the user to input:
   - A course name to be used to identify the course.
   - The course's credits.
   - The period and year when the course starts.
   - A textual description of the course.
3. The user inputs the requested information.
4. The system validates input.
5. The system requests the user to input whether he or she wishes to import a schedule.
6. The user declines.
7. The system requests the user to input whether he or she wishes to add an information page.
8. The user declines.
9. The system requests the user to input whether he or she wishes to add a deadline.
10. The user declines.
11. The system presents a summary of the information that the user has entered and requests confirmation.
12. User confirms the information entered.
13. The system creates the course website
14. The system displays a confirmation that the course website has been created.

Extensions

4a. The user input doesn't validate.
   1. The system notifies the user that the validation failed and allows the user to correct the problem.
   2. The user corrects the problem and resubmits.
   3. Continue from step 4.

6a. The user accepts to import a schedule.
   1. The system requests the user to input:
      - A file containing an iCalendar representation of the schedule to be imported.
   2. The user provides the schedule to import.
   3. The system imports the schedule.

      3a. The schedule provided was not in an iCalendar compatible format.
         1. The system notifies the user of the problem and allows the user to correct it.
         2. The user edits the input and resubmits.
         3. Continue from step 6a.3.

    4. The system displays a confirmation that the schedule was imported.
    5. Continue from step 7.

8a. The user accepts to add an information page.

1. The system requests the user to input:
   - Title for the information page.
   - Content for the information page.
2. The user inputs the title and content.
3. The system validates the input and displays a preview.

   3a. The title or content is missing.
   1. The system notifies the user of the problem and allows the user to correct it.
   2. The user edits the input and resubmits.
   3. Continue from step 8a.3.
4. The user accepts the preview.

   4a. The user is not satisfied with the preview and wants to make changes to the page.
   1. The user edits the input and resubmits.
   2. Continue from step 3.
5. The system stores the information page.
6. The system displays a confirmation that the information page was saved.
7. Continue from step 7.

10a. The user accepts to add a deadline.

1. The system requests the user to input:
   - A title for the deadline.
   - The date for the deadline.
   - A description for the deadline.
2. The user inputs a title, a date and a description.
3. The system validates the input.

   3a. The input doesn’t validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 10a.3.
4. The system displays a summary of the new deadline, and requests the user to confirm.
5. The user confirms the deadline.

   5a. The user doesn’t confirm the deadline.
   1. The system allows the user to edit the input.
   2. The user edits the input and resubmits.
   3. Continue from step 10a.3.
6. The system saves the deadline.
7. The system displays a confirmation that the new deadline was successfully saved.
8. Continue from step 9.

12a. The user doesn’t confirm the entered information.

1. The system allows the user to make changes to the entered information.
2. The user edits the input and resubmits.
3. Continue from step 5.

Frequency of Occurrence: One time for each course, but overall more often in between periods when new courses start.

Use Case UC5: Edit Existing Course Description

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to correct the course description.
- *Course assistants:* Want an accurate course description.
- *Students:* Want an accurate course description.

Trigger: The user selects to edit the course description for a specific course.
Preconditions: The user is authenticated (UC1) and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course description is either fully saved or not saved at all.
Success Guarantee: The changes are saved and the course description on the website is edited, and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - A course name to be used to identify the course.
   - The course's credits.
   - The period and year when the course starts.
   - A textual description of the course.
2. The user performs the desired changes to the course description.
3. The system validates the input.
4. The system saves the changes.
5. The system displays a confirmation that the changes have been saved.

Extensions

3a. The input doesn't validate.
   1. The system notifies the user of the problem and allows the user to correct it.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

Frequency of Occurrence: Once or a few times at the start of each course, then very rarely.

Use Case UC6: View Course Description

Primary Actor: Student.
Stakeholders and Interests:

- *Course leaders:* Want students to be able to get basic information for the course.
- *Course assistants:* Want students to be able to get basic information for the course.
- *Students:* Want to be able to get basic information for the course.
- *University:* Want students to be able to get basic information for the course.

Trigger: The user selects the course to view the course description for.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to the personal page.
Minimal Guarantee: The course description has not been changed.
Success Guarantee: The student views the course description.

Main Success Scenario

1. The system displays the course website for the selected course.
2. The user selects to display course description.
3. The system displays the course description.

Frequency of Occurrence: About one time per student at the beginning of each course.

Use Case UC7: Add Course News

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to be able to add course news.
- *Course assistants:* Want the course website to contain news regarding the course.
- *Students:* Want the course website to contain news regarding the course.
- *University:* Wants course websites to contain up-to-date information so that misunderstandings are kept at a minimum, and that students are satisfied with the information provided.

Trigger: The user selects to add course news.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course news is either fully saved or not saved at all.
Success Guarantee: The added course news is published on the course web site, and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Headline for course news.
   - Content for course news.
2. The user inputs the headline and content.
3. The system validates the input and displays a preview.
4. The user accepts the preview.
5. The system stores the course news.
6. The system displays a confirmation that the course news has been published.

Extensions

3a. The headline or content is missing.
  1. The system notifies the user of the problem and allows them to edit.
  2. The user edits the input and resubmits.
  3. Continue from step 3.

4a. The user is not satisfied with the preview and does not accept it.
  1. The user edits the input and resubmits.
  2. Continue from step 3.

Frequency of Occurrence: A couple of times per week for each course when the course is active.

Use Case UC8: Edit Existing Course News

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to edit existing course news in case changes need to be done.
- *Course assistants:* Want the course website to contain up-to-date information.
- *Students:* Want the course website to be updated frequently and contain up-to-date information.
- *University:* Wants course websites to contain up-to-date information so that misunderstandings are kept at a minimum, and that students are satisfied with the information provided.

Trigger: The user selects to edit existing course news.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course news or changes to course news are either fully saved or not saved at all.
Success Guarantee: The changes are saved and the course news on the website is updated, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing course news.
2. The user selects the course news to edit.
3. The system requests the user to input:
   - Headline for course news.
   - Content for course news.
4. The user inputs headline and content for course news.
5. The system validates the input and displays a preview.
6. The user accepts the preview.
7. The system stores the edited course news.
8. The system displays a confirmation that the course news was edited.

Extensions

5a. The input doesn’t validate.
   1. The system notifies the user of the problem and requests the user to correct it.
   2. The user edits the input and resubmits.
   3. Continue from step 5.

6a. The user is not satisfied with the preview and does not accept the preview.
   1. The user edits the input entered previously and resubmits.
   2. Continue from step 5.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC9: Remove Existing Course News

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to remove existing course news.
- *Course assistants:* Want the course website to contain up-to-date information.

- *Students:* Want the course website to be updated frequently and contain up-to-date information.
- *University:* Wants course websites to contain up-to-date information so that misunderstandings are kept at a minimum, and that students are satisfied with the information provided.

Trigger: The user selects to remove existing course news.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course news is either fully removed or not removed at all.
Success Guarantee: The course news is removed and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing course news.
2. The user selects the course news to remove.
3. The system displays the course news selected by the user and requests confirmation.
4. The user confirms the removal.
5. The system removes the course news.
6. The system displays a confirmation that the course news was removed.

Extensions

4a. The user doesn't want to remove the course news and cancels the removal.
   1. Continue from step 1.

Frequency of Occurrence: Very rarely throughout the duration of a course.

Use Case UC10: View Course News

Primary Actor: Student.
Stakeholders and Interests:
- *Course leaders:* Want the students to be able stay up-to-date with news in the course.
- *Course assistants:* Want the students to be able stay up-to-date with news in the course.
- *Students:* Want to stay up-to-date with news in the course.
- *University:* Wants students to be able to stay up-to-date with activities in the course, minimizing the number of misunderstandings.

Trigger: The user selects the course for which to view news.
Preconditions: The user is using a computer with a working Internet connection and has started the web browser Mozilla Firefox.
Minimal Guarantee: No course news has been changed.
Success Guarantee: The system displays a list of course news from the selected course.

Main Success Scenario

1. The system displays a list of course news for the selected course.

Frequency of Occurrence: Daily or almost daily for most students.

Use Case UC11: Add Information Page

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to add an information page for the course website.
- *Course assistants:* Want a course website with up-to-date information.
- *Students:* Want a course website with up-to-date information.
- *University:* Wants a course website with information that satisfies the students' needs.

Trigger: The user selects to add a new information page.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The information page is either fully saved or not saved at all.
Success Guarantee: The information page is created and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Title for the information page.
   - Content for the information page.
2. The user inputs the title and content.
3. The system validates the input and displays a preview.
4. The user accepts the preview.
5. The system stores the information page.
6. The system displays a confirmation that the information page was saved.

Extensions

3a. The title or content is missing.
1. The system notifies the user of the problem and allows the user to correct it.
2. The user edits the input and resubmits.
3. Continue from step 3.

4a. The user is not satisfied with the preview and wants to make changes to the page.
1. The user edits the input and resubmits.
2. Continue from step 3.

Frequency of Occurrence: Once or a few times at the start of each course, then quite rarely during the course.

Use Case UC12: Edit Existing Information Page

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want a course website with updated information pages.
- *Course assistants:* Want a course website with up-to-date information.
- *Students:* Want a course website with up-to-date information.
- *University:* Wants a course website with information that satisfies the students' needs.

Trigger: The user selects to edit an existing information page.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The information page is either fully saved or not saved at all.

Success Guarantee: The changes are saved and the information pages on the website are edited, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing information pages.
2. The user selects the information page to edit.
3. The system requests the user to input:
   - Title for the information page.
   - Content for the information page.
4. The user edits the information.
5. The system validates the input and displays a preview.
6. The user accepts the preview.
7. The system stores the updated page.
8. The system displays a confirmation that the information page was edited.

Extensions

5a. The input doesn't validate.
   1. The system notifies the user of the problem and allows the user to correct it.
   2. The user edits the input and resubmits
   3. Continue from step 5.

6a. The user is not satisfied with the preview and does not accept the preview.
   1. The user edits the input previously entered and resubmits.
   2. Continue from step 5.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC13: Remove Existing Information Page

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to remove information pages which are no longer needed or relevant.
- *Course assistants:* Want a course website with up-to-date information.
- *Students:* Want a course website with up-to-date information.
- *University:* Wants a course website with information that satisfies the students' needs.

Trigger: The user selects to remove an existing information page.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The information page is either fully removed or not removed at all.
Success Guarantee: The information page is removed and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing information pages.
2. The user selects the information page to remove.
3. The system displays the page selected by the user and requests confirmation.
4. The user confirms the removal.
5. The system removes the information page.

6. The system displays a confirmation that the information page was removed.

Extensions

4a. The user doesn't want to remove the information page and cancels the removal.
   1. Continue from step 1.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC14: View Information Page

Primary Actor: Course leader, course assistant and student.
Stakeholders and Interests:

- *Course leaders:* Want to be able to view the content on the information page that they have published. Also want the students and course assistants to be able to access necessary information throughout the course.
- *Course assistants:* Want to be able to view the information for the course that the course leader has published.
- *Students:* Want to be able to view the information for the course that the course leader has published.

Trigger: The user selects to view an information page.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to a specific course.
Minimal Guarantee: No information page has been changed.
Success Guarantee: The system displays the information page.

Main Success Scenario

1. The system displays the information page.

Frequency of Occurrence: Students will view the information page for courses more often than the course leader and the course assistants. It will be viewed quite often before and at the beginning of every course, and then rather frequently throughout the duration of the course.

Use Case UC15: Import Course Schedule

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to be able to add a schedule to the course website and to be able to edit the schedule if changes of the course activities are made.
- *Course assistants:* Want an accurate schedule so that they know when activities are held.
- *Students:* Want an accurate schedule so that they know when activities are held.

Trigger: The user selects to add a schedule.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The schedule is either successfully imported or not imported at all.
Success Guarantee: The schedule is available to any user and a confirmation has been displayed. The schedule is added to the compiled schedules of any students currently registered for the course.

Main Success Scenario

1. The system requests the user to input:
   - A file containing an iCalendar representation of the schedule to be imported.
2. The user provides the schedule to import.
3. The system imports the schedule.
4. The system displays a confirmation that the schedule was imported.

Extensions

3a. The schedule provided was not in an iCalendar compatible format.
   1. The system notifies the user of the problem and allows the user to correct it.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

Frequency of Occurrence: Once at the creation of each new course website, then very rarely.

Use Case UC16: Remove Existing Course Schedule

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to remove existing course schedules.
- *Course assistants:* Do not want an outdated or irrelevant schedule online.
- *Students:* Do not want an outdated or irrelevant schedule online.

Trigger: The user selects to remove an existing schedule.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The schedule is either fully removed or not removed at all.
Success Guarantee: The schedule is removed from the website and a confirmation has been displayed.

Main Success Scenario

1. The system requests a confirmation for the removal.
2. The user confirms the removal.
3. The system removes the schedule.
4. The system displays a confirmation that the schedule was removed.

Extensions

2a. The user doesn’t confirm the removal.
   1. System redirects the user to the course website.

Frequency of Occurrence: Rarely during a course and almost never otherwise.

Use Case UC17: Add Scheduled Activity

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to add a scheduled activity to the schedule.

- *Course assistants:* Want an accurate schedule so that they know when activities are held.
- *Students:* Want an accurate schedule so that they know when activities are held.

Trigger: The user selects to add a scheduled activity.
Preconditions: The user is <u>authenticated</u> (UC1), has selected a specific course and <u>assigned as the course leader</u> (UC44) for the course in question.
Minimal Guarantee: The scheduled activity is either fully saved or not saved at all.
Success Guarantee: The scheduled activity is added to the specific course schedule and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Title.
   - Description.
   - Date.
   - Time.
2. The user inputs values.
3. The system validates the input and displays a preview.
4. User accepts the preview.
5. The system saves the schedule.
6. The system displays a confirmation that the edit was successful.

Extensions

3a. The input doesn't validate.
   1. The system notifies the user of the problem and allows the user to correct the problem.
   2. User edits the input and resubmits.
   3. Continue from step 3.

4a. The user is not satisfied with the preview and does not accept the preview.
   1. The user edits the values and resubmits.
   2. Continue from step 3.

Frequency of Occurrence: Rarely during a course and almost never otherwise.

Use Case UC18: Edit Existing Scheduled Activity

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to edit an existing schedule if changes of the course activities are made.
- *Course assistants:* Want an accurate schedule so that they know when activities are held.
- *Students:* Want an accurate schedule so that they know when activities are held.

Trigger: The user selects to edit an existing scheduled activity.
Preconditions: The user is <u>authenticated</u> (UC1), has selected a specific course and <u>assigned as the course leader</u> (UC44) for the course in question.
Minimal Guarantee: The changes to the schedule are either fully saved or not saved at all.

Success Guarantee: The schedule, including changes made by the course leader, is available to all students and course assistants, and a confirmation has been displayed. The changes are reflected in the compiled schedules of any students currently registered for the course.

Main Success Scenario

1. The system displays a list of all activities currently in the schedule.
2. The user selects an activity.
3. The system requests the user to input:
   - Title.
   - Description.
   - Date.
   - Time.
4. The user inputs new values.
5. The system validates the input and displays a preview.
6. User confirms the preview.
7. The system saves the schedule.
8. The system displays a confirmation that the edit was successful.

Extensions

5a. The input doesn't validate.
1. The system notifies the user of the problem and allows the user to correct the problem.
2. User edits the input and resubmits.
3. Continue from step 5.

6a. The user is not satisfied with the preview and does not confirm.
1. The user edits the values and resubmits.
2. Continue from step 5.

Frequency of Occurrence: Rarely during a course and almost never otherwise.

Use Case UC19: Remove Existing Scheduled Activity

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to remove a scheduled activity in case the activity is cancelled.
- *Course assistants:* Want an accurate schedule so that they know when activities are held.
- *Students:* Want an accurate schedule so that they know when activities are held.

Trigger: The user selects to remove a scheduled activity.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The scheduled activity is either fully removed or not removed at all.
Success Guarantee: The scheduled activity is removed from the schedule and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of all activities currently in the schedule.
2. The user selects an activity to remove.
3. The system requests a confirmation for the removal.
4. The user confirms the removal.
5. The system removes the activity.
6. The system displays a confirmation that the activity was removed.

Extensions

4a. The user doesn't confirm the removal.
   1. Continue from step 1.

Frequency of Occurrence: Rarely during a course and almost never otherwise.

Use Case UC20: View Scheduled Activity from Course Schedule

Primary Actor: Student.
Stakeholders and Interests:
- *Course leaders:* Want students to be able to view the schedule so that they can attend the teaching activities as they wish.
- *Course assistants:* Want students to be able to view the schedule so that they can attend the teaching activities that the course assistant is involved in.
- *Students:* Want to be able to attend the teaching activities as they wish and get an overview of the courses' activities.

Trigger: The user selects to view the course schedule.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to the course website.
Minimal Guarantee: The schedule has not been changed.
Success Guarantee: The system displays information on a scheduled activity in the web browser.

Main Success Scenario

1. The system displays the course schedule.
2. The user selects an activity.
3. The system displays detailed information for the selected activity.

Frequency of Occurrence: Almost daily for students all days preceding a weekday.

Use Case UC21: View Scheduled Activity from Compiled Schedule

Primary Actor: Student.
Stakeholders and Interests:
- *Course leaders:* Want students to be able to view the schedule so that they can attend the teaching activities as they wish.
- *Course assistants:* Want students to be able to view the schedule so that they can attend the teaching activities that the course assistant is involved in.
- *Students:* Want to be able to get an overview of the courses' activities.

Trigger: The user selects to view the compiled schedule.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to the personal page.
Minimal Guarantee: The schedule has not been changed.
Success Guarantee: The system displays information on a scheduled activity in the web browser.

Main Success Scenario

1. The system displays the compiled schedule.
2. The user selects an activity.
3. The system displays the description for the selected activity.

Frequency of Occurrence: Almost daily for students all days preceding a weekday.

Use Case UC22: Export Schedule in iCalendar Format

Primary Actor: Student.
Stakeholders and Interests:
- *Course leaders:* Want students to be able to export the schedule so that they can follow the course activities when they don't have Internet access.
- *Course assistants:* Want students to be able to export the schedule so that they can follow the course activities when they don't have Internet access.
- *Students:* Want to be able to export the schedule so that it can be used in external systems, such as calendar applications.

Trigger: The user selects to export the schedule.
Preconditions: The user is viewing his or her schedule.
Minimal Guarantee: The schedule has not been changed.
Success Guarantee: The user has acquired a file in the iCalendar format representing his or her personal schedule.

Main Success Scenario

1. The system provides the user with a file containing the schedule in the iCalendar format.
2. The user downloads the iCalendar file to his or her computer.

Frequency of Occurrence: More often during the beginning of periods (but reasonably no more than once per student).

Use Case UC23: Add Deadline

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to give information about deadlines for a course to students and course assistants.
- *Course assistants:* Want to be able to get information about deadlines that they have responsibility for.
- *Students:* Want to know deadlines for courses to plan their work.

Trigger: The user selects to add a new deadline.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.

Minimal Guarantee: The deadline is either fully saved or not saved at all.
Success Guarantee: The list of deadlines for the course in question is updated, including the new deadline added by the course leader, and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - A title for the deadline.
   - The date for the deadline.
   - A description for the deadline.
2. The user inputs a title, a date and a description.
3. The system validates the input.
4. The system displays a summary of the new deadline, and requests the user to confirm.
5. The user confirms the deadline.
6. The system saves the deadline.
7. The system displays a confirmation that the new deadline was successfully saved.

Extensions

3a. The input doesn't validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

5a. The user doesn't confirm the deadline.
   1. The system allows the user to edit the input.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

Frequency of Occurrence: A couple of times at the beginning of each course, but might also be used at rare occasions during the course.

Use Case UC24: Edit Existing Deadline

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to give updated information about deadlines for a course to students and course assistants.
- *Course assistants:* Want to be able to get updated information about deadlines that they have responsibility for.
- *Students:* Want to know deadlines for courses to plan their work.

Trigger: The user selects to edit existing deadlines.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The deadline or changes to a deadline is either fully saved or not saved at all.
Success Guarantee: The list of deadlines for the course in question is updated, reflecting the changes made by the course leader, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing deadlines.
2. The user selects the deadline to be edited.
3. The system requests the user to input:
   - A title for the deadline.
   - Date for the deadline.
   - A description for the deadline.
4. The user inputs changes.
5. The system validates the input.
6. The system saves the deadline.
7. The system displays a confirmation that the new deadline was successfully saved.

Extensions

5a. The input doesn't validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 5.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC25: Remove Existing Deadline

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to be able to give updated information about deadlines for a course to students and course assistants.
- *Course assistants:* Want to be able to get updated information about deadlines that they have responsibility for.
- *Students:* Want to know deadlines for courses to plan their work.

Trigger: The user selects to remove existing deadline.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The deadline is either fully removed or not removed at all.
Success Guarantee: The list of deadlines for the course in question is updated, reflecting the changes made by the course leader, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing deadlines.
2. The user selects the deadline to be removed.
3. The system displays details for the deadline and requests confirmation.
4. The user confirms removal.
5. The system removes the deadline.
6. The system displays a confirmation that the deadline has been removed.

Extensions

4a. The user doesn't want to remove the deadline and cancels the removal.
   1. Continue from step 1.

Frequency of Occurrence: Very rarely throughout the duration of a course.

Use Case UC26: View Deadlines

Primary Actor: Student.
Stakeholders and Interests:

- *Course leaders:* Want students to be able to get information on deadlines (so assignments are more likely to be handed in on time).
- *Course assistants:* Want students to be able to get information on deadlines (so assignments are more likely to be handed in on time).
- *Students:* Want information on deadlines for a specific course (to better manage their workload).

Trigger: The user selects to view deadlines.
Preconditions: The user is using a computer with a working Internet connection and has started the web browser Mozilla Firefox and has navigated to the course website.
Minimal Guarantee: No deadlines have been changed.
Success Guarantee: The system displays a list of deadlines from the selected course.

Main Success Scenario

1. The system displays the deadlines for the selected course.

Frequency of Occurrence: A couple of times per week for each individual student, but depends very much on the course and the course layout that the student is attending.

Use Case UC27: View Overview of Deadlines

Primary Actor: Student.
Stakeholders and Interests:

- *Course leaders:* Want students to be able to get information on deadlines (so assignments are more likely to be handed in on time).
- *Course assistants:* Want students to be able to get information on deadlines (so assignments are more likely to be handed in on time).
- *Students:* Want information on deadlines for all courses the student is registered for (to better manage their workload).

Trigger: User selects to view deadlines.
Preconditions: The user is using a computer with a working Internet connection, has started the web browser Mozilla Firefox and has navigated to the personal page.
Minimal Guarantee: No deadlines have been changed.
Success Guarantee: A list of all deadlines for all the courses the user is registered for and a description of a specific deadline has been displayed.

Main Success Scenario

1. The system displays a list of all deadlines for the courses the user is registered for.
2. The user selects a deadline.
3. The system displays more detailed information for the selected deadline.

Frequency of Occurrence: A couple of times per week for each individual student, but depends very much on the courses and the course layouts that the student is attending.

Use Case UC28: Upload File

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to make files available to students and course assistants via the course website.
- *Course assistants:* Want the course website to be up-to-date with files related to the course.
- *Students:* Want the course website to be up-to-date with files related to the course.
- *University:* Wants relevant course material to be provided to the students via the course website.

Trigger: The user selects to upload a file.
Preconditions: The user is authenticated (UC1), has a selected a specific course and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The file is either successfully uploaded or not uploaded at all.
Success Guarantee: The file has been uploaded and is available on the course website, and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - A title for the file.
   - A description for the file.
   - The file itself.
2. The user enters the required information and selects the file to upload.
3. The system validates the input.
4. The system saves the file, title and description.
5. The system publishes the file on the course website.
6. The system displays a confirmation that the file has been uploaded and published.

Extensions

3a. The input doesn’t validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 4.

Frequency of Occurrence: A couple of times per week for each active course, but varying a lot depending on the course layout.

Use Case UC29: Edit Existing File

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want the files on the course website to be up-to-date.
- *Course assistants:* Want the course website to be up-to-date with files related to the course.
- *Students:* Want the course website to be up-to-date with files related to the course.
- *University:* Wants relevant and up-to-date course material to be provided to the students via the course website.

Trigger: The user selects to edit existing files.

Preconditions: The user is authenticated (UC1), has selected a specific course to work with and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The changes to a file is either fully saved or not saved at all.
Success Guarantee: The requested changes have been made and are available on the course website, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing files.
2. The user selects the file to edit.
3. The system requests the user to input:
   - A title for the file.
   - A description for the file.
   - The file itself.
4. The user edits the input.
5. The system validates the input.
6. The system replaces the old file and displays the new version on the course website.
7. The system displays a confirmation that the file has been edited.

Extensions

5a. The input doesn't validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 6.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC30: Remove Existing File

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to remove files on the course website.
- *Course assistants:* Want the course website to be up-to-date with files related to the course.
- *Students:* Want the course website to be up-to-date with files related to the course.
- *University:* Wants relevant course material to be provided to the students via the course website.

Trigger: The user selects to remove an existing file.
Preconditions: The user is authenticated (UC1), has selected a specific course to work with and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The file is either fully removed or not removed at all.
Success Guarantee: The file is removed and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing files.
2. The user selects the file to remove.
3. The system displays the file selected by the user and requests confirmation.
4. The user confirms the removal.
5. The system removes the file.

6. The system displays a confirmation that the file has been removed.

Extensions

4a. The user doesn't want to remove the file selected and cancels removal.
   1. Continue from step 2.

Frequency of Occurrence: A few times per week for each active course, but varying a lot depending on the course layout.

Use Case UC31: Download File

Primary Actor: Course leader, course assistant and student.
Stakeholders and Interests:
- *Course leaders:* Want to provide students and course assistants material essential for the course.
- *Course assistants:* Want to be able to get material provided by the course leader.
- *Students:* Want to be able to get material provided by the course leader.

Trigger: The user selects to download files.
Preconditions: The user is authenticated (UC1) and registered for the course (UC10 and UC39) and have navigated to the course website.
Minimal Guarantee: No uploaded files have been changed.
Success Guarantee: The file has been downloaded to the user's computer.

Main Success Scenario

1. The system displays a list of all uploaded files for the course.
2. The user selects a file to download.
3. The system sends the file to the user.

Frequency of Occurrence: A couple of times per week, much depending on how frequently course material is made available.

Use Case UC32: Add Course Assignment

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want themselves and course assistants to be able to enter results for the students for each course assignment.
- *Course assistants:* Want to be able to enter results for the students for each course assignment.
- *Students:* Want course leaders and course assistants to be able to enter results for the assignments.

Trigger: The user selects to add course assignments.
Preconditions: The user is authenticated (UC1), has selected the course and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course assignment is either fully saved or not saved at all and belongs to a course.
Success Guarantee: The course assignment is added for the course in question and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - A title for the course assignment.
   - A description for the course assignment.
2. The user inputs a title and a description.
3. The system validates the input.
4. The system displays a summary of the new course assignment, and requests the user to confirm.
5. The user confirms the course assignment.
6. The system saves the course assignment.
7. The system displays a confirmation that the new course assignment was successfully saved.

Extensions

3a. The title or description is invalid.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

5a. The user doesn't confirm the course assignment.
   1. The user edits the course assignment and resubmits.
   2. Continue from step 3.

Frequency of Occurrence: A couple of times at the beginning of each course, but might also be used at rare occasions during the course.

Use Case UC33: Edit Existing Course Assignment

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want the information regarding course assignments to be correct.
- *Course assistants:* Want the information regarding course assignments to be correct.
- *Students:* Want course leaders and course assistants to be able to enter results for the assignments.

Trigger: User selects to edit an existing course assignment.
Preconditions: The user is authenticated (UC1), has selected a specific course and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The changes to the course assignment are either fully saved or not saved at all and the course assignment belongs to a course.
Success Guarantee: The course assignments for the course in question are updated, reflecting the changes made by the course leader, and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing course assignment.
2. The user selects the course assignment to be edited.
3. The system requests the user to input:
   - A title for the course assignment.
   - A description for the course assignment.

4. The user inputs changes.
5. The system validates the input.
6. The system saves the course assignment.
7. The system displays a confirmation that the new course assignment was saved.

Extensions

5a. The title or description is invalid.
  1. The system notifies the user of the problem.
  2. The user edits the input and resubmits.
  3. Continue from step 5.

Frequency of Occurrence: A couple of times at the beginning of each course, but might also be used at rare occasions during the course.

Use Case UC34: Remove Existing Course Assignment

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Only want relevant assignments in the system.
- *Course assistants:* Only want relevant assignments in the system.
- *Students:* Want course leaders and course assistants to be able to enter results for the assignments.

Trigger: The user selects to remove an existing course assignment.
Preconditions: The user is authenticated (UC1), has selected the course and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course assignment is either fully removed or not removed at all.
Success Guarantee: The course assignment is removed and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing course assignments.
2. The user selects the course assignment to be removed.
3. The system displays details for the course assignment and requests confirmation.
4. The user confirms removal.
5. The system removes the course assignment.
6. The system displays a confirmation that the course assignment has been removed.

Extensions

4a. The user doesn't want to remove the course assignment and cancels the removal.
  1. Continue from step 1.

Frequency of Occurrence: A couple of times at the beginning of each course, but might also be used at rare occasions during the course.

Use Case UC35: Register Results

Primary Actor: Course leader and course assistant.
Stakeholders and Interests:

- *Course leaders:* Want an efficient way to register results for students registered for the course they are responsible for.
- *Course assistants:* Want an efficient way to register results for students registered for the course that they have been assigned to.
- *Students:* Want results to be saved and updated as soon as possible after the grading is finished.

Trigger: The user selects the course for which to register results.
Preconditions: The user is authenticated (UC1), has selected a specific course and have been assigned the privileges (UC44) to register results for the course.
Minimal Guarantee: The results are either registered properly or not registered at all.
Success Guarantee: One or more students' results are registered to the system and saved and a confirmation has been displayed.

Main Success Scenario

1. The user selects to register results.
2. The system displays a list of the registered students and course assignments.
3. The user assigns the grades for students and course assignments.
4. The user saves the changes.
5. The system validates the input.
6. The system saves the changes.
7. The system displays a confirmation that the results have been saved.

Extensions

5a. The system doesn't validate the input.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 5.

Frequency of Occurrence: Frequently during each course and student depending very much on the course layout.

Use Case UC36: View Results

Primary Actor: Student.
Stakeholders and Interests:

- *Course leaders:* Want students to be able to view their results.
- *Course assistants:* Want students to be able to view their results.
- *Students:* Want to view their own results, and don't want other students to be able to view their results.
- *University:* Wants students to be able to view their results, and only their own results, respecting the user's privacy.

Trigger: The user selects to view results for a course.
Preconditions: The user is authenticated (UC1) and registered for the course (UC38 and UC39) he or she is trying to look up his or her results and has navigated to the personal page.
Minimal Guarantee: No results have been changed.
Success Guarantee: The student has seen his or her results in the specified course.

Main Success Scenario

1. The system displays a list of courses the user is registered for.
2. The user selects the course for which to view results.
3. The system displays all the student's results for the selected course.

Frequency of Occurrence: Frequently during the course, and probably more frequently towards the end of the course when more results are registered. The frequency depends very much on how many assignments are included in the course. The student is expected to view his or her results a couple of times per assignment, depending on how quickly the result is registered.

Use Case UC37: View Registered Students

Primary Actor: Course leader and course assistant.
Stakeholders and Interests:
- *Course leaders:* Want to see registered students.
- *Course assistants:* Want to see registered students.
- *Students:* Want the course leader to see that they are registered.

Trigger: User selects to view registered students.
Preconditions: The user is authenticated (UC1), has selected a specific course and is assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: No registered users have been unregistered and no unregistered students have been registered.
Success Guarantee: The system displays a list of registered students for the course in question.

Main Success Scenario

1. The system displays a list of registered students for the course in question.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC38: Confirm Application to Get Registered for Course

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to maintain an accurate list of students taking the course and be able to view a list of registered students.
- *Course assistants:* Want to have a complete list of registered students in the course when they register results for students.
- *Students:* Want to get their application to join a course confirmed so they can access files and have results registered.
- *University:* Wants an efficient administration of registration for courses.

Trigger: The user selects to confirm course applications.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: A student who was previously awaiting confirmation is either fully registered for the course, or awaiting confirmation for the course.

Success Guarantee: All users approved for the course by the course leader have been accepted into the course, a confirmation has been displayed and the course leader can view registered users.

Main Success Scenario

1. The system displays a list of students who have applied for the course.
2. The user selects all students that are to be accepted into the course and accepts these.
3. The system updates the status for the selected students.
4. The system displays a confirmation of which students have been accepted to the course.

Frequency of Occurrence: A couple of times during the start of each course but very rarely otherwise. Most courses starts in between periods.

Use Case UC39: Apply for Course

Primary Actor: Student.
Stakeholders and Interests:

- *Course leaders:* Want the students to apply for the course so that they can get registered, which enables the course leader to plan resource allocation and register results for students.
- *Course assistants:* Want the students to apply for the course so that they can get registered, which enables the course assistant to plan resource allocation and register results for students.
- *Students:* Want to apply for the course so that they can get registered, which enables them to get results and credits.
- *University:* Wants to know how many students are taking a course, and wants an efficient administration.

Trigger: The user selects to apply to the course.
Preconditions: The user is authenticated (UC1), not registered or applied for the course in question and has navigated to the course website.
Minimal Guarantee: The application to join the course is either fully saved or not saved at all.
Success Guarantee: The student has applied to the course, awaiting confirmation from the course leader to get registered, and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to confirm the application to join the course.
2. The user confirms the application.
3. The system saves the application.
4. The system displays a confirmation that the user has applied for the course, and is awaiting confirmation from the course leader to get registered.

Extensions

2a. The user doesn't want to apply for the course and cancels.
   1. The system redirects the user to the course website.

Frequency of Occurrence: Once per student and course when a new course starts. Most courses start in between periods.

Use Case UC40: Unregister Registered Student

Primary Actor: Course leader.
Stakeholders and Interests:

- *Course leaders:* Want to maintain an accurate list of students taking the course and be able to view a list of registered students.
- *Course assistants:* Want to have a complete list of registered students in the course when they register results for students.
- *Students:* Want to get their application to take a course confirmed so they can access files and have results registered.
- *University:* Wants an efficient administration of registration for courses.

Trigger: The user selects to unregister registered students.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: Any student who was previously registered for the course is either still fully registered for the course or not registered for the course at all.
Success Guarantee: The selected student is unregistered and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of registered students.
2. The user selects a student to unregister from the course.
3. The system displays the student selected by the user and requests confirmation.
4. The user confirms.
5. The system unregisters the student from the course, updating the student’s status.
6. The system displays a confirmation that the student was unregistered from the course.

Extensions

4a. The user doesn’t confirm.
   1. Continue from step 1.

Frequency of Occurrence: Rarely, when students have been accepted by mistake.

Use Case UC41: Add User Account

Primary Actor: System administrator.
Stakeholders and Interests:

- *Course leaders:* Want accounts so they can get privileges to manage their course websites.
- *Course assistants:* Want accounts to be able to register results in courses they are involved.
- *Students:* Want an account so they can get registered for courses and get easy access to news and information about the courses they apply for.
- *System administrators:* Want to add user accounts.
- *University:* Unauthorized people shouldn’t be able to change the content of course websites, such as adding inappropriate content.

Trigger: The user selects to add a user account.

Preconditions: The user is authenticated (UC1) and assigned as system administrator (UC44).
Minimal Guarantee: The user account is either properly saved or not saved at all.
Success Guarantee: The user account is added and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Username.
   - Password.
2. The user inputs a username and password.
3. The system validates the input.
4. The system creates the account.
5. The system displays a confirmation that the user account has been added.

Extensions

3a. The username or password doesn't validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

Frequency of Occurrence: When a new person joins the university. The frequency depends much on the size of the university and the employee turnover. Occurs more frequently in between periods when new courses start and the need for user accounts arises, and especially at the beginning of each semester.

Use Case UC42: Edit User Account Password

Primary Actor: System administrator.
Stakeholders and Interests:
- *Course leaders:* Want the system administrator to be able edit the password in case he or she forgets the password.
- *Course assistants:* Want the system administrator to be able edit the password in case he or she forgets the password.
- *Students:* Want the system administrator to be able edit the password in case he or she forgets the password.
- *System administrators:* Want to allow users to access their accounts.
- *University:* Don't want the risk of having compromised accounts.

Trigger: The user selects to change a user account password.
Preconditions: The user is authenticated (UC1) and assigned as system administrator (UC44).
Minimal Guarantee: The user account is either properly saved or not saved at all.
Success Guarantee: The user account password has been updated and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Search text for a user account.
2. The user enters the search text for a specific user account to be searched for.

3. The system displays all user accounts with username containing the search text.
4. The user selects the user account for which password the user wants to change.
5. The system requests the user to input:
   - Password.
6. The user enters a new password.
7. The system validates the input.
8. The system saves the new password.
9. The system displays a confirmation that the desired changes have been saved.

Extensions

3a. No matching user exists.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

7a. The input doesn't validate.
   1. The system notifies the user of the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 7.

Frequency of Occurrence: Varies depending on how many accounts there are, and how forgetful the users of the system are.

Use Case UC43: Remove User Account

Primary Actor: System administrator.
Stakeholders and Interests:
- *System administrators:* Want to be able to remove user accounts.
- *University:* Don’t want unauthorized people to be able to access restricted parts of the system.

Trigger: The user selects to remove a user account.
Preconditions: The user is authenticated (UC1) and assigned as system administrator (UC44).
Minimal Guarantee: The user account is either fully removed or not removed at all.
Success Guarantee: The user account is removed and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Search text for a user account.
2. The user enters the search text for a specific user account to be searched for.
3. The system displays all accounts with username containing the search text.
4. The user selects the account to be removed.
5. The system requests the user to confirm the removal.
6. The user confirms the removal.
7. The system removes the account.
8. The system displays a confirmation that the user account was removed.

Extensions

3a. No matching user exists.
    1. The system notifies the user of the problem.
    2. The user edits the input and resubmits.
    3. Continue from step 3.

6a. The user doesn't want to remove the selected user account and cancels the removal.
    1. Continue from step 1.

Frequency of Occurrence: Rarely.

Use Case UC44: Edit User Privileges

Primary Actor: System administrator.
Stakeholders and Interests:
- *Course leaders:* Want to have the privileges required to manage course website.
- *Course assistants:* Want to have the privileges required to register students' results.
- *System administrators:* Assigning the necessary privileges to the user accounts.
- *University:* Unauthorized people shouldn't be able to change the content of course websites, such as adding inappropriate content.

Trigger: The user selects to edit user privileges.
Preconditions: The user is <u>authenticated</u> (UC1) and <u>assigned as system administrator</u> (UC44).
Minimal Guarantee: User accounts have either the previous privileges or the new privileges as intended by the primary actor.
Success Guarantee: The user privileges are edited and a confirmation has been displayed. Privileges for course leaders or course assistants are bound to specified courses.

Main Success Scenario

1. The system requests the user to input:
    - Search text for a user account.
2. The user specifies the search text for a specific user account to be searched for.
3. The system displays all user accounts with username containing the search text.
4. The user selects an account.
5. The system requests the user to input which of the following privileges the user account should have:
    - Course leader.
    - Course assistant.
    - System administrator.
6. The user inputs which privileges the user account shall have.
7. The system requests the user to confirm.
8. The user confirms the input.
9. The system saves the changes.
10. The system displays a confirmation that the privileges have been assigned.

Extensions

3a. No matching users exist.
   1. The user edits the input and resubmits.
   2. Continue from step 3.

6a. The user selects to add privileges for course leader or course assistant.
   1. The system requests the user to input:
      - Search text for a course code.
   2. The user specifies the search text for a specific course code to be searched for.
   3. The system displays all courses that contain the course code.
      3a. No matching courses were found.
         1. The user edits the input and resubmits.
         2. Continue from step 6a.3.
   4. The user selects a course.
   5. The system saves the course to which the privileges will be associated with.
   6. Continue from step 7.

8a. The user doesn't want to assign privileges to the user account and cancels.
   1. Continue from step 1.

Frequency of Occurrence: When a new course leader is assigned to a course. The frequency depends a lot on the number of courses and how long the course leader stays with his or her course. Occurs more frequently in between periods when new courses start. Most of the course assistants are expected to be assigned by their course leaders.

Use Case UC45: Add Course Assistant

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to add course assistants who can register results for graded assignments on their own, to lessen the workload of the course leader.
- *Course assistants:* Want to be assigned the right user privileges to be able to help the course leader register results.
- *Students:* Want course assistants to help the course leader, so they (the students) can view their results as soon as possible.
- *University:* Wants the course to be run smoothly without major delays for results to get registered.

Trigger: The user selects to add a course assistant.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course assistant privilege is either fully saved or not saved at all.
Success Guarantee: The privilege for the course assistants' user accounts is updated and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - The username of the course assistant.
2. The user enters the username of the assistant.
3. The system validates the username.

4. The system requests the user to confirm.
5. The user confirms that the course assistant should be added.
6. The system adds the course assistant to the course.
7. The system displays a confirmation that the course assistant has been added to the course.

Extensions

3a. No matching user exists.
  1. The system notifies the user of the problem.
  2. The user corrects the problem and resubmits.
  3. Continue from step 3.

5a. The user doesn't want to add the user as course assistant, and cancels the operation.
  1. Continue from step 1.

Frequency of Occurrence: Once or a few times at the start of the course for each course, then very rarely.

Use Case UC46: Remove Course Assistant

Primary Actor: Course leader.
Stakeholders and Interests:
- *Course leaders:* Want to remove course assistants who are no longer active.
- *Course assistants:* Don't want to be assigned course assistants for a course they're not responsible for.
- *Students:* Want only authorized course assistants to be able to enter results.
- *University:* Wants the course to be run smoothly without major delays for results to get registered.

Trigger: The user selects to remove a course assistant.
Preconditions: The user is authenticated (UC1), has selected a specific course and assigned as the course leader (UC44) for the course in question.
Minimal Guarantee: The course assistant privilege is either fully removed or not removed at all.
Success Guarantee: The course assistant privilege is removed from the user account and a confirmation has been displayed.

Main Success Scenario

1. The system displays a list of existing course assistants.
2. The user selects the course assistant to be removed.
3. The system displays the course assistant that the user selected and requests confirmation.
4. The user confirms removal.
5. The system removes the course assistant from the course.
6. The system displays a confirmation that the course assistant was removed.

Extensions

4a. The user doesn't want to remove the course assistants and cancels the removal.
  1. Continue from step 1.

Frequency of Occurrence: Rarely throughout the duration of a course.

Use Case UC47: Add Course

Primary Actor: System administrator.
Stakeholders and Interests:

- *Course leaders:* Want courses to be added so they can create websites for courses they are responsible for.
- *Course assistants:* Want courses to be added to be able to register results for students.
- *Students:* Want courses to be added so that they can apply and get registered for them.
- *System administrators:* Want to create courses to help administration of course websites.
- *University:* Wants courses to be added so students can apply and get registered for them, and administration to be efficient.

Trigger: The user selects to add a new course.
Preconditions: The user is <u>authenticated</u> (UC1) and <u>assigned as system administrator</u> (UC44).
Minimal Guarantee: The course is either fully saved or not saved at all.
Success Guarantee: The course is added and a confirmation has been displayed.

Main Success Scenario

1. The system requests the user to input:
   - Course code.
2. The user inputs the course code.
3. The system validates the course code.
4. The system saves the new course.
5. The system displays a confirmation that the course was added.

Extensions

3a. The course code already exists.
1. The system notifies the user of the problem.
2. The user inputs a new course code.
3. Continue from step 3.

3b. The course code was invalid.
1. The system notifies the user of the problem.
2. The user inputs a new course code.
3. Continue from step 3.

Frequency of Occurrence: When a new course is added. Occurs mostly in between periods when new courses start.

Use Case UC48: Edit Existing Course

Primary Actor: System administrator.
Stakeholders and Interests:

- *Course leaders:* Want course codes to be correct and up-to-date.
- *Course assistants:* Want course codes to be correct and up-to-date.

- *Students:* Want course codes to be correct and up-to-date so that they can apply and get registered for them.
- *System administrators:* Want the system to only contain the courses that actually exist.
- *University:* Wants course information to be correct to avoid misunderstandings.

Trigger: The user selects to edit an existing course.
Preconditions: The user is <u>authenticated</u> (UC1) and <u>assigned as system administrator</u> (UC44).
Minimal Guarantee: The changes to a course is either fully saved or not saved at all.
Success Guarantee: The changes are saved, a confirmation has been displayed and the course is updated.

Main Success Scenario

1. The system requests the user to input:
   - Search text for a course code.
2. The user inputs a search text.
3. The system displays a list with matching courses.
4. The user selects a course to be edited.
5. The system requests the user to input:
   - Course code.
6. The user inputs a new course code.
7. The system validates the course code.
8. The system saves the changes to course.
9. The system displays a confirmation that the course has been edited.

Extensions

3a. No matching courses were found.
   1. The system notifies the user about the problem and allows the user to correct the problem.
   2. The user edits the input and resubmits.
   3. Continue from step 3.

7a. The course code already exists.
   1. The system notifies the user of the problem.
   2. The user inputs a new course code.
   3. Continue from step 7.

7b. The course code was invalid.
   1. The system notifies the user of the problem.
   2. The user inputs a new course code.
   3. Continue from step 7.

Frequency of Occurrence: When a new course is created. Occurs mostly in between periods when new courses start.

System Architecture

Client-server Architecture

The system has a typical client-server architecture, where the clients are made up of users interpreting and viewing the system's output through their web browsers. The architecture used is a thin-client architecture as the clients are only performing some simple processing mostly associated to the user interface, and is implemented using JavaScript.

The server is where the main part of the system lies. It consists mainly of a web server (handling basic HTTP connections and communication with the client) capable of producing dynamic web pages (using JSP) and a database server (namely MySQL 5.0).

As illustrated in the figure below, the system is using three-tier architecture.

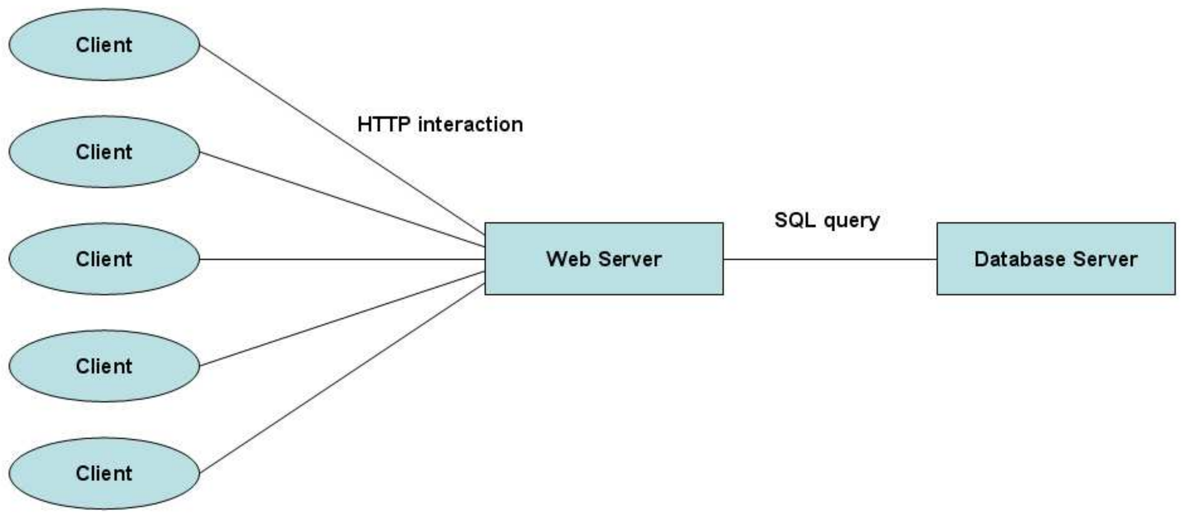

Figure 1: A schematic illustration of the system's three-tier architecture.

Distribution of Functionality

The system will be developed using an object oriented approach, and will mainly contain two groups of classes.

The first group contains classes representing real world entities in the system, such as a user, a course, a deadline or an information page. These classes will map directly to the data represented in the database. Furthermore these classes will not contain any functionality for manipulating the data in the database.

The second group of classes contains classes for manipulating the data. Functionality for editing, adding and removing data is gathered in this group of classes called controller classes. The functionality for adding a course is for example in the course controller, the functionality for reporting a result is in the result controller and the functionality for editing an information page lies in the information page controller.

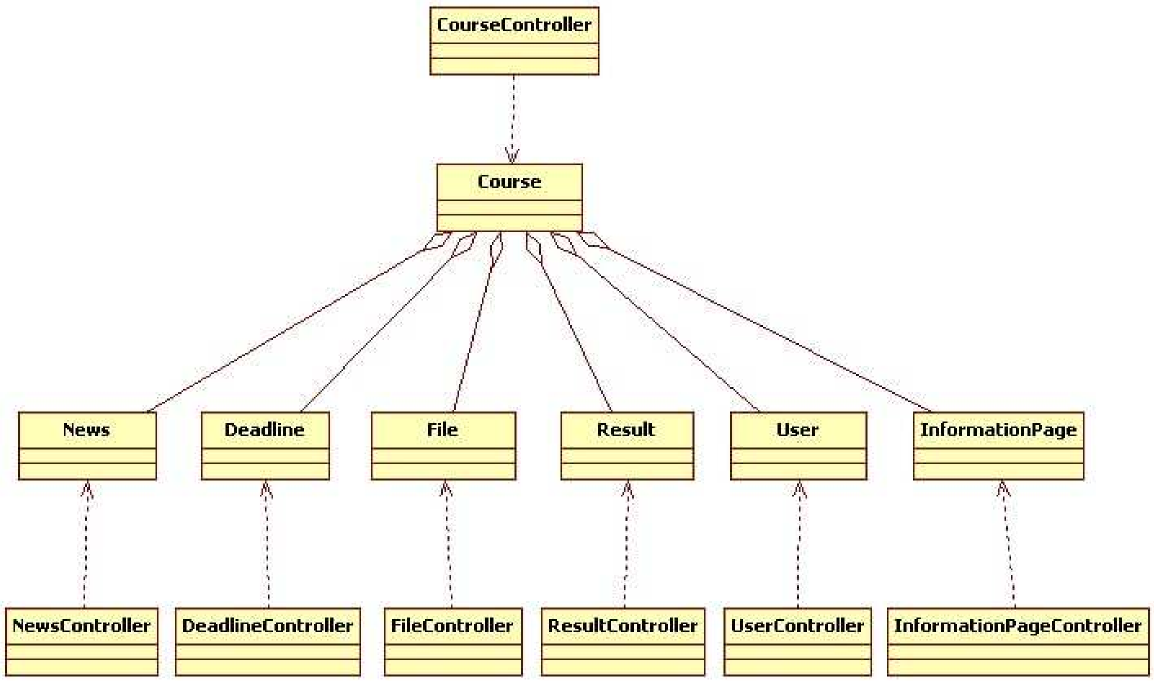

Figure 2: A schematic illustration of the system's classes. The controller classes contain functions for manipulating the data, while the classes representing real world entities only represent data.

System Requirements Specification

Functional Requirements

1. Authentication System

User Requirement	System Requirement
1.1	The system shall enable users to authenticate themselves, checking the login details against authorized users in the database. **Input:** Username and password. **Source for Input:** User (web browser). **Output:** User session and confirmation. **Destination for Output:** User (web browser). **Use Case:** UC1 (Authenticate to the system).

2. Personal Pages

User Requirement	System Requirement
2.1	The system shall provide authenticated users with a personal page where course news, scheduled activities, results and deadlines relevant for the user are available. Which course news, scheduled activities and deadlines to display is decided by which courses the user is registered for, which is available in the database. **Input:** Request to view a user's personal page. **Source for Input:** User (web browser). **Output:** Personal page. **Destination for Output:** User (web browser).

	Use Case: UC2 (View Personal Page).
2.2	The system shall enable any user, authenticated or otherwise, to access the schedule, deadlines and course news on a personal page. **Input:** Request to view personal page. **Source for Input:** User (web browser). **Output:** Personal Page. **Destination for Output:** User (web browser). **Use Case:** UC2 (View Personal Page).
2.3	The system shall display an overview of course news for courses that the user is registered for to the user. **Input:** Request to display overview of course news. **Source for Input:** User (web browser). **Output:** List of course news for the courses the student is registered for. **Destination for Output:** User (web browser). **Use Case:** UC3 (View Overview of Course News).

3. Creation Guide for Course Website

User Requirement	**System Requirement**
3.1	The system shall provide a creation guide for course websites to course leaders for courses they are responsible for, where course description, schedule, information pages and deadlines can be added to the course website. The system shall not save any parts of the course website permanently until the guide is completed. **Input:** Request to create a course website. **Source for Input:** User (web browser). **Output:** Course website and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC4 (Create Course Website).

4. Course Description

User Requirement	**System Requirement**
4.1	The system shall provide forms for course leaders to add and edit the course description, as well as a JSP page for managing the input from the form (adding and editing the course description in the database). **Input:** Course description. **Source for Input:** Course leader. **Output:** Course description and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC4 (Create Course Website) and UC5 (Edit Existing Course Description).
4.2	The system shall display the course description on the course website for any user. **Input:** Request to display course description. **Source for Input:** User (web browser).

	Output: Course description. **Destination for Output:** User (web browser). **Use Case:** UC6 (View Course Description).

5. Course News

User Requirement	System Requirement
5.1	The system shall provide a form for course leaders to add course news for the courses they are responsible for, as well as a JSP page for managing inputs from the form (adding course news in the database). **Input:** Course news. **Source for Input:** User (web browser). **Output:** Added course news and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC7 (Add Course News).
5.1	The system shall enable course leaders to choose course news in a course they are responsible for to be edited, a form for editing the course news and a JSP page for managing inputs from the form (editing the course news in the database). **Input:** Request to edit specific course news. **Source for Input:** User (web browser). **Output:** Edited course news and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC8 (Edit Existing Course News).
5.1	The system shall enable course leaders to choose course news in a course they are responsible for to be removed, as well as a JSP page for removing the course news from the database. **Input:** Request to remove specific course news. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC9 (Remove Existing Course News).
5.2	The system shall display course news for a specific course. **Input:** Request to display a course's course news. **Source for Input:** User (web browser). **Output:** The course news for a course order by date in descending order. **Destination for Output:** User (web browser). **Use Case:** UC10 (View Course News).

6. Information Pages

User Requirement	System Requirement
6.1	The system shall provide a form for course leaders to add information pages for the courses they are responsible for, as well as a JSP page for managing inputs from the form (adding information pages in the database).

	Input: Information page. **Source for Input:** User (web browser). **Output:** Added information page and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC11 (Add Information Page).
6.1	The system shall enable course leaders to choose an information page in a course they are responsible for to be edited, a form for editing the information page and a JSP page for managing inputs from the form (editing the information page in the database). **Input:** Request to edit a specific information page. **Source for Input:** User (web browser). **Output:** Edited information page and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC12 (Edit Existing Information Page).
6.1	The system shall enable course leaders to choose an information page in a course they are responsible for to be removed, as well as a JSP page for removing the information page from the database. **Input:** Request to remove a specific information page. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC13 (Remove Existing Information Page).
6.2	The system shall display an information page for a specific course. **Input:** Request to display an information page. **Source for Input:** User (web browser). **Output:** The selected information page. **Destination for Output:** User (web browser). **Use Case:** UC14 (View Information Page).

7. Schedule

User Requirement	**System Requirement**
7.1	The system shall provide forms for course leaders to add a schedule by importing a file in iCalendar format, as well as a JSP page for managing the input from the form (adding a schedule in the database). **Input:** iCalendar formatted file. **Source for Input:** User (web browser). **Output:** Schedule and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC15 (Import Course Schedule).
7.1	The system shall enable course leaders to choose to remove a schedule in a course they are responsible for, as well as a JSP page for removing the schedule activity from the database. **Input:** Request to remove a course schedule. **Source for Input:** User (web browser). **Output:** Confirmation.

	Destination for Output: User (web browser). **Use Case:** UC16 (Remove Existing Course Schedule).
7.2	The system shall provide a form for course leaders to add scheduled activities for the courses they are responsible for, as well as a JSP page for managing inputs from the form (adding a scheduled activity in the database). **Input:** Scheduled activity. **Source for Input:** User (web browser). **Output:** Added scheduled activity and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC17 (Add Scheduled Activity).
7.2	The system shall enable course leaders to choose a scheduled activity in a course they are responsible for to be edited, a form for editing the scheduled activity and a JSP page for managing inputs from the form (editing the scheduled activity in the database). **Input:** Request to edit a specific scheduled activity. **Source for Input:** User (web browser). **Output:** Edited scheduled activity and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC18 (Edit Existing Scheduled Activity).
7.2	The system shall enable course leaders to choose a scheduled activity in a course they are responsible for to be removed, as well as a JSP page for removing the scheduled activity from the database. **Input:** Request to remove a specific scheduled activity. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC19 (Remove Existing Scheduled Activity).
7.3	The system shall display the course schedule and display details for a specific scheduled activity upon request from the user. **Input:** Request to display course schedule. **Source for Input:** User (web browser). **Output:** Course schedule. **Destination for Output:** User (web browser). **Use Case:** UC20 (View Scheduled Activity from Course Schedule).
7.4	The system shall display a compiled schedule for a student to any user (authenticated or not) containing scheduled activities from all the courses the student is registered for. **Input:** Request to display compiled schedule. **Source for Input:** User (web browser). **Output:** Compiled schedule. **Destination for Output:** User (web browser). **Use Case:** UC21 (View Scheduled Activity from Compiled Schedule).
7.5	The system shall be able to export the schedule to iCalendar format

	for any user. **Input:** Request to export schedule. **Source for Input:** User (web browser). **Output:** iCalendar formatted file. **Destination for Output:** User (web browser). **Use Case:** UC22 (Export Schedule in iCalendar Format).

8. Deadlines

User Requirement	**System Requirement**
8.1	The system shall provide a form for course leaders to add deadlines in courses they are responsible for, as well as a JSP page for managing inputs from the form (adding a deadline in the database). **Input:** Deadline. **Source for Input:** User (web browser). **Output:** Added deadline and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC23 (Add Deadline)
8.1	The system shall enable course leaders to choose a deadline in a course they are responsible for to be edited, a form for editing the deadline and a JSP page for managing inputs from the form (updating the deadline in the database). **Input:** Request to edit a specific deadline. **Source for Input:** User (web browser). **Output:** Updated deadline and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC24 (Edit Existing Deadline).
8.1	The system shall enable course leaders to choose a deadline in a course they are responsible for to be removed, as well as a JSP page for removing the deadline from the database. **Input:** Request to remove a specific deadline. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC25 (Remove Existing Deadline).
8.2	The system shall display details for a deadline. **Input:** Request to display details for a deadline. **Source for Input:** User (web browser). **Output:** Deadline. **Destination for Output:** User (web browser). **Use Case:** UC26 (View Deadlines).
8.3	The system shall display an overview of deadlines in courses that the user is registered for to students. **Input:** Request to display overview of deadlines.

	Source for Input: User (web browser). **Output:** List of deadlines for the courses the student is registered for. **Destination for Output:** User (web browser). **Use Case:** UC27 (View Overview of Deadlines).

9. Uploaded Files

User Requirement	System Requirement
9.1	The system shall provide a form for course leaders to upload files for the courses they are responsible for, as well as a JSP page for managing inputs from the form (saving the uploaded file in the file system and saving the file description in the database). **Input:** File. **Source for Input:** User (web browser). **Output:** Uploaded file, file description and confirmation. **Destination for Output:** File system, database and user (web browser). **Use Case:** UC28 (Upload File).
9.1	The system shall enable course leaders to choose a file in a course they are responsible for to be edited, a form for editing the file and a JSP page for managing inputs from the form (updating the file in the file system and/or the description for the file in the database). **Input:** Request to update a specific file. **Source for Input:** User (web browser). **Output:** Updated file and/or updated file description and confirmation. **Destination for Output:** File system and/or database and user (web browser). **Use Case:** UC29 (Edit Existing File).
9.1	The system shall enable course leaders to choose a file in a course they are responsible for to be removed, as well as a JSP page for removing the file from the file system and the description for the file from the database. **Input:** Request to remove a specific file. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC30 (Remove Existing File).
9.2	The system shall enable course leaders, course assistants and students registered for a course to download files belonging to the course. **Input:** Request to download a file. **Source for Input:** User (web browser). **Output:** File. **Destination for Output:** User (web browser). **Use Case:** UC31 (Download File).

10. Course Assignments

User Requirement	System Requirement
10.1	The system shall provide a form for course leaders to add course assignments for the courses they are responsible for, as well as a JSP page for managing inputs from the form (adding course assignments in the database). **Input:** Course assignment. **Source for Input:** User (web browser). **Output:** Added course assignment and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC32 (Add Course Assignments).
10.1	The system shall enable course leaders to choose a course assignment in a course they are responsible for to be edited, a form for editing the course assignment and a JSP page for managing inputs from the form (editing the course assignment in the database). **Input:** Request to edit specific course assignment. **Source for Input:** User (web browser). **Output:** Edited course assignment and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC33 (Edit Existing Course Assignment).
10.1	The system shall enable course leaders to choose course assignment in a course they are responsible for to be removed, as well as a JSP page for removing the course assignment from the database. **Input:** Request to remove specific course assignment. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC34 (Remove Existing Course Assignment).

11. Results

User Requirement	System Requirement
11.1	The system shall provide a form for course leaders and course assistants to register results for the courses they are responsible for, as well as a JSP page for managing inputs from the form (saving the results in the database). **Input:** Results. **Source for Input:** User (web browser). **Output:** Results and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC35 (Register Results).
11.2	The system shall enable authenticated students registered for a course to view results belonging to the course. **Input:** Request to view results. **Source for Input:** User (web browser).

	Output: Results. **Destination for Output:** User (web browser). **Use Case:** UC36 (View Results).

12. Course Registration

User Requirement	System Requirement
12.1	The system shall display the registered students to course leaders for courses they are responsible for. **Input:** Request to display registered students. **Source for Input:** User (web browser). **Output:** List of registered students for a course. **Destination for Output:** User (web browser). **Use Case:** UC37 (View Registered Students).
12.2	The system shall require students to be verified by the course leader responsible for the course before they are accepted into the course. **Input:** Accepted users to be registered into the course. **Source for Input:** User (web browser). **Output:** Updated user status and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC38 (Confirm Application to Get Registered for Course).
12.3	The system shall enable students to apply for a course (but not registering them for the course until the course leader has verified the students). **Input:** User and course to be applied for. **Source for Input:** User (web browser). **Output:** Updated user status and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC39 (Apply for Course).
12.4	The system shall enable course leaders to unregister registered students in courses they are responsible for. **Input:** User and course to be unregistered for. **Source for Input:** User (web browser). **Output:** Updated user status and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC40 (Unregister Registered Student).

13. Miscellaneous

User Requirement	System Requirement
13.1	The system shall enable system administrators to edit the privileges of other users (enabling them to give a user privileges as a course leader, course assistant and system administrator), including assigning several privileges to one user (one user can be course leader for several courses and system administrator at the same time). **Input:** Username and privileges. **Source for Input:** User (web browser).

	Output: Updated user privileges and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC44 (Edit User Privileges).
13.2	The system shall enable course leaders to add course assistants for the courses they are responsible for. **Input:** Username. **Source for Input:** User (web browser). **Output:** Updated user privileges and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC45 (Add Course Assistants).
13.2	The system shall enable course leaders to remove course assistants for the courses they are responsible for. **Input:** Username. **Source for Input:** User (web browser). **Output:** Updated user privileges and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC46 (Remove Course Assistants).
13.3	The system shall provide a form for system administrators to add a course, as well as a JSP page for managing inputs from the form (adding a course in the database). **Input:** Course code. **Source for Input:** User (web browser). **Output:** Added course and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC47 (Add Course).
13.3	The system shall enable system administrators to choose a course to be edited, a form for editing the course and a JSP page for managing inputs from the form (updating the course in the database). **Input:** Request to update a specific course. **Source for Input:** User (web browser). **Output:** Updated course and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC48 (Edit Existing Course).
13.4	The system shall provide a form for system administrators to add a user account, as well as a JSP page for managing inputs from the form (adding a user account in the database). **Input:** Username and password. **Source for Input:** User (web browser). **Output:** Added user account and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC41 (Add User Account).

13.4	The system shall enable system administrators to choose a user account which's password is to be edited, a form for editing the password and a JSP page for managing inputs from the form (editing the user account's password in the database). **Input:** Request to edit a user account's password. **Source for Input:** User (web browser). **Output:** Edited user account and confirmation. **Destination for Output:** Database and user (web browser). **Use Case:** UC42 (Edit User Account Password).
13.4	The system shall enable system administrators to choose a user account to be removed, as well as a JSP page for removing the user account from the database. **Input:** Request to remove a specific user account. **Source for Input:** User (web browser). **Output:** Confirmation. **Destination for Output:** User (web browser). **Use Case:** UC43 (Remove User Account).

Non-functional Requirements

14. Product Requirements

User Requirement	**System Requirement**
14.1	The system shall make the personal page available by limiting the downtime caused by an unavailable database. For this reason the system will use caching to limit the number of database accesses needed. **Test:** It shall be possible to able to fetch data even when the database server is not running.
14.2	The system shall make the loading of the personal page efficient by caching data from the database as fetching data from the database is what will be the most demanding task. **Test:** It shall be possible to able to fetch data even when the database server is not running.
14.3	The system shall improve the scalability by caching data from the database, and thereby removing some of the processing that is (often) the same between page requests. **Test:** It shall be possible to able to fetch data even when the database server is not running.
14.4	The system shall only provide English as a language, and therefore doesn't need to implement solutions for choosing language or displaying translations of the system. **Test: -**

15. External Requirements

User Requirement	System Requirement
15.1	The system shall check the user’s privileges upon request for administration pages for course news and display an error message if the user is not authorized to view the page. **Test:** If a user tries to administer course news for a course for which he or she is not authenticated as course leader for, the system will display an error message.
15.2	The system shall check the user’s privileges upon request for a student’s results and display an error message if the user is not authorized to view the page. **Test:** If a user tries to view results for a student that he or she is not authenticated as, the system will display an error message.

System Evolution

Fundamental Assumptions

- The use of the system will require an active Internet connection.
- Users of the system will be required to use Mozilla Firefox 2.0.
- The system will only be required to work with the MySQL 5.0 database server.
- The system will run on the Apache Tomcat web server version 6.0.
- The users of the system will need no specific training in order to use the system.

Anticipated Changes

- We foresee that multiprocessor systems will become a lot more common in the future, however we do not believe this will require any changes to the system itself, but rather to the web- and database servers used.
- The users of the system could possibly want it to provide more functions in the future, in order to provide a common interface to all university related information. In the first version of the system, we don’t intend to provide any sort of plug-in functionality to allow for development of new add-in modules that can provide additional functionality.
- Since the system stores results for students, which could potentially be sensitive information, legislative changes could introduce additional requirements on how the system stores these results.
- Security vulnerabilities may be found in the software needed to run the system when these software products are too old to still be supported, forcing the use of newer versions for which the system hasn’t been built.

Appendices

Minimal Configuration

Hardware

- 1 GB RAM
- 2 Ghz x86 single core CPU (or equivalent)
- Internet connection

Software for Server

- Apache Tomcat 6.0
- MySQL 5.0

Software for Client

- Mozilla Firefox 2.0

Optimal Configuration

Hardware

- 3+ GB RAM
- 3+ GHz x86 CPU (or equivalent)
- Internet connection with at least 10 Mbps bandwidth

Software for Server

- Apache Tomcat 6.0
- MySQL 5.0

Software for Client

- Mozilla Firefox 2.0

Database Description

Below is a description of the anticipated database structure presented in an entity-relationship model.

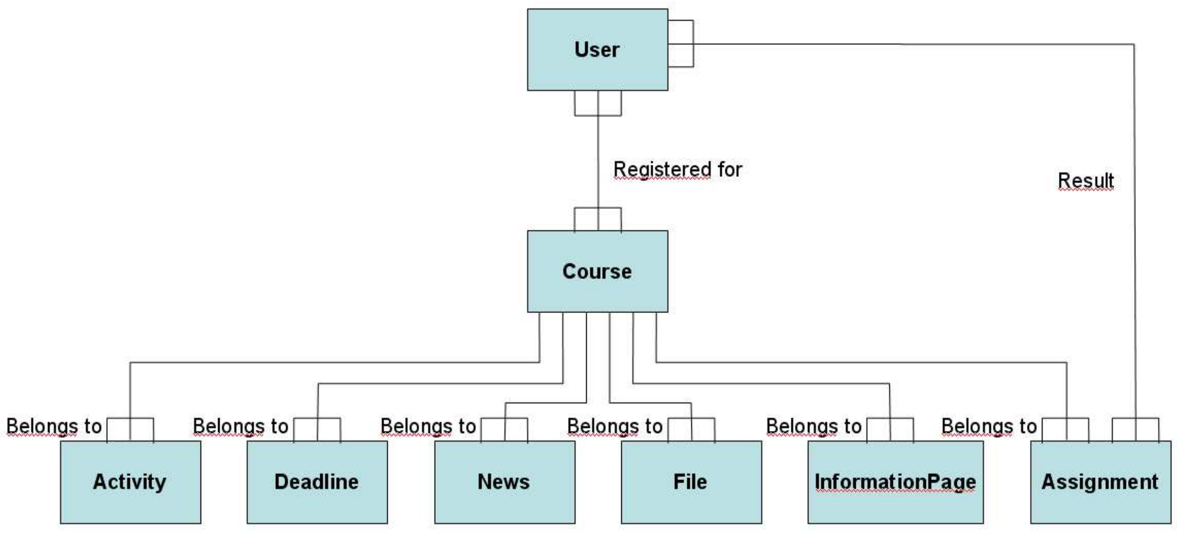

Figure 3: An entity-relationship model of the anticipated database structure.

Index

Index of Diagrams