Software Requirements Specification
Mark Management System

Version: 1.0

Organization:
University of Pretoria: Group 5

GitHub:

https://github.com/Lecton/301_Group5

Authors:
Mr Lecton Ramasila (10637291)
Mr Michael Johnston (12053300)
Mr Kgothatso Ngako (12236731)
Mr Rüdiger Roach (11004322)
Mr Martin Schoeman (10651994)
Mr Stephan Viljoen (11008408)
Mr Pule Legodi (29302732)

February 27, 2014

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to form a legally binding common ground between Jan Kroeze and the developers. This document will stipulate the functional as well as non-functional requirements of a system that will allow various contributors to collect, navigate and manipulate the marks of students studying at the university of Pretoria, as well as allow the students to view their marks.

1.2 Document conventions

Documentation formulation: LaTeX

Unified Modelling Language: version 2.0

2 Vision

The vision of the project is to create multiple interfaces, for use by multiple user levels, that communicate with a centralized database to perform recording, auditing, managing, manipulating, modifying and reporting tasks on mark results. A PDF or csv format document will then be attainable from these results. The system will be scalable and therefore allow multiple users to interface with the it at the same time. It will provide resource management utilities that will cater to the performance and reliability demands, ensuring the clients actions will be correctly reflected on the system.

3 Background

Lecturers, Tutors and Teaching assistants often find that recording and evaluating student assessments is very inconveniently and inefficiently done. Even more so if a recoding or multiple recordings need to be modified. Crucial information seemed to get lost along the way.
It was also a cumbersome task to generate reports for different needs as well as perform statistical analysis of the data to determine vital constraints such as the performance of the students in assignments.
With this in mind, a centralized schema is designed where multiple clients can perform data input, output, manipulations and reporting at the same time, and to do so on different interfaces such as on conventional computers and mobile phones running android as well as on laptops and tablets. These clients must communicate over a protected (encrypted) and synchronized framework to enable efficient, effective and secure transactions.

4 Architecture requirements

4.1 Access channel requirements

Browser Clients

- The browser clients that would be able to access the application from different platforms are as follows...
  - Google Chrome, FireFox Mozilla, Internet Explorer and Safari and any web browsers that run on conventional computers as well as tablets and mobile phones.
  - Because this access channel is architecture independent, it is cross platform.

Application client

- The Android clients that would be able to access the application are as follows...
  - Any device that runs Android can use the application.

SOAP

- This is the access channel segment that handles network communication with the database and passes requests to it, as well as accept responses from it.

Database

- The database that is used for data storage, management, manipulation will be instantiated using MySQL.

4.2 Quality requirements

Scalability

- The system is vastly scalable and should be able to handle a moderately large amount of users, before the performance starts deteriorating.
  - The application uses a centralized MySQL database, and all the users across the different platforms will connect to it and communicate with it.
  - The approximation of the maximum number of users that MySQL can handle before performance seizes to be optimal is about 150,000 users, this is due to the fact that the application uses this centralized MySQL database, and based on the MySQL documents it can handle about that amount, performing roughly a maximum of 4,294,967,295 connections to the database.
- The system provides resource management utilities that cater to the performance and reliability of the system. Examples of these are thread, object and connection pooling amongst others.

  - Making the performance independent to the number of number of users.

- The system supports clustering and load balancing.
- Connection to the Database is persistent, and connection can only be lost when the users decide to terminate the connection.

Authentication

- The users have to have an account in order access the application; this will provide the users with a username and password. These associated pieces of information are important as they will be required for the authentication process i.e. logging.
  - A user will have to provide his or her username as well as their associated password in order for them to be logged in and for a session to be created for them.
- The Session that is created when the user logs in with continues be checked for authentication throughout the time that the user is accessing the application. These checks could be done when the state of the application changes, when a request is sent or after a specified period of time.
  - If the authentication fails at any point, then the user will be kicked out of the system and will be forced to try to log in again.

Authorization

- Authorization tokens will be assigned to each user, irrespective of the sort of account that they may hold.
  - The tokens determine the users privileges.
  - They also determine what the user is allowed to read e.g view all the marks or just specific ones.
  - They determine whether a user can modify marks or any specific fields at all.
  - They determine whether a user can execute certain actions on the application.
- The privileges layers that the tokens offer can be elevated /promoted to a lower state or a higher state with 0 being the most privileged state as depicted in diagram.

Tokens	Privileges
n	More privileges then level n+1
1	Less privileges then level 0
0	Most privileged

Security

- The passwords that are assigned are assigned to each user are encrypted using a sophisticated encryption algorithm called the "message-digest algorithm" otherwise known as md5.
  - This algorithm prevents people who have access to the control (database) from being able to view the actual password and associate it with the user name and thus seize control of a user's account.
- When a user's fails to provide the correct login parameters 3 times in a row, the user account is suspended for 10 minutes.
  - The idea is to try and prevent bots (computer programs that try all the possible password combinations) from accessing an account, any users account.

Audit-ability

- Once the users have successfully logged into the application, they have the ability to perform certain tasks depending on their assigned privilege tokens.
- The users activity is logged in detail so that they can be referred to later if need be. Some of the activates that are logged are as follows...
  - Changes to fields or rows in the database.
  - Updates to fields or rows in the database.
  - Additions to any aspect of the database.
  - The execution of activities that directly or indirectly affect the database or another user.

Reliability

- There will be easy and fast access to the database.
- The system will save made changes if the system crashed while the user was working
- Users can still resume their sessions, if the device closed the application for some reason (e.g battery died).
- Errors will be handled and the system will show enough information about the errors.
- In a situation when a database crashed. There will be a backup database used to recover within a reasonable time.

Availability

- Online vs offline db connect
- Local database must be utilized when mobile interface is offline.
- Local database must synchronise with server once it is online.
- The system will be available in different platforms. Users can either access the system using mobile devices or by using any device with a web browser.
- The system can also be accessed anytime and anywhere.
- Inputs will be validated to prevent errors, before the execution of any input command.
- If the server is down the services or requests can be redirected to the backup server.

Flexibility

- The system will use SOAP based services, where changes to some of the parts of the service will not affect the design and some functionality of the system.
- The system allows integration with other systems.
- The system will take into account if the user loses internet connection or for whatever reason they cannot establish a connection to the server. The user can still be able to use the application but any transaction done while the system is disconnected will be cached until the connection is restored.
- Object Oriented Programming will be used to allow changes to some parts of the code(program). Those changes will not affect other objects. Reason for this would be to allow the system's design and implementations to be reusable for upgrading the system later on for new platforms.

Performance

- System should respond in timely fashion allowing markers to work at the quickest pace possible.

4.3 Integration requirements

LDAP

- The system is used for the authentication requirements of the application. This ranges from user login authentication to the permission based token assignment.

SOAP

- This protocol does the communication within the network. It achieves this by accepting request and sending responses to and from different interfaces and processing them accordingly.
  e.g.Receiving a request to view marks from an android application interface and passing the request to the database, then returning the result.

ADOBE API

- This API is used when generating a report. The result is then presented as a PDF using the fore mentioned API.

Mark API

- Must interface with CS website login details.
- Must use Soap interface through out all applications.
- Must Facilitate marking sessions.
- Marking sessions must coincide with practical sessions.
- Marking session durations must also be personalizable by the client.
- Marker(s) must be able to record marks during marking sessions.
- Marks must be stored in a database.
- Database must be exportable to a .csv file.
- Database must have lock/unlock capabilities that are also automated depending on whether are marking session is currently happening.
- Unlocked databases must allow marker(s) to alter marks.
- Altered marks must include reason for alteration.
- Marker(s) must be able to alter marks from previous marking session only with the permission/knowledge of the client.

Booking API
Must interface with Computer Science website booking system.

4.4 Architecture constraint

- Architecture: Waterfall Architecture
- Database technology: MySQL
- It must run using HTTPS requests and response events to connect to the database.
- Authentication: LDAP (LOGIN)
- Importable Data Format: CSV
- Exportable Data Format: CSV
- Interface: SOAP
- Server communication: Django
- Security/Authentication: LDAP

- Text encoding: UTF-8
- Client interface platforms
  - Android
  - Web browser
  - Cross Compatible OS support for Linux, Mac OS X and Windows
  - Any sensitive information stored must be stored in encrypted format.

5 Functional Requirements

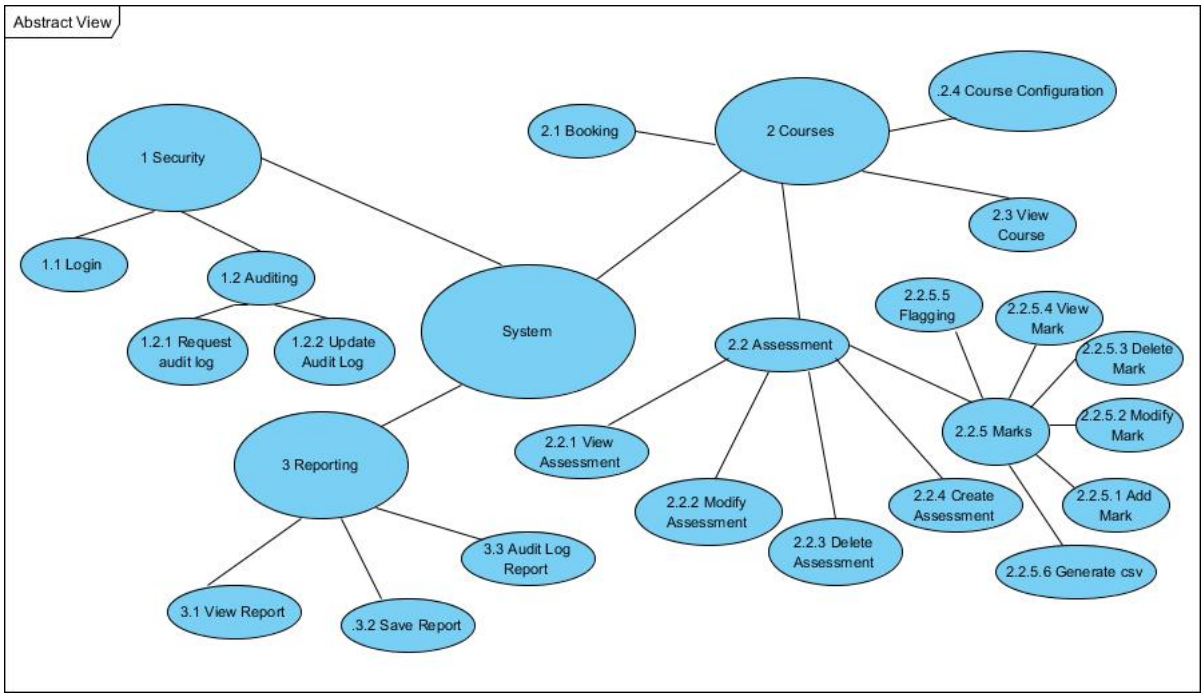
5.1 Introduction

This project ,when finalized, should act as a holistic marks management system. The system will have different privilege levels, when assigned to a level , users will have a restricted set of tools stipulated as follows:

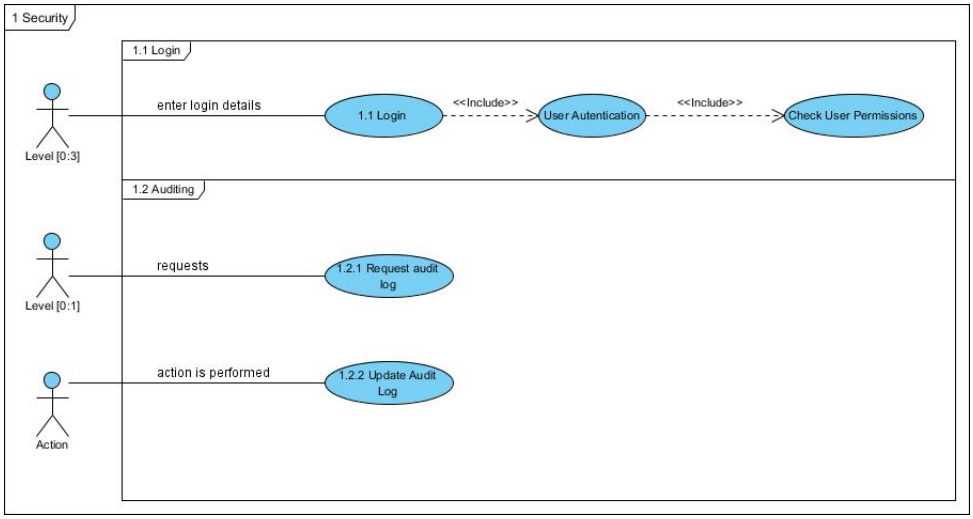
1. Level 0. This level can be seen as a level with "root" privileges. It is intended to only be assigned to either one or two people and will give access to all level 1, 2 ,3 and 4 privileges as well as be able to add and remove subjects from the system.
2. Level 1 users (such as Students) should be able to view their own marks.
3. Level 2 users (such as Teaching Assistants and Tutors) will be able to view any students marks that are in a session that the level 2 person is assigned to. This level will also give access to adding as well as modifying marks.
4. Level 3 users (such as lecturers) will be able to configure mark roll-ups, add assessment opportunities and draw reports from all of the marks of the subjects of which he is assigned to. This level will also have all of the privileges of level 1 as well as level 2 users.

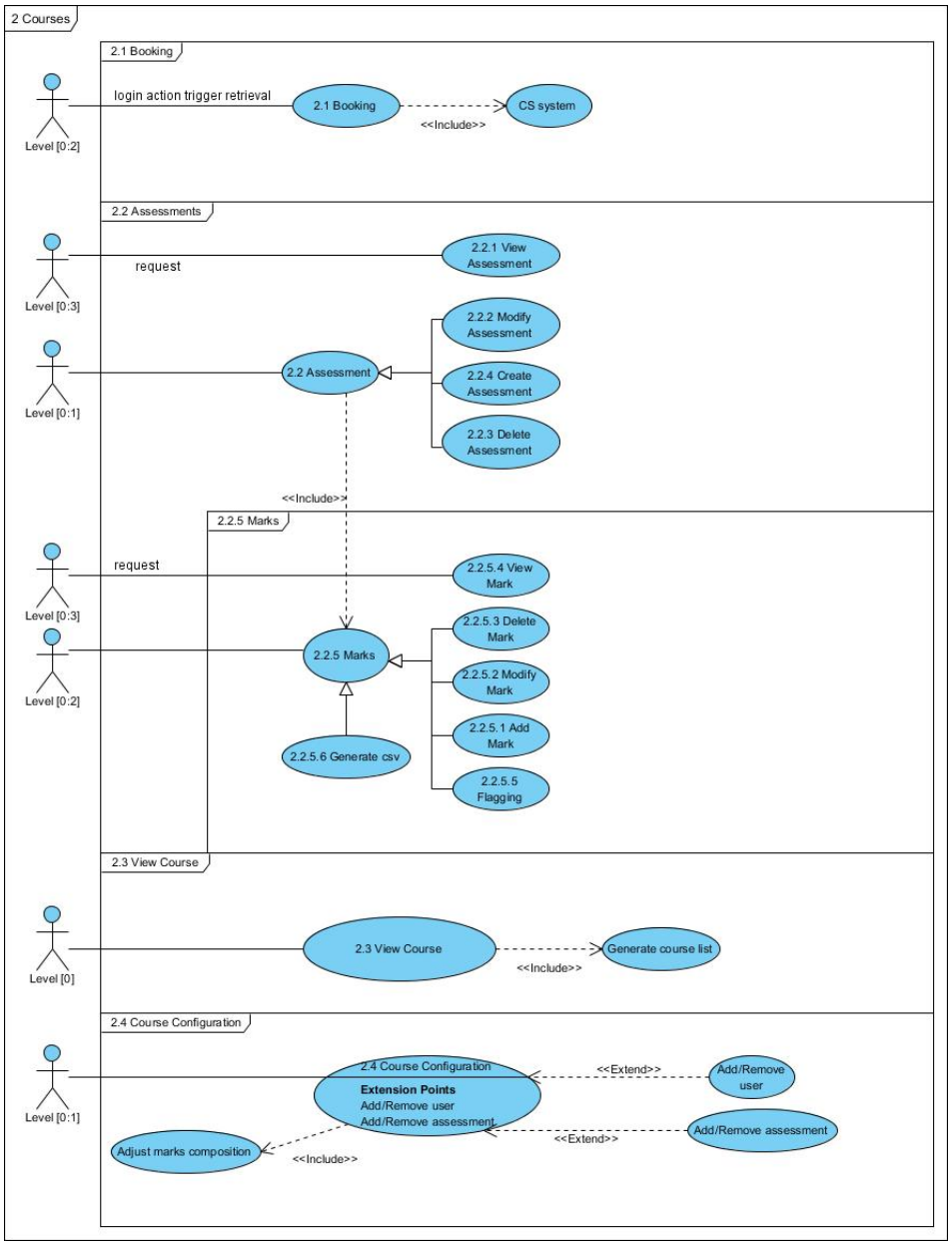
These tools will be available in a web based interface as well as an Android interface.

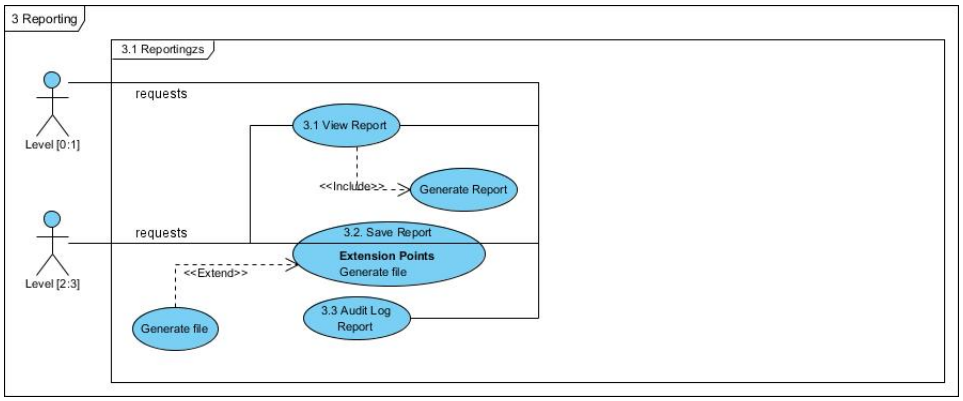
5.2 Scope and Limitations



5.3 Required functionality




2 Courses
2.1 Booking
login action trigger retrieval
2.1 Booking
CS system
<<Include>>
Level [0:2]
2.2 Assessments
request
2.2.1 View Assessment
Level [0:3]
2.2.2 Modify Assessment
2.2 Assessment
2.2.4 Create Assessment
2.2.3 Delete Assessment
Level [0:1]
<<Include>>
2.2.5 Marks
request
2.2.5.4 View Mark
Level [0:3]
2.2.5.3 Delete Mark
2.2.5 Marks
2.2.5.2 Modify Mark
Level [0:2]
2.2.5.1 Add Mark
2.2.5.6 Generate csv
2.2.5.5 Flagging
2.3 View Course
2.3 View Course
Generate course list
<<Include>>
Level [0]
2.4 Course Configuration
2.4 Course Configuration
<<Extend>>
Add/Remove user
Extension Points
Add/Remove user
Add/Remove assessment
Level [0:1]
Add/Remove assessment
<<Extend>>
Adjust marks composition
<<Include>>



5.4 Use case prioritization

Critical:

- Security, the system must be secure to protect the marks.
- login, must be able to differenciate between users.
- Auditing, must record changes made to the system for safety.
- Booking, must get the information from the cs website.
- Assessments, must state the different practicals, test ens.
- Marks, the system is made to collect marks.

Important:

- Course, the system is used by multiple courses.
- Course configuration, add and remove user.

Nice-To-Have:

- Reporting, different ways the view the information.
- View course, view extra information.

5.5 Use case/Services contracts

Use Case Name	Login
Reference	Section 1.1
Trigger	The user accesses the login page of the application.
Precondition	The user must enter the correct username and password.
Basic Path	1. The user enters his/her login details and submits. 2. The system verifies that the information provided is valid.
Alternative	In step 2 if the information provided is invalid, the system will return the user to the login page.
Postcondition	The home page will be displayed.

Use Case Name	Auditing
Reference	Section 1.2
Trigger	The user changes in formation on the system.
Precondition	The user must be logged in.
Basic Path	1. The user performs an action that changes information on the syetem. 2. The system logs the change and the user, with date and time.
Alternative	N/A
Postcondition	The information will be logged in the audit log.

Use Case Name	Booking
Reference	Section 2.1
Trigger	The user logs in.
Precondition	The system must be connected to the cs website.
Basic Path	The user enters the home page.
Alternative	The system can't access the cs website and an error message is displayed.
Postcondition	The system displays the booking information.

Use Case Name	Assessments
Reference	Section 2.2
Trigger	The user accesses an assessment.
Precondition	The assessment must be available.
Basic Path	1. The user selects an assessment like a practical, test ens.
Alternative	There is no assessment to be selected.
Postcondition	The assessment information will be displayed.

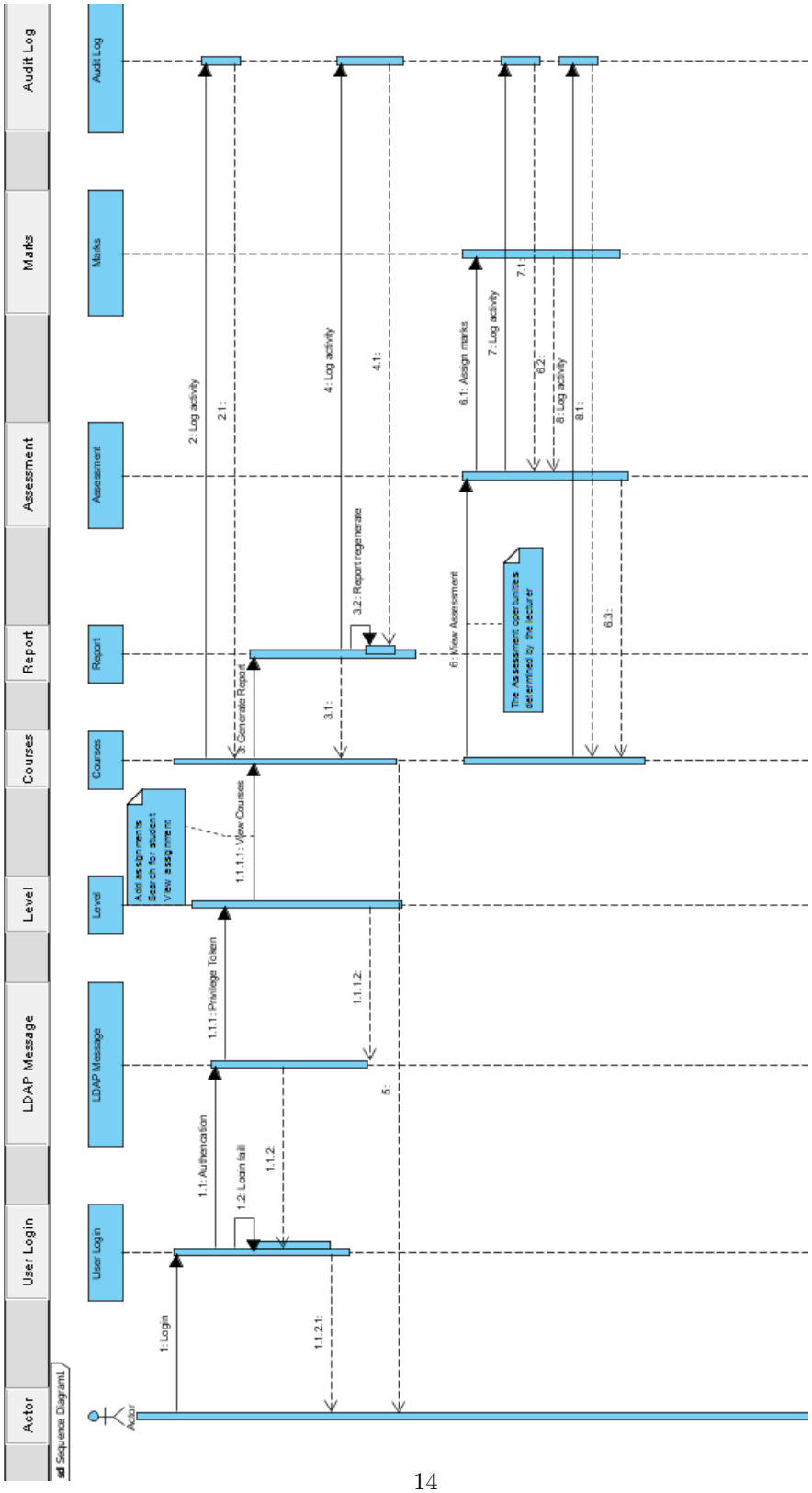
Use Case Name	Marks
Reference	Section 2.2.5
Trigger	The user selects the marks of an assessment.
Precondition	The user must have selected an assessment.
Basic Path	1. The user marks to display the options available for marks.
Alternative	There is no marks available
Postcondition	The options is given to CRUD the marks.

Use Case Name	View course
Reference	Section 2.3
Trigger	The user selects view course.
Precondition	The course must be available.
Basic Path	1. The user selects the course he wants to view.
Alternative	No courses are available.
Postcondition	The course information will be shown.

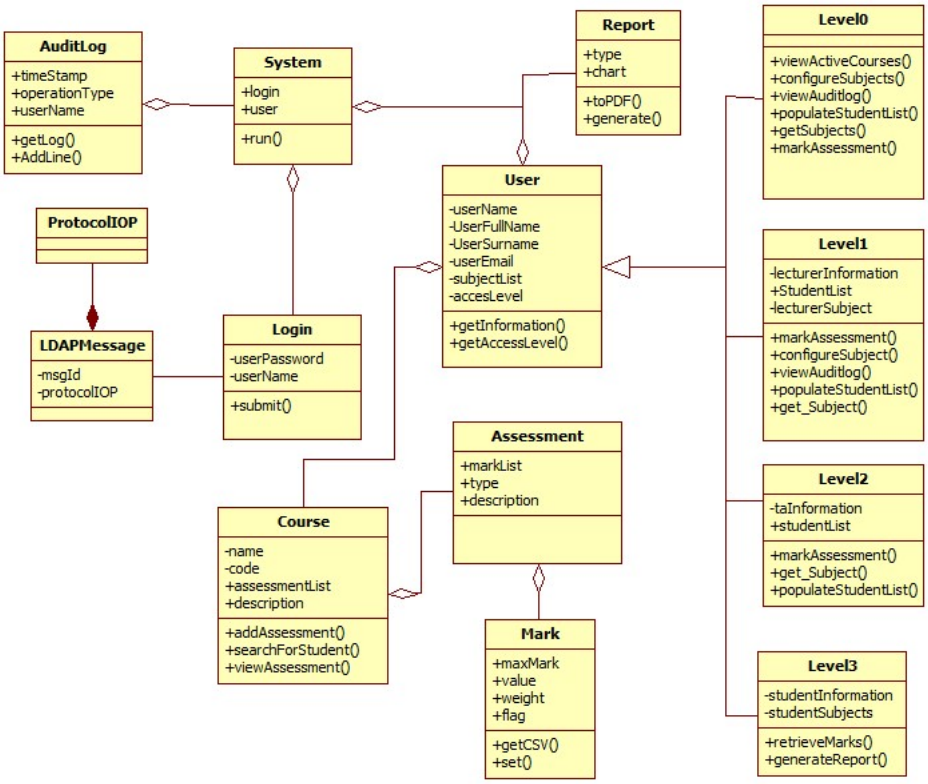
Use Case Name	Course configuration
Reference	Section 1.1
Trigger	The user selects course configuration.
Precondition	The user must have the right security priviliges.
Basic Path	1. The user selects course configurations. 2. The changes the configurations.
Alternative	The user does not have the right security priviliges.
Postcondition	The configurations for the course is changed.

Use Case Name	Reporting
Reference	Section 1.1
Trigger	The user selects reporting.
Precondition	The must have selected a course.
Basic Path	1. The user selects the type of the report he wants and generates it. 2. The user saves the report.
Alternative	The user only views the report.
Postcondition	The report is saved.

5.6 Process specifications



5.7 Domain Objects



6 Open Issues

7 Glossary

API Application Program Interface specifying how some software components should interact with each other.

CS Computer Science

CSV Comma-Separated Value file used to store tabular data in plain-text form.

db Database

Django Defined as "The Web framework for perfectionists with deadlines." Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.

HTTPS Hypertext Transfer Protocol Secure is a communications protocol for secure communication over a computer network, with especially wide deployment on the Internet.

LDAP Lightweight Directory Access Protocol is an application protocol for accessing and maintaining distributed directory information services over an Internet Protocol network.

md5 The MD5 message-digest algorithm is a widely used cryptographic hash function that produces a 128-bit hash value.

MySQL MySQL (Structured Query Language) is an open-source relational database management system.

OS Operating System

PDF Portable Document Format is a file format for capturing and sending electronic documents in exactly the intended format.

UML Unified Modelling Language is a general-purpose modelling language in the field of software engineering. It provides a set of graphic notation techniques to create visual models of object-oriented software-intensive systems.

UTF-8 A variation of UTF (Unicode Transformation Format) which is a variable-width encoding that can represent every character in the Unicode character set.

SOAP Simple Object Access Protocol is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks.

Level[M:N] Means from security level M up until security level N.