|  |
| --- |
| **Exam Scheduling System VIA University College** |

Eva Nikolaeva, 293164

Marton Pentek, 293649

Miladin Asenov, 240054

Sabina Elena Baghiu, 293119

Supervisors:

Steffen Vissing Andersen

Mona Wendel Andersen

**Software Technologies Engineering**

**1st Semester**

** [Date]**

**Table of content**

Contents

[Abstract iv](#_Toc27663613)

[1 Introduction 1](#_Toc27663614)

[2 Analysis 2](#_Toc27663615)

[2.1 Use Case Diagram 2](#_Toc27663616)

[2.1.1 Use Case Description 3](#_Toc27663617)

[2.1.2. Activity Diagram 5](#_Toc27663618)

[3 Requirements 6](#_Toc27663619)

[3.1 Functional Requirements 6](#_Toc27663620)

[3.2 Non-Functional Requirements 7](#_Toc27663621)

[4 Design 8](#_Toc27663622)

[5 Implementation 9](#_Toc27663623)

[6 Test 9](#_Toc27663624)

[6.1 Test Specifications 9](#_Toc27663625)

[7 Results and Discussion 9](#_Toc27663626)

[8 Conclusions 10](#_Toc27663627)

[9 Project future 11](#_Toc27663628)

[10 Sources of information 11](#_Toc27663629)

[11 Appendices 1](#_Toc27663630)

**List of figures and tables**

Optional

# Abstract

*This report serves the purpose of providing a detailed overview through the development stages of a Single User Exam Booking System, developed in Java. The system was created for VIA University College, that used excel sheets to book exams, but this was causing too many problems, due to human errors. The solution to the problem was found by analysing the requirements given by the university. Creating use case descriptions for the use cases, activity diagram, sequence diagram as well as a class diagram.*

*Based on the requirements the important classes were implemented with necessary functionality and tested successfully against the use cases. A functional GUI was designed with an objective of making the daily tasks of the user less complicated. The use cases were again tested against the program, successfully. The result is a functional program that fulfils all the requirements, making the everyday work of the user easier.*

# Introduction

The team needs to create a program for VIA University College that will correctly be able to show available dates for exams, choose a room, date and time for a given exam, implement the data needed (examiner data, available dates ) The program will be used by the secretary, so she needs to be able to get implement data, and if needed change anything implemented by the user at any given time. Once an exam is booked, there will be a calendar showing all the exams booked and all rooms that will be still available, then save all the data for further use. The program’s design should give the user a simpler way of booking an exam, see the exam period start and end of it and allow them to search for already booked exams.  
  
The secretary in VIA University College needs a single user booking system for the hotel. The lack of an exam booking system causes problems for choosing correct rooms and dates. Currently the university is using excel sheets where they write all exams that need to be booked. Using excel sheets is also very time-consuming since they need to go through it every morning to find the available rooms and figure out the equipment needed for a given exam. The needed booking system should be a single user system, since the program will be used only by the secretary. The program’s design should give the user a simpler way of booking an exam, see the exam period start and end of it and allow them to search for already booked exams.

Furthermore, the program should allow the user to change and delete booked exams, as well as changing the room needed. In addition, the program should minimize the number of mistakes that the university is currently struggling with. The following pages explains in depth how the program has been developed.

# Analysis

In the analysis part, a Diagram of the Use Cases was developed to underline the main capabilities of the system, together with a table that elaborates on the correlation between Use Case Diagram and the described above requirements.

## Use Case Diagram

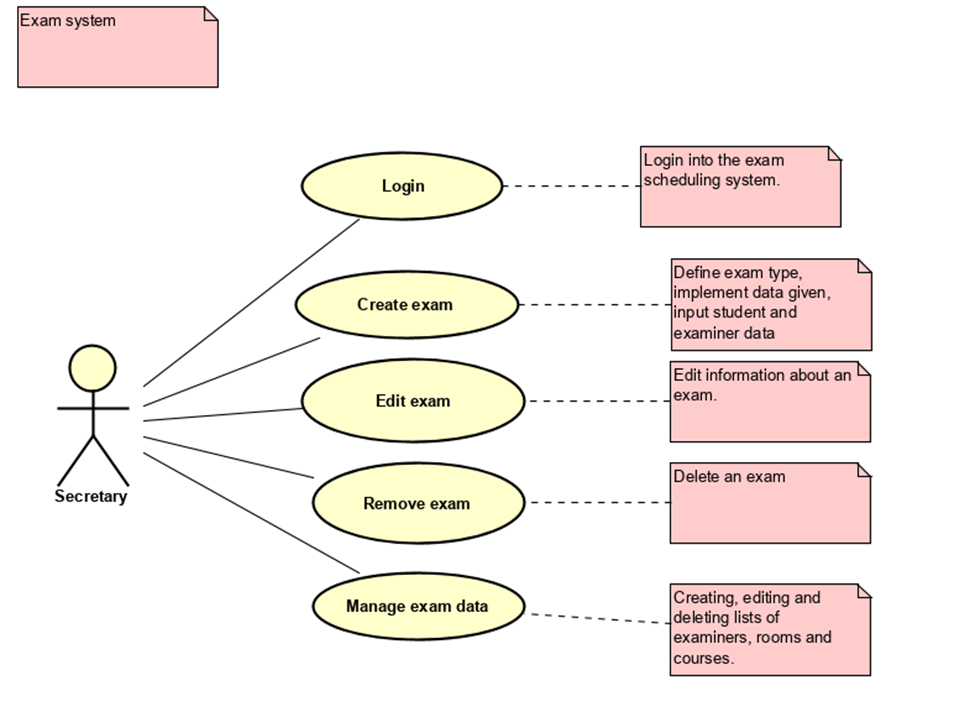


Figure 1: Use Case Diagram

The Use Case Diagram was developed to help visualize the main requirement “being able to show, add, change, edit and remove data from the lists. The user will be able to implement data and change it whenever needed. The diagram addresses the requirements 1, 2, 3, 4, 5, 6,7,8, 9, 10,11, 12, 13.

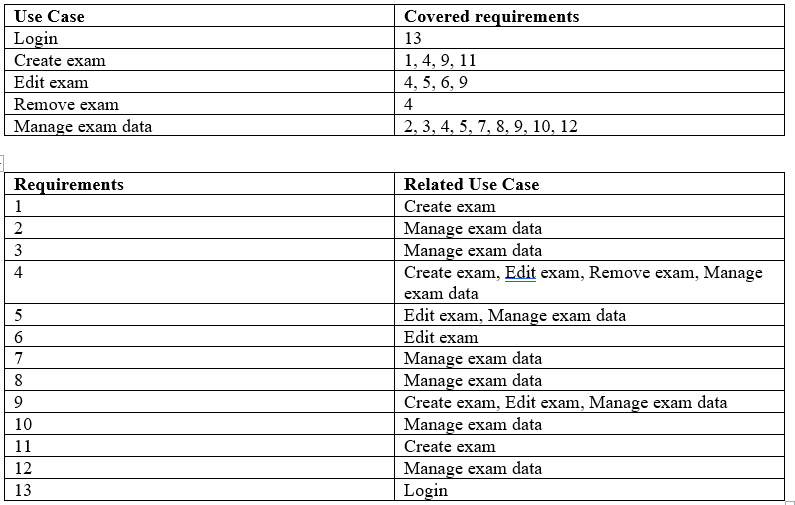


Figure 2: Use Case Requirements

The fact that some Use Cases address more than one requirement and vice versa means that the system is interrelated and works rather than being an array of 1 to 1 requirement – use case, because of that there will be a higher complexity when designing the system

## 2.1.1 Use Case Description

As a more in-depth analysis of the Use Cases, for each Use Case were developed Use Case Description to elaborate in a step by step manner. Below will be shown the Manage Data Use Case Description, refer to Appendix to see the other description.

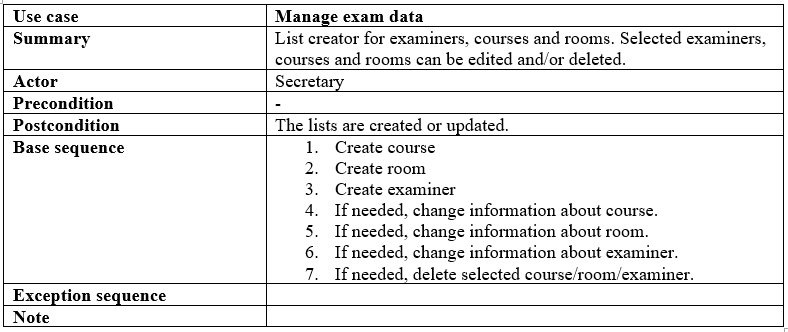


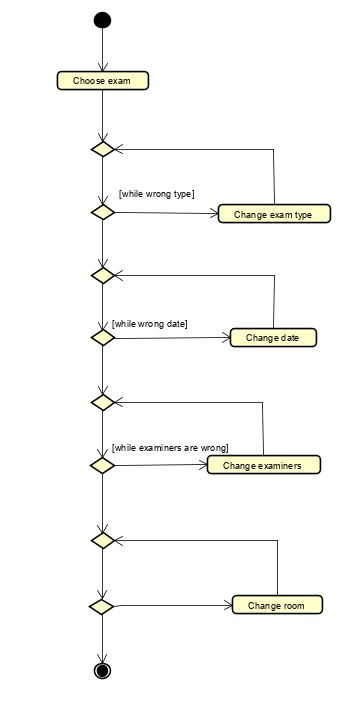
Figure 3: Use Case – Manage Exam Data Description

There are a few points that need to be addressed when analyzing the Use Case Description:

* First of all, the Secretary needs to be logged in to access the Use Case, meaning that the system provides a certain level of security;
* Another thing to be pointed out, is that the Use Case will not allow the user book an exam in a “forbidden” month;

## 2.1.2. Activity Diagram

As well as Use Case Description for each Use Case were developed Activity Diagram which provides a better visual understanding of the Use Case, below is presented an Activity Diagram for the Manage exam data Use Case, and the rest of them can be viewed in the Appendix



.

Figure 4: Activity Diagram – Manage Exam Data

Manage Exam Data Activity diagram provides a good overview of the entire process. Because Manage Exam Data Use Case has many options and possibilities and englobes different operation, it can be seen that the activity diagram is rather complex, having many turning points. The Secretary can chose from a range of option to edit the data about the exam, from adding a new exam and determening the right date, room and examiner, to deleting it from the system.

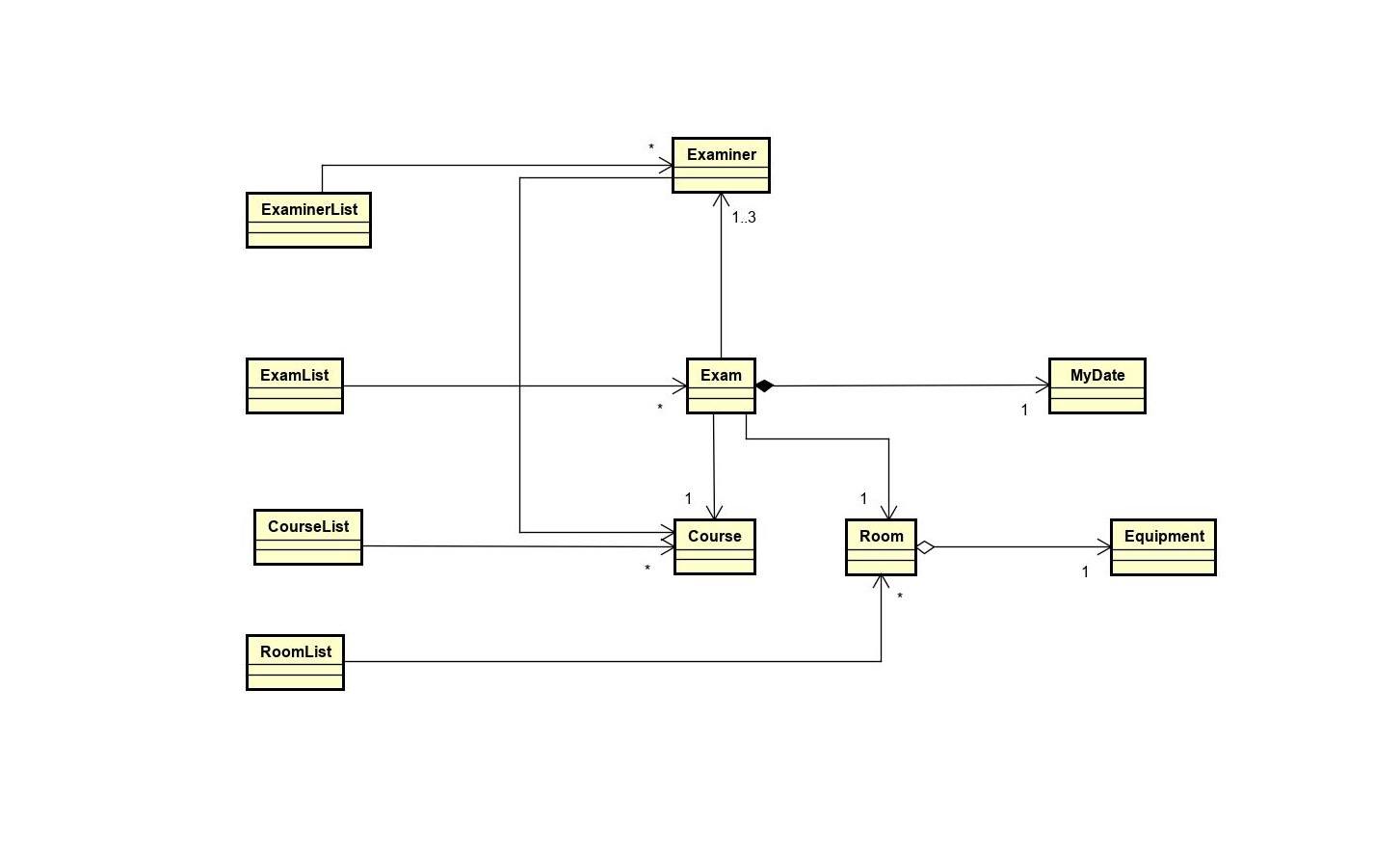


Figure 5: Class Diagram

The diagram shown in figure 5 illustrates how the program has been build up, using an object-oriented approach. The relationships between classes are also depicted here. Each class is connected to the “Exam” class, reason being, one of the most importantin this chapter, the structure of the system will be outlined through development of the Class Diagram and Sequence Diagram for the system. The focus will be on providing a blue print for the programming that will be used in the Implementation Phase.

The Class Diagram covers all the classes and methods that will be included in the system as well as the relationship between classes.

# Requirements

The requirements were made taking in consideration the described problem by the customer and imposed delimitations. The requirements have been divided into two parts non-functional and functional requirements and sorted based on their priority.

## Functional Requirements

1. As a secretary, I want to be able to add an exam at a specific date and room, in order to prevent overlapping.

2. As a secretary, I want to be able to make lists of exams, for students to know where and when the exam is held.

3. As a secretary, I want to have a list of examiners and which exams they are attending, in order to add them in the system.

4. As a secretary, I want to be able to add, edit and delete scheduled individual exams, in order to make, change and remove data and exams if necessary.

5. As a secretary, I want to be able to change and update the schedule at any given time, in order to prevent overlapping on dates and rooms.

6. As a secretary, I want to be able to check and edit if necessary, whether a class has more than one written exam in one day, in order to not exceed the limit of one written exam per day per student.

7. As a secretary, I want to have access to student information, in order to be able to arrange proper exams.

8. As a secretary, I want to have a list with equipment for each class, in order to make sure oral exams can take place there, e. g.: has HDMI.

9. As a secretary, I want to be able to add teacher names for the exams and change when necessary, in order to keep track of number of examiners attending.

10. As a secretary, I want to be able to edit the lists of examiners, courses and rooms, in order to keep the information updated.

11. As a secretary, I want to be able to arrange a 3-day SEP exam, as this is the standard procedure.

12. As a secretary, I want to be able to notify students for upcoming exams, room numbers and dates and send them a reminder message 3 days before the exam, in order to make sure they are aware of the schedule.

13. As a secretary, I want to be able to have a login, for security reasons.

## Non-Functional Requirements

1. User interface should be practical and easy to use.
2. The main programming language of the system must be JAVA;
3. The system must be a single user software;

A thing needs to be pointed out:

One of the most important requirements (ranked 1) is being able to add different exam types, rooms and dates with no possibility to overlap any of the data implemented. The user should not be able to put two or more different exams on the same date, room or time.

# Design

# Implementation

The purpose of the implementation section is to explain interesting code snippets. An idea is to explain the complete path through your system from UI to database etc.

Remember that your implementation must be consistent with your design (Larman 2004, chap.20).

Which standard libraries are used? How are design patterns implemented, etc.

Hint: Implement your code in a testable manner.

# Test

The purpose of the test section is to document the result of your testing; to verify if the content of the requirements section has been fulfilled. How is the system tested, which strategy has been used; e.g. White Box (Unit Test), Black Box, etc.

## Test Specifications

For functional requirements, test specifications must be listed. These test specifications can be described as soon as the functional requirements have been completed (Use Cases including descriptions).

IEEE can be used as a template for test specification (IEEE Computer Society 2008). VIA Library can give you access to this standard.

# Results and Discussion

In this chapter, the results of the software will be discussed, which of the requirements are ready for the actor to be used and which are not. From the requirements point of view, a requirement is ready to use when it performs without any issues.

1. The first requirement is that the software should provide a possibility of adding an exam at a specific date and room, as tested the software does that, it provides the possibility to add the necessary information, along with viewing the saved information – the software provides functionality for each aspect of this matter;
2. The secretary should be able to make a list of exams, so the students can access the information they need – the software provides a list, but it can only be viewed by the user (the secretary);
3. List of examiners and the exams they can attend in order to add them to the system – The secretary can create a list of examiners, and specify the subject they can attend, so the system works properly;
4. The ability to add, delete and edit scheduled exams implemented in the system - the save button in the “Edit” tab does not work, instead it deletes the exam;
5. Being able to change and update the schedule at any time – the software provides functionality for each aspect of this matter;
6. Able to check and edit if necessary, whether a class has more than one written exam in one day – The Edit function in the software does not work, but the user can still check the information saved before;
7. Access student information (number of students that need to attend to the exam) - the software provides functionality for each aspect of this matter;
8. List with equipment in every class - the software provides functionality for each aspect of this matter;
9. Be able to add teacher names for the exams and change when necessary, in order to keep track of number of examiners attending - the software provides functionality for each aspect of this matter;
10. Being able to edit the lists of examiners, courses and rooms - the software provides functionality for each aspect of this matter;
11. Being able to arrange a 3-day SEP exam – The software is not able to perform this action;
12. Notify students about future exams – The software is not able to perform this action;
13. Have a login for security reasons - the software provides functionality for each aspect of this matter;

# Conclusions

When summarizing everything that was done, beginning with the introduction to the project case and finish up with the testing, it can be observed that this project has achieved most of what it was supposed to.

As the requirements was defined, in the analysis phase, a more in-depth look to the structure of the project was payed. The use cases and description for them was developed providing a prime view of how the system should look and perform. Activity diagrams were developed for the use cases to observe how they should perform on a step-by-step manner.

In the design phase the system started to take even more shape, the sequence and class diagrams were developed providing a structure for the shape as well as a more detailed view of what the system will include and how it will work. In the implementation based on what was developed before a program was developed in JAVA programing language. All the phases above presented their specific challenges that had to be overcame for the system to exist.

In the last two chapters the performance of the system was assessed, which requirements were achieved and functional, and which were not. It is important to know what works and what does not to be able to anticipate different issues.

Overall, the system does perform the main features, and can be improved by a lot, but with the knowledge and team work, for the given time, we are proud of what we accomplished with this project work.

# Project future

When defining the project, the main aim was to provide a software system that will ease the process of booking an exam at VIA University College.

Due to the lack of time and knowledge, as well as efficiency, the system still needs to be worked upon, in order to become what it was supposed to. The functionalities that are there, are good and useful, the only issue is that they are not enough for the software to be used in a practical way.

In order for the project to become scalable, it is needed a design as well as code optimization and proof testing to ensure that any type of issues were addressed and solved. In the future, the software developed by the team has potential to be used daily with success.

# Sources of information

**Note: Use the standard reference method: Harvard Anglia. A very good reference tool is Mendeley** (Mendeley.com 2016), **ask VIA Library if you need help.**

Banger, D., 2014. A Basic Non-Functional Requirements Checklist « Thoughts from the Systems front line.... Available at: https://dalbanger.wordpress.com/2014/01/08/a-basic-non-functional-requirements-checklist/ [Accessed January 31, 2017].

Business Analyst Learnings, 2013. MoSCoW : Requirements Prioritization Technique — Business Analyst Learnings. , pp.1–5. Available at: https://businessanalystlearnings.com/ba-techniques/2013/3/5/moscow-technique-requirements-prioritization [Accessed January 31, 2017].

Dawson, C.W., 2009. *Projects in Computing and Information Systems*, Available at: http://www.sentimentaltoday.net/National\_Academy\_Press/0321263553.Addison.Wesley.Publishing.Company.Projects.in.Computing.and.Information.Systems.A.Students.Guide.Jun.2005.pdf.

Gamma, E. et al., 2002. *Design Patterns – Elements of Reusable Object-Oriented Software*, Available at: http://books.google.com/books?id=JPOaP7cyk6wC&pg=PA78&dq=intitle:Design+Patterns+Elements+of+Reusable+Object+Oriented+Software&hl=&cd=3&source=gbs\_api%5Cnpapers2://publication/uuid/944613AA-7124-44A4-B86F-C7B2123344F3.

IEEE Computer Society, 2008. *IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation*,

Larman, C., 2004. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*,

Mendeley.com, 2016. Homepage | Mendeley. Available at: https://www.mendeley.com/ [Accessed February 2, 2017].

YourCoach, S.M.A.R.T. goal setting | SMART | Coaching tools | YourCoach Gent. Available at: http://www.yourcoach.be/en/coaching-tools/smart-goal-setting.php [Accessed August 19, 2017].

# Appendices

The purpose of your appendices is to provide extra information to the expert reader. List the appendices in order of mention.

Examples of appendices

* Project Description
* User Guide
* Source code – source documentation
* Diagrams
* Data sheets
* Etc.

**Appendix A Project Description**

Insert the original Project Description here