Process Report

Exam Scheduling System

Audrius Sauciunas (293156)

Gabriela Melania Poteras (293122)

Pal Jambor (293158)

Supervisors:

Mona Wendel Andersen

Steffen Vissing Andersen

Software Technology Engineering

I. Semester

20/12/2019

Table of contents

1. Introduction

The main purpose of the process report is to present a detailed overview of the process the group went through from forming the group and receiving the case to finalizing the project and handing it in.

The focus is on how the group was formed, developed and collaborated throughout the first semester along with personal reflections from each member upon finishing the project.

The goal of the first-semester project is to develop an exam scheduling system for VIA University College in order to replace the manual process of scheduling exams by the secretary. The main requirement for the project is to create a single user system to schedule exams for Software Engineering students.

In this project, we are utilizing concepts and knowledge gained from three different courses: Study Skills for Engineering Students (SSE), Responsive Web Design (RWD) and Software Development with UML and Java (SDJ).

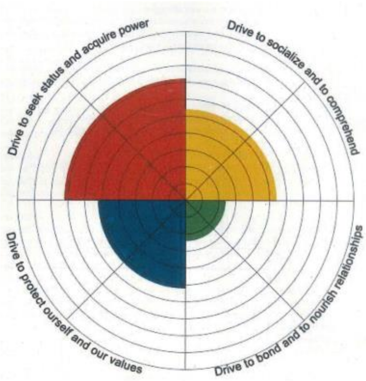
The group agreed to meet every Wednesday and during implementation weeks every workday from 8:20 until 16:00.

At the beginning of the semester, temporary groups were formed by the supervisors. Half a month later, based on our personal preferences and our requests being accepted, the final group was formed. In week 49 Samantha Nettesheim decided to leave the group. In week 50 the implementation period started.

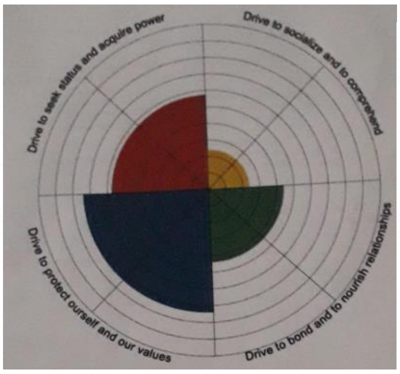
2. Group Description

Our first-semester project group is named MAP, an acronym for our names. Our group consists of three members from three different cultures, Hungary, Lithuania, and Romania. The class created individual Learning Style profiles utilizing the Felder-Soloman model during the second week of the semester (week 37) prior to the formation of the group. Furthermore, individual E-stimate profiles were created during week 45. Our group was created independently of any of the profiles mentioned before.

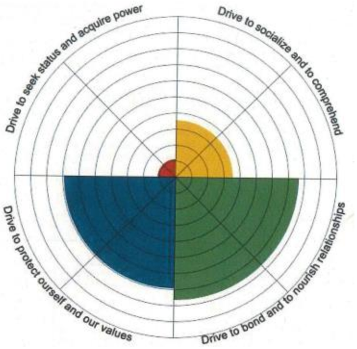
E-stimate profiles



Pal



Audrius



Melania

The individual E-stimate profiles we have gotten during one of our SSE classes help to understand the individuals' group behaviour and get to know the strengths and weaknesses of each member. It is useful to recognize the different communication behaviours and team roles that play an important role when working in a group environment (E-stimate International, 2019).

Audrius being mostly blue and somewhat red, Melania representing almost only green and blue and Pal representing the most red meant the group's colour ratio is more on the blue-green side. This required communication to be straight to the point, respectful and calm.

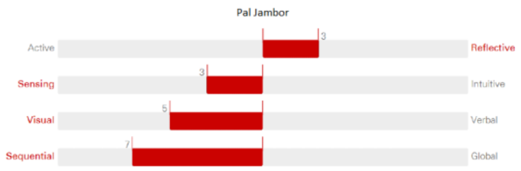
As far as group roles are concerned, Pal, having the highest red score, took up the role of coordinator. Also, red being significant resulted in being result- and goal-oriented rather than focusing on communication. The other members' high blue properties were complementary with it and balanced the group dynamics. Audrius and Melania shared high blue properties meaning they had similar roles in the group. They were both responsible for communication, giving feedback and quality control. With their blue properties being high, they tend to be more methodical and detail-oriented, which results in a precise and organized work manner. This helped the group recognize mistakes and details that otherwise would not have been noticed.

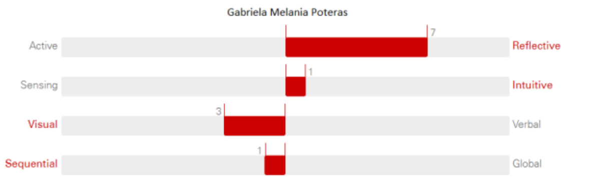
By having somewhat opposite profiles, conflicts and opposing views could have arisen between Pal and Melania, but instead led to better work and connection between them. The profiles are complementary and it allowed them to compensate for each other weaknesses and strengths, leading to better collaboration.

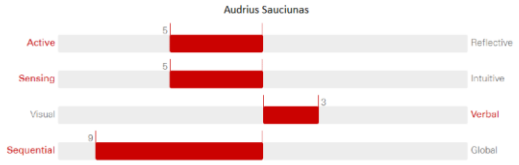
Learning Style profile

Every person is different, and each one of us takes in and processes the information in different ways. To better understand ourselves and to achieve optimal performance in the process of learning and working, Felder's & Soloman's Index Of Learning Styles could be a helpful asset worth consulting (Webtools.ncsu.edu, 2019).

After taking the test at one of our supervisor's suggestions during SSE, we received the following results:







After consulting the diagrams we came up with different conclusions and strategies to help divide team roles and tasks, thus administrating the group work properly. We tried to split the tasks accordingly but also equally.

Having both reflective and active learners, the group was equilibrated: Pal and Melania were the ones who first thought things through, while Audrius started working right away and kept the members from staying stuck in the thinking phase.

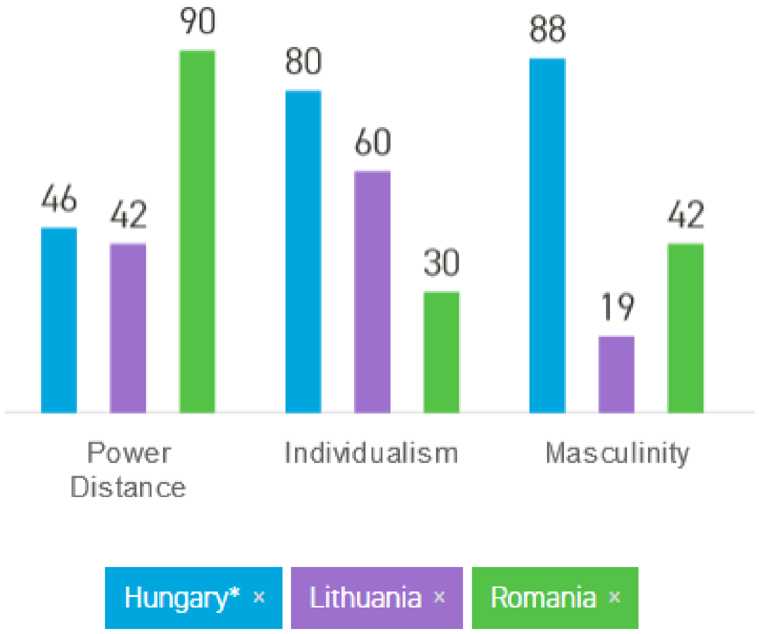
Pal and Audrius being more of a sensing type, they prefer facts and like to be more careful and practical with their work thus seeking to work more on the coding part. Melania, as an intuitive type, prefers to look for connections, relationships and the abstract parts of the project, taking preference on working on the reports.

As visuals, Pal and Melania decided on which diagrams should be included in the documentation and made decisions on the overall look of the pages and GUI, while Audrius took more interest in the wording and grammar.

Unfortunately, as we don't have any global learners, only sequential, there was always a risk of getting caught up in details and losing sight of the big picture. Knowing this from the beginning helped us keep an eye out and steer each other in the right direction.

Cultural influence

One of the requirements in SSE is to understand how workplace values are influenced by culture utilizing Hofstede's Six Dimensions of National Culture. According to Geert Hofstede, "National Culture cannot be changed, but you should understand and respect it."(Hofstede Insights, 2019). In order to understand how we are influenced by our cultures, we created a country comparison chart for the first three dimensions. Each group member's individual country profile reflects both similarities and differences. Our group analyzed the Hofstede indexes of Power distance, Individualism, and Masculinity. The Power Distance index for Hungary and Lithuania were relatively similar in contrast to the high score of Romania. According to Hofstede's theory: "People in societies exhibiting a large degree of Power Distance accept a hierarchical order in which everybody has a place, and which needs no further justification."(Hofstede Insights, 2019). The Individualism index in our group is descending from Hungary to Romania as it covers all three possible levels from high to low. The Index of Masculinity in our group is fairly dominated by Hungary, whereas Lithuania scores the lowest.



Pal (Hungary): In some aspects, I represent my country's scores well, but there are some areas I don't agree with anymore. With the Power Distance score being in the middle, hierarchy is not important for me but being independent is a key in my personal life. Also, I consider myself a very individualistic person but I don't believe that ”relationships are contracts based on mutual advantage”(Hofstede Insights, 2019). I believe relationships have to be genuine and independent of any ulterior motives. The masculinity score did represent me well when I came to Denmark. Now, after four months of living here, I learned that although competition is important to a certain extent that is not something that one should be driven by and that score doesn't match me anymore.

Audrius (Lithuania): In my opinion, what is shown in the diagram only represents me half truthfully. While the Power Distance index for Lithuania isn't considered to be high, I still believe that in my case it is less than it is shown. I do not believe a hierarchy is important for me at all and everyone should be accounted for as an equal. While I do agree that Lithuanians are fairly individualistic, it does not reflect me truthfully, I believe that in our group there is only a "we", and everyone is supposed to help each other achieve on agreed goals. Lithuania is considered to be a very feminine country but I don't agree that it reflects on me too well. In my opinion, I am somewhere in between. I am competitive to a certain extent, but I also do care about the well-being of others and the quality of the work environment.

Melania (Romania): Looking back, I can say I mostly agree with the diagram, several aspects being quite easily seen in me. While I strive to consider everyone equal, it's easy for me to accept a hierarchy and set a distance between me and those that are considered "higher-ups", Romania's high power distance score reflects on me correctly. After living in Denmark for a while where the culture is very different, I got to lessen the distance and gain a new view but no matter how many changes I go though, in the end, the core cannot be changed. I agree with the individualism score as well. While, as a person, I prefer to work alone, being part of a collective and having bonds with the members is important, after a long period of time working together getting to think of the group as a secondary family. As for the masculinity aspect, I would place myself in the middle, trying to achieve a balance in it.

Group Contract

Group Name: **MAP** **Group 2** Date: **08/12/2019**

These are the terms of group conduct and cooperation that we agree on as a team.

Participation: We agree that a successful team requires a high level of interaction and collaboration amongst all team members. A group is a cohesive unit and we will strive to maintain a cohesive unit to the best of our ability. We agree to allow group members the creativity and means necessary to be productive. All team members should participate equally despite strengths and weaknesses.

Members should attend classes and meetings consistently and inform group members of intended absences or late arrival. An effective team needs to be organized and prepared for each meeting. If a team member fails to prepare effectively for a meeting, the team suffers as team progress is dependent on meeting deadlines. If a team member experiences trouble with an assigned task, the team member should inform the other team members and ask for assistance 7 days prior to the deadline agreed to by either the team members or the official deadline. We will address what prohibited the member from meeting expectations and reassign tasks accordingly, only if necessary.

If the team member misses two consecutive meetings without adequate reasons (I.e. sickness, unexpected life events) the person will get a warning. If a team member misses a meeting, it is the team members responsibility to review the pertinent material prior to the next session. Any team member that misses a meeting without advanced notice, loses their right to the work created during that session. Each work session duration will be 45-50 minutes followed by 10-minute breaks.

Communication: We agree to open communication without judgements. We agree to check our messages daily and respond as soon as possible, to at least one team member (no later than 24 hours). We agree to try to understand each other's communication styles and ask for clarification if needed. We will utilize Wisinski's A-E-I-O-U model from SSE to communicate concerns. Each group member should address their concerns as early as possible, allowing the other group members time to understand and discuss concerns followed by resolution. Members are expected to be open and honest during meetings.

Meetings: We agree to meet every Wednesday from 9:00-16:00.
If someone is going to be late, she/he should notify the group as soon as it is possible, in the case of being late more than 15 minutes.
During the project period (week 50-51), all members agree to be at the university between 8:20 and 16:00 on weekdays.
Extra work sessions during the tuition period will be scheduled for two days in advance after classes.

Conduct: We agree not to use social media, gaming, listen to music loudly and to put our phones on silent during work. The members must be serious and not fool around during the work sessions.

Conflict: We agree to manage conflict using the 4R approach, Reasons, Reactions, Results, Resolutions. We agree to address conflicts through collaboration and compromise for substantive and procedural conflict. Any unresolved substantive or procedural conflicts will be resolved through arbitration by the class teacher or assistant.

Affective conflict, such as constant complaining and competition is destructive to the team and will not be tolerated. Members agree to understand that everyone has their own set of problems and that each member's problems should not affect the group's ability to achieve the desired goals and deadlines. Members understand that conflict is a natural occurrence in groups and that conflict is to be resolved through compromise and collaboration, not through avoidance or accommodation.

Group member's name	Student number	Signature
Gabriela Melania Poteras	293122	
Pál Jámbor	293158	Jámbor Pál
Audrius Sauciunas	293156	

3. Project Initiation

Before the formation of final first semester groups, the project was initiated by supervisors through a case presentation. The presentation was held in the middle of September by Mona Wendel Andersen (week 38). After the presentation, specific questions could be asked about the requirements and the scope of the project to understand what is required and expected from the groups. This meant that the information still had to be filtered and analyzed. At the time, all students were in temporary groups of 6, which meant Pal and Audrius were in the same group, while Melania was in a different one.

Half a month later, the final groups were created based on personal choice, which also included Samantha Nettesheim, since the aim of formation was to make groups of four. The only requirement was not to have more than two of the same nationalities.

The project required choosing online tools for the group to make collaboration easier. First, we used Microsoft Teams, a chat-based workspace in Office 365 (Microsoft 365 Blog, 2019), for creating and editing documents. Coming closer to the beginning of implementation weeks it was decided to switch to Google Drive, a file storage and synchronization service developed by Google (SearchMobileComputing, 2019), as it provided more options for organization and easier access to the group's materials. Secondly, we started to use Git for coding, a version control system, as it allowed members to work on coding together and task division (Atlassian, 2019). Lastly, itsLearning and Studienet were also used to check on the templates and guides in order to make proper documentation.

As far as communication is concerned, the group only had a shared conversation on Facebook at the end of the project period, from week 49, and did not have any form of team communication online until that point, except Teams' chat, which was not often checked. The conversations until that point were handled privately between members or in person. Looking back, we realize that the lack of clear group communication was one of the reasons for conflicts and misunderstandings.

The very first task after the formation of final groups was to merge our previously written project descriptions, which were made while being in temporary groups, and making a group contract. The group used this version of the project description throughout the rest of the project work. The contract made upon formation helped us set up guidelines, but in week 49 it was updated, right before the start of implementation.

While being in a group of four we worked on both the analysis and the design. However, Samantha decided to leave the group in week 49, because of arguments and conflicts, which she felt as if it couldn't be solved anymore. Afterwards working in a group of three we reprioritized and adjusted the project as well as the workload to be fit for three people.

Concerning time management, throughout the semester the group set its own deadlines so that work would be optimal and no tasks would be rushed. Our group met regularly every Wednesday as required for the Semester Project (SEP) as well as routinely throughout the week based on current assignments, over a 13 week period. This approach helped the team a lot since it reduced stress about both the hand-ins and assignments. While working on the project it allowed time for extra time and review, ensuring that we were staying on track.

As tuition finished, our team was motivated to work on the project in the following two week period (week 50-51) and met daily for eight hours every workday. Despite our efforts and planning, we still had issues with time management during the implementation week and not all of the requirements were completed.

4. Project Description

As the four members of the group were from three different groups at the time of formation, we had three versions of project descriptions. We felt as if it would be best to merge them instead of picking a single one, so all of the best parts could come together.

Although the making of the project description was overall challenging, in the process of the project work we read through the description and tried working accordingly throughout the execution. In the end, it proved to be useful as it constantly reminded us of the aim of our project. The main difficulty appeared to be the identification of the proper way of execution. The group has never had any project experience prior to the start of the semester, therefore it was difficult at the beginning for us to comprehend the correct way of execution. Despite that, through perseverance and constant feedback, we managed to grasp what is the proper and expected way of project completion. Unfortunately, the goal set at the beginning proved to not be realistic with our current skills and we felt we aimed too high. Thus, our goal was achieved only partially. Despite that, with the help of our peers, teachers, and presentations we got more confident in our development skills and learned a lot during this process.

5. Project Execution

In the project execution, all the knowledge gained from courses during tuition had to be used in order to achieve a final product.

For our first semester project, it was decided that the Waterfall method should be used, where “there is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks”. This approach doesn't allow flexibility as the progress flows in one direction, in contrast to the iterative approach, which allows developing a system in repeated cycles and in smaller sections (Tutorialspoint.com, 2019).

In spite of our efforts, the group had to go back to earlier stages and make modifications in order to achieve a functional system, resulting in not respecting the Waterfall method entirely.

Some of the requirements had to be edited and rearranged as well as the corresponding use case descriptions and activity diagrams.

As a whole, we found the Waterfall approach too restrictive and we would not use it again. The overall results of the project were not satisfactory, there being many issues with the pre-planning phase, resulting in making the program with the implementation of all our requirements less successful. This resulted in the risk of motivation loss, as it caused stress but with the uplift of each other, we managed to get through it.

Despite not following the approach completely, it provided a great structure and some parts of the method would also be useful in future projects. With the combination of the two methods, it would be possible to create an ideal method of working on future projects (Powell-Morse, 2019).

6. Personal Reflections

Audrius Sauciunas: Before I came to Denmark I always wondered what is waiting for me in the future during my studies. Now as the semester is closing in and as I reflect I feel overwhelmed by all that has happened.

During my school years, I have never had any experience with project work, as the education I've had barely focused on a team environment. Instead, it was all about memorizing various information and applying it in tests and examinations. As my school years were coming to an end I started researching for a place to study, I always dreamed of studying abroad but I never thought I would actually do it. Since my previous education did not bother focusing on teamwork, I decided that I would like to focus more on teamwork based studies instead.

At the beginning of the semester, we were divided into temporary groups of six people, which I believe helped me a lot to get a sense of what teamwork is. I got used to working in that particular group and it was hard for me to decide who I would like to be in a team with. But after a few months, the time had come when we had to create a group of four people. It was a stressful experience as I was hoping to find a group where people were motivated to learn and improve. Now as I reflect I believe that I should have not based my circle around the people I was the best friends with but with the people that I felt like I would be able to work the best.

During the times we worked as a group the road was bumpy, to say the least. There were many times where we would have multiple heated conflicts about things that were meaningless for the project we were working on. It was a very stressful and toxic environment and right before the tuition ended our group member Samantha left the group. It wasn't a very pleasant moment, because it felt as if we were not the same group as we were at the beginning and it did not reflect on our values, but we did not give up and were looking forward to a great head start for our semester project. The beginning of it was hard, we did not have any plans on how we will do things and we only had a very minimal schedule. I believe that my biggest mistake was that I did not focus on making the base structure of our system first before implementing the functionality. Having a little bit of everything is equivalent to having nothing. Because of this mistake, I was extremely stressed because nothing was working and as a result of a lot of the work we made during the first three days and some of the parts we made during tuition, had to be completely remade from scratch. As time progressed I started to feel as if I was overburning from all the stress I was going through and the feeling of stress coming from the group members did not help at all either. But I believe that it is normal for everyone to feel stressed during times like this and it is not always a bad thing as it can also work as a motivation to work towards the finish. Even though we went through many struggles as a group and as individuals at the end with perseverance we managed to find our way through this bumpy road.

Now, as I reflect back, I learned quite a few lessons for the future and have a clear idea of what group work is. Having been through a negative and fairly toxic environment, now I know what a healthy environment is. Even though the start was unpleasant I feel like SEP1 was a great start of my future career as every lesson I learned will prepare me for the industry, because I know that even at a professional level I will not always have the perfect team.

Gabriela Melania Poteras: Coming to Denmark was a huge change for me. A new country, a new culture, a new schooling system. I researched, read and talked to a lot of people in order to prepare myself for what is to come, but in the end, no piece of text can encompass the experience of being there. To say I wasn’t scared and anxious would be a lie. But to say I wasn’t excited and looking forward to it would be an even bigger lie.

Before coming here, I did not have any experience of working in a group or a project whatsoever. Accommodating to a group environment was no easy task for me. It required a lot of work and, effort but I believe I managed to achieve a sense of being able to work in a group and being more comfortable in it even though it is not at the level I would have expected to be after a semester. Sometimes I get the feeling I lost myself somewhere in the process and forgot to let myself shine as bright at the others, not to dim my light in order to make space for theirs. Looking back and considering this, I am hoping better is to come and I will work harder in the next semester to advance on the path to the full potential.

Being with my team members made me realize the different perspectives everyone has regarding their work and the way one approaches it. I am grateful for that and their hard work and dedication. There were fights and conflicts, some that unfortunately led to straining the bonds between us followed by the departing of Samantha from the group. This caused me a lot of distress to which I am not sure how well I responded.

Working on a project where we would directly apply the theory we learned and put to practice our knowledge was another new thing for me. The idea seemed attractive to me at the beginning and after finishing the semester I can say it still is. Being presented with a problem and being told “you are at point A and need to figure it by yourself how to get to point B” is both exciting and frustrating but this is what makes you truly learn.

The project was assigned to us by our “client”. Unfortunately, I haven’t felt much passion about it, but I still worked hard on it to fulfil all the requirements and make it good as it’s a great learning experience and I am aware that after I will finish school and start working I will be faced with other projects I might not passionate me but still need to be seen through the end.

In the making, we hit multiple walls and faced many problems. One main cause was us aiming too high at the beginning without taking into proper consideration of our abilities and not being aware enough of what and how. When not having a good base, the rest will be unstable. Now I can say that in the future I will better assess the situation and problem with my current skills and with the ones I will gain over the course of the semester.

Looking back, I realize I could have done much more this semester, from personal life to school life and the project, and I feel regretful for that. I know I haven’t met the expectation I had, and the ones others had on me, but I still believe the time wasn’t wasted. Unfortunately, I was slow in different aspects, but slow progress is still progress.

Pal Jambor: My name is Pál Jámbor and I am from Hungary. As I knew the educational system in Denmark is more practical, at the end of my high school studies with the support of my family I made the decision to come and study Software Engineering at VIA. I have always wanted to study abroad and I finally got the chance to do so. Since I didn't have any previous experience concerning project work I was scared and excited at the same time. I was uncertain if I would be able to take the responsibility of making a project every semester but after group formation, I had more confidence in myself.

When conflicts arose in the group I was confident that we would be able to resolve them in a calm and controlled manner. Unfortunately, this did not work out and Samantha left the group. This period of time caused a lot of stress and staggered me about the success of the project, especially because I felt I was blamed wrongly by Samantha when all I wanted was to create a peaceful and calm environment and because this happened right before implementation weeks.

Luckily after the situation has been dealt with I felt more confident about the project and was ready to give it my all. During the two weeks of implementation, more problems arose concerning the project and especially the pre-planning, which was already made during tuition. The plans we made for our program seemed to be not working well and it needed reconsideration. Now I realize that pre-planning is a huge part of the process and should not be rushed nor not thought through carefully. As I was the one who did most of the coding of the program but with the help of my teammates, the program seemed to me as a huge challenge. Despite the previous plans at certain times it felt it was trial and error, which I know actually was not. It seemed to me as if we were behind and not making enough progress. Also, the pressure of the deadline made these two weeks one of the most stressful periods of this year for me.

Since I am the type of person who hardly gives up, it was very hard for me not to work on the project every hour of the day, as I wanted to do it to the best of my abilities. I often worked after my team members went home until late at night and I also was working on the project during the weekend. I realize that this approach is not the best, as it is very straining but I know, if I had not worked this way, I would have felt I didn't do everything I could in order to get the best results. Despite how hard these two weeks were, I feel I learned a lot about group work and learned how to apply theoretical knowledge in a more practical environment.

7. Supervision

During the course of the project, the group was supported by two supervisors: Mona Wendel Andersen and Steffen Vissing Andersen.

During tuition, we felt we would have liked more supervision and feedback regarding documentation. From the group's point of view, it would have been better to set up supervisor meetings earlier in the semester, which could have resulted in making fewer mistakes during the pre-planning process.

As conflicts were present in our group, reaching out to Mona was a good decision, her's and the counsellor's help solved the problems quickly and efficiently. Their assistance allowed the group to continue with a peaceful and healthy work environment, which was crucial for the success of the project.

The group set up a supervisor meeting with Steffen on the 13th December since we ran into several problems during the implementation of our program. This meeting helped us get clear answers about certain parts of the code, realize some of our mistakes and also helped us define goals in our project, as at this point we were doubtful about the entire success.

Another meeting was set up on the 16th and 18th, which both provided further guidance in the implementation of the system and cleared our doubts about the structure and content of the documentation.

A final meeting was set up on the 19th, the last meeting before the deadline, in order to finalize the project and put every file together to be ready for hand-in for the following day.

Aside from the scheduled meetings, the group sought out the teachers each time it was considered necessary to which they responded positively and provided the needed help. Despite their efforts certain times it was still necessary to go back and ask about the same problem, as we did not always manage to grasp their suggestions.

Overall, the group used the supervisors during the implementation weeks regularly to gain feedback on the progress and get answers to specific questions concerning documentation and coding. Luckily, they both were very helpful and the communication was great as well.

8. Conclusion

Time management issues were present throughout the period of the project, but by having regular meetings and by making deadlines in our group, the problem was mostly manageable and unnecessary stress was avoided. This approach of working behaviour was effective during the semester and helped us stay on track, allowing extra time for fixing errors. Considering the results, this method is something we will definitely continue using in future group work.

In the beginning, communication between members has been handled poorly. The conversations were held privately and excluding members at times as well as having information not being transmitted properly. This resulted in conflicts that severed the connection between the members. The collaboration between members got better after the unanimous decision to focus more on communication and setting a group chat. In the future, communicating more in an honest and transparent manner in the group without the exclusion of any members will be a priority from the beginning as it will lead to better teamwork and reduce the stress and negativity. All information should also be easily accessible to everyone concerning collaboration.

As the quit of Samantha Nettesheim made the workload on the team bigger, the project required more time and motivation, which proved to be challenging at times. Someone leaving the group slows down the process and results in time loss as well as the decrease of the collective's moral. The whole process takes a toll on the members and overall has a negative impact on both work and teamwork. In the future, problems that arise should be handled as soon as possible, preventing them to escalate and lead to more conflicts.

Having a lot of problems concerning the implementation, more careful pre-planning is required in future work as well as the scope of the project has to be analysed in much more detail with regard to the gained knowledge before starting the implementation. We learned that although it seemed tedious at times, planning is the most important aspect of writing a program and it should be very well thought through. We also need to better consider our own abilities and make realistic goals of what can be achieved over the course of the project when making pre-plans.

Over the course of the semester, we learnt that communication in a group environment, as well as time management, are key points towards being as successful as possible. As long as these are handled properly, most problems can be avoided. Moreover, splitting tasks and responsibilities from the beginning is a great idea, so no member feels like someone is doing more than others.

References:

E-stimate International. (2019). *e-stimate - professional personal profiles and personality tests*. [online] Available at: https://www.e-stimate.com [Accessed 27 Nov. 2019].

Webtools.ncsu.edu. (2019). *Index of Learning Styles Questionnaire*. [online] Available at: https://www.webtools.ncsu.edu/learningstyles/ [Accessed 5 Dec. 2019].

Hofstede Insights. (2019). *Compare countries - Hofstede Insights*. [online] Available at: https://www.hofstede-insights.com/product/compare-countries/ [Accessed 6 Dec. 2019].

Microsoft 365 Blog. (2019). *Introducing Microsoft Teams—the chat-based workspace in Office 365 - Microsoft 365 Blog*. [online] Available at: https://www.microsoft.com/en-us/microsoft-365/blog/2016/11/02/introducing-microsoft-teams-the-chat-based-workspace-in-office-365/ [Accessed 7 Dec. 2019].

SearchMobileComputing. (2019). *What is Google Drive? - Definition from WhatIs.com*. [online] Available at: https://searchmobilecomputing.techtarget.com/definition/Google-Drive [Accessed 7 Dec. 2019].

Atlassian. (2019). *What is Git: become a pro at Git with this guide | Atlassian Git Tutorial*. [online] Available at: https://www.atlassian.com/git/tutorials/what-is-git [Accessed 7 Dec. 2019].

Felder, R. and Soloman, B. (2019). Learning Styles And Strategies. [online] Engr.ncsu.edu. Available at: https://www.engr.ncsu.edu/wp-content/uploads/drive/1WPAfj3j5o5OuJMiHorJ-lv6fON1C8kCN/styles.pdf [Accessed 9 Dec. 2019].

Tutorialspoint.com. (2019). *SDLC - Waterfall Model - Tutorialspoint*. [online] Available at: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm [Accessed 9 Dec. 2019].

Powell-Morse, A. (2019). *Iterative Model: What Is It And When Should You Use It?*. [online] Airbrake Blog. Available at: https://airbrake.io/blog/sdlc/iterative-model [Accessed 9 Dec. 2019].

Project Report

Exam Scheduling System

Audrius Sauciunas (293156)

Gabriela Melania Poteras (293122)

Pal Jambor (293158)

Supervisors:

Mona Wendel Andersen

Steffen Vissing Andersen

VIA University College

Software Technology Engineering

I. Semester

20/12/2019

Table of contents

Abstract

The purpose of this report is to document the development of the exam scheduling system. The system was developed as a request from the Software Engineering Department of VIA University College Horsens. The focus is on creating and modifying exams based on class, course, date, and room.

The document presents all the stages of system development. The requirements have been decided upon analyzing the case presented by the client followed by the diagrams, design, and implementation.

The program has been developed using Java programming language and JavaFX (Openjfx.io, 2019), following the Waterfall method. The website has been made using HTML, CSS, and JavaScript.

The result of the project is an exam scheduling system, consisting of two parts: a Java Application and a Website.

The report includes analysis, design, implementation, test, results, discussion, conclusion and the future of the project.

1. Introduction

VIA University College, located in the Central region of Denmark, educates future engineers, designers, teachers, education specialists, nurses among others. It offers more than 40 full degree programmes at eight campuses across the region, being the largest institution for further education in Northern Europe (Studyindenmark.dk, 2019). Despite the well-structured working system, with so many students and faculty staff, the university faces challenges scheduling exams without conflict. Students and teachers, the stakeholders, are unable to efficiently access information about when, where and who will be administering an exam. That's why the Software Engineering Department of VIA University College Horsens (the customer) requested students to make an exam scheduling program for the department. The exam planning is currently maintained manually in a static excel spreadsheet, which causes unnecessary stress for both students and teachers due to the difficulty of planning ahead.

In the current system, it is difficult to keep track of the room availability, size, and available equipment resulting in ad-hoc room changes leading to lost exam time and potentially inadequate presentation equipment in the room. Room overlapping is one of the major problems occurring every exam period, therefore causing a lot of stress on examiners and students alike. In addition, the equipment in every room on the campus may vary, depending on the usual usage of said room.

Every other problem considered is as follows – the schedule is preliminary, which means it can be updated at all times, sometimes without the knowledge of the students and examiners. Students may feel uncomfortable in unfamiliar rooms, causing them to be nervous, which reflects poorly on their performance. Moreover, the teacher schedules can interfere causing exam delays.

The program only includes a single user (the secretary) and it is only for Software Engineering students.

2. Analysis

The analysis segment includes the functional and non-functional requirements, use case diagrams, use case descriptions, and domain model that has been created after analyzing the case given by the client and the interview with them held shortly after the presentation.

2.1 Functional requirements:

The purpose of this part is to present the functional requirements of the system. They have been prioritized according to the difficulty and necessity assigned to each one of them.

Critical priority:

1. As a secretary, I want to be able to schedule an exam using the following information: room and room equipment, course, class, and date.
2. As a secretary, I should have the ability to edit and remove exams in order to accommodate necessary changes.
3. As a student, I want to be able to access my exam schedule on the website by my class, so that I know when to show up for my exam.
4. As a teacher, I want to be able to access the schedule by my name on the website so that I know where and when I am supposed to supervise exams.
5. As a secretary, I want to select rooms by the availability of functioning HDMI and VGA connectors so that the students and teachers can connect their computers for presentations.

High priority:

6. As a secretary, I want the system to create a file of the scheduled exams, so it can be uploaded to the website.
7. As a secretary, I want to type using a graphical user interface, so that it is easier to enter and overview the data inside the system.
8. As a secretary, I want to be able to check the schedule and each exam’s data, so that I can see the already scheduled exams.

9. As a secretary, I want to limit oral exams to 20 people per day because that is the maximum number of students that can take an oral exam.
10. As secretary, I would like to filter the schedules by the teacher’s id, courses or classes so that it is easier to find the needed information.
11. As a student, I want to be able to filter the information on the website by course, so that it's easier to access the specific information that I need.
12. As a teacher, I want to be able to filter information on the website by my VIA ID, so that I can see in which exams I have to be present.

Low priority:

13. As a secretary, I would like to schedule the oral exams in the classes’ designated classrooms, so that their testing experience is optimal.
14. As a secretary, I want the schedule of exams by the teacher and by classes to be spread out throughout the exam period so that each teacher or class is not scheduled an exam on three consecutive days.
15. As a secretary, I want the system to be able to identify scheduling conflicts by the teacher, class, room number and by room size, date, and time so that exams are not double booked by room, teacher, class or time.
16. As a student, I want to be able to access the examiners’ contact information, so that in case of emergency I can quickly contact them.
17. As a secretary, I want to be able to delete the exam schedule, so that the website contains only the most recent schedule.

2.2 Non-functional requirements

18. The system must support Windows 10 and the website must support Google Chrome 78, Microsoft Edge 20, Opera 60.

2.3 Use case diagram and use case descriptions

The use case diagram is based on the requirements mentioned above. The exam scheduling program is single-user but the overall system has three actors: secretary, student, and teacher. As can be seen in Figure 1, the secretary has the option to check the schedule in the program and also to plan an exam and make changes if necessary. Only the teacher and student will check the schedule on the website. The diagram represents what functionalities the actors can use.

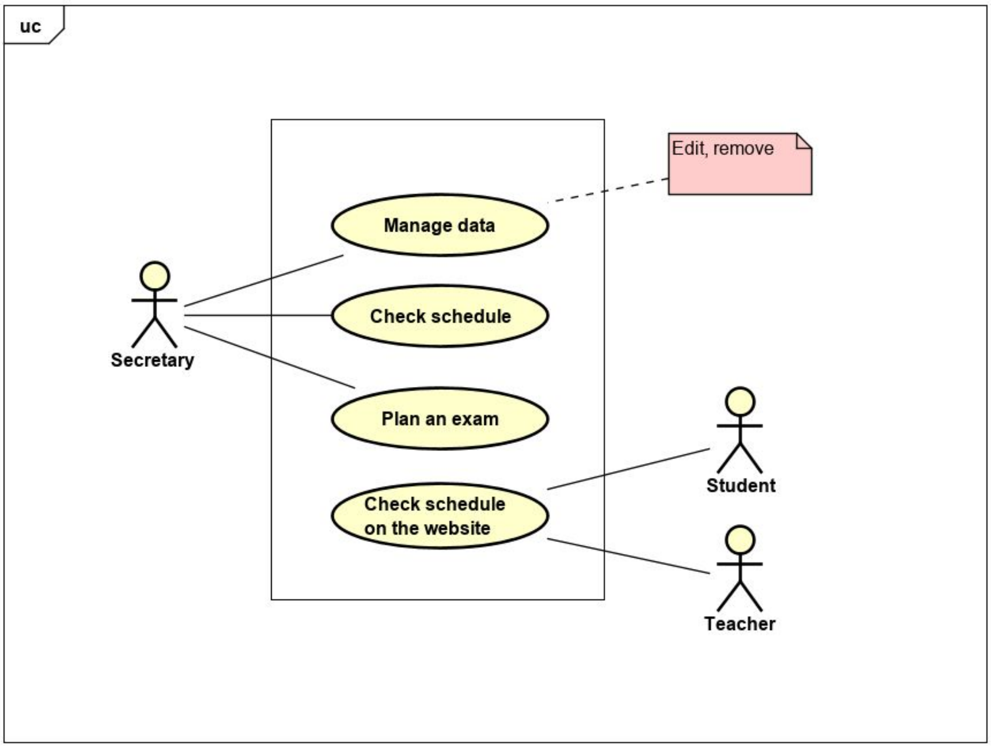
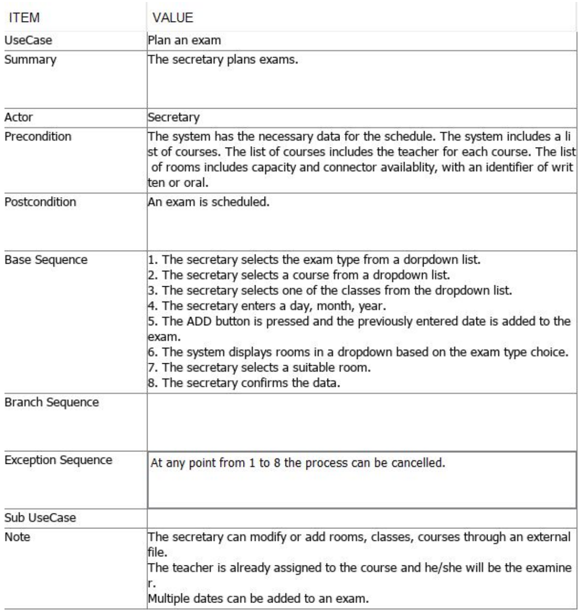


Figure 1

Each use case has a corresponding use case description which depicts the process of each functionality and shows the steps the user must follow when using the system. In Figure 2, the Plan an exam use case description can be seen as this functionality is the most important one. Every use case description has also an activity diagram, which shows the flow of activities in a process including concurrent activities and branches. In Figure 3, the activity diagram of a Plan an exam can be seen. The other use case descriptions can be found in Appendix 1-3 and activity diagrams in Appendix 4-6.



ITEM	VALUE
UseCase	Plan an exam
Summary	The secretary plans exams.
Actor	Secretary
Precondition	The system has the necessary data for the schedule. The system includes a list of courses. The list of courses includes the teacher for each course. The list of rooms includes capacity and connector availablity, with an identifier of written or oral.
Postcondition	An exam is scheduled.
Base Sequence	1. The secretary selects the exam type from a dorpdown list. 2. The secretary selects a course from a dropdown list. 3. The secretary selects one of the classes from the dropdown list. 4. The secretary enters a day, month, year. 5. The ADD button is pressed and the previously entered date is added to the exam. 6. The system displays rooms in a dropdown based on the exam type choice. 7. The secretary selects a suitable room. 8. The secretary confirms the data.
Branch Sequence	
Exception Sequence	At any point from 1 to 8 the process can be cancelled.
Sub UseCase	
Note	The secretary can modify or add rooms, classes, courses through an external file. The teacher is already assigned to the course and he/she will be the examiner. Multiple dates can be added to an exam.

Figure 2

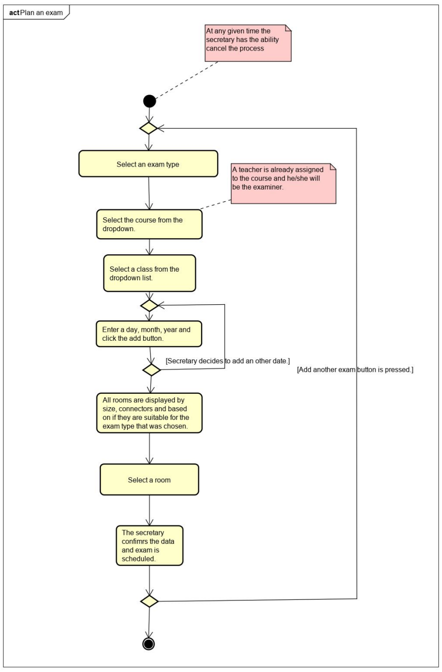

actPlan an exam
At any given time the secretary has the ability cancel the process
Select an exam type
A teacher is already assigned to the course and he/she will be the examiner.
Select the course from the dropdown.
Select a class from the dropdown list.
Enter a day, month, year and click the add button.
[Secretary decides to add an other date.]
[Add another exam button is pressed.]
All rooms are displayed by size, connectors and based on if they are suitable for the exam type that was chosen.
Select a room
The secretary confimrs the data and exam is scheduled.

Figure 3

Note: The Activity Diagram doesn't fully correspond with the java program. For that, the program needs to be updated to disable the next steps before the previous ones are completed.

As use cases and requirements correspond a clear link can be seen between them. In the following table the use cases and its' covered requirements are presented:

Use Case	**Covered requirements**
Plan an exam	1,5,6,7,9,13,14,15
Check schedule	7,8,10
Manage data	2,5,7
Check schedule on the website	3,4,11,12,16,17

2.4 Domain model

Based on the requirements, use case descriptions and activity diagrams, as the last step of analysis a domain model was created to provide an overview of the classes and relations between them. The domain model can be seen in Figure 4:

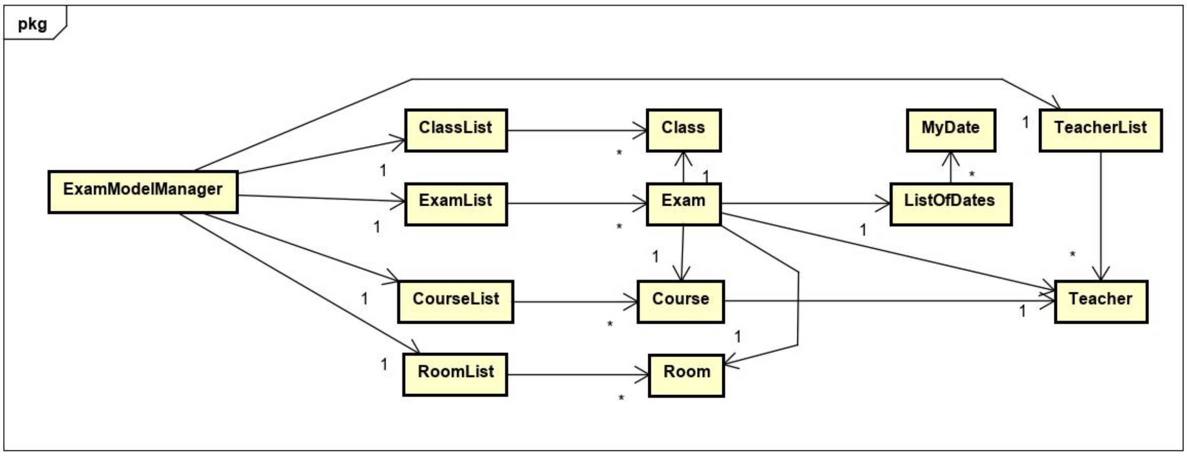


Figure 4

The diagram represents the connection between classes and there are some important facts that must be stated based on the domain:

1. Every List class has an ArrayList of objects.
2. Each course has a pre-assigned teacher.
3. An exam has a list of dates in order to allow oral exams to be scheduled on multiple days.

3. Design

3.1 Class diagram

The class diagram can be divided into three sections: domain (Figure 5), mediator and view. In order to make a class diagram, these three parts had to be developed simultaneously. The full diagram with all the methods and variables can be found in Appendix 9.

Domain

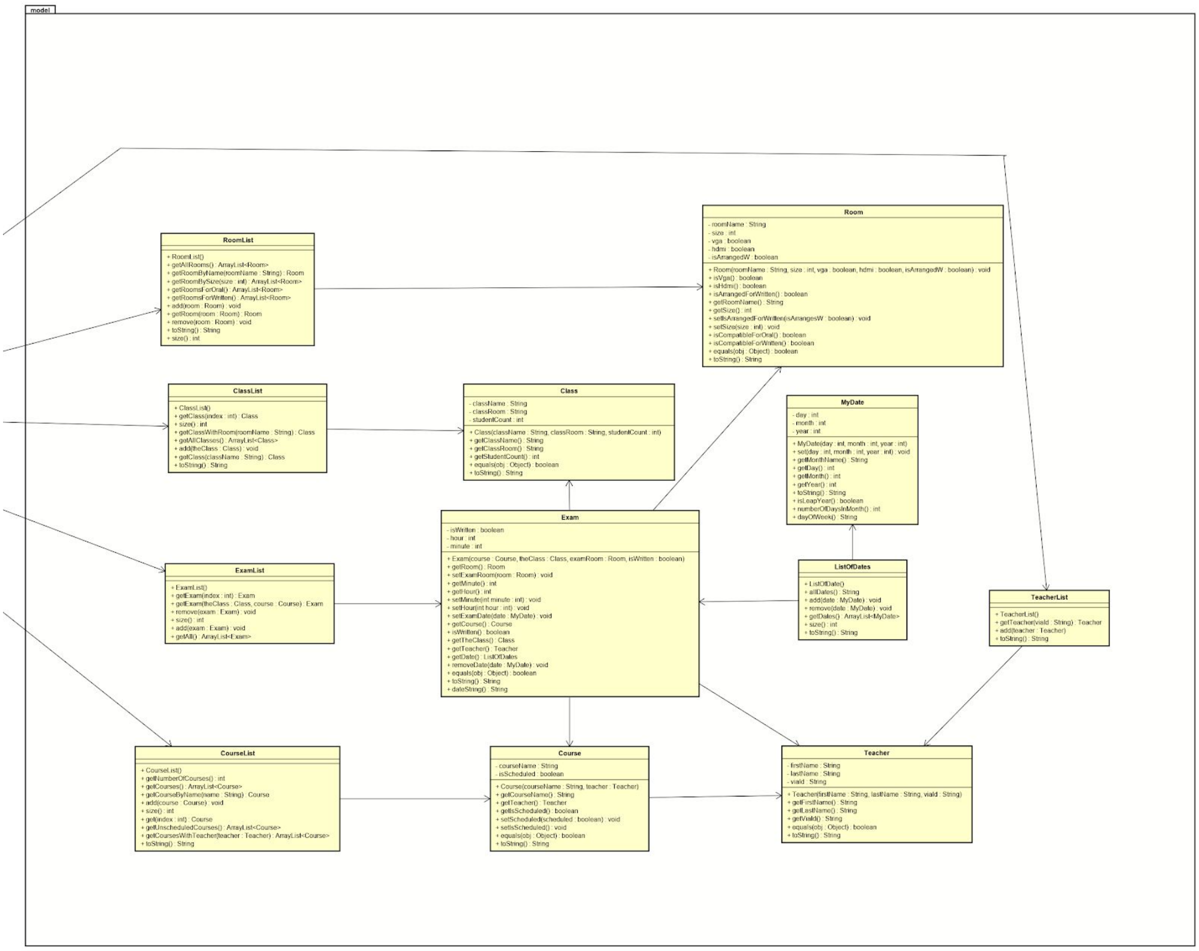

model
RoomList
+ RoomList()
+ getAllRooms() : ArrayList<Room>
+ getRoomByName(roomName : String) : Room
+ getRoomBySize(size : int) : ArrayList<Room>
+ getRoomsForOral() : ArrayList<Room>
+ getRoomsForWritten() : ArrayList<Room>
+ add(room : Room) : void
+ getRoom(room : Room) : Room
+ remove(room : Room) : void
+ toString() : String
+ size() : int
Room
- roomName : String
- size : int
- vga : boolean
- hdmi : boolean
- isArrangedW : boolean
+ Room(roomName : String, size : int, vga : boolean, hdmi : boolean, isArrangedW : boolean) : void
+ isVga() : boolean
+ isHdmi() : boolean
+ isArrangedForWritten() : boolean
+ getRoomName() : String
+ getSize() : int
+ setIsArrangedForWritten(isArrangesW : boolean) : void
+ setSize(size : int) : void
+ isCompatibleForOral() : boolean
+ isCompatibleForWritten() : boolean
+ equals(obj : Object) : boolean
+ toString() : String
ClassList
+ ClassList()
+ getClass(index : int) : Class
+ size() : int
+ getClassWithRoom(roomName : String) : Class
+ getAllClasses() : ArrayList<Class>
+ add(theClass : Class) : void
+ getClass(className : String) : Class
+ toString() : String
Class
- className : String
- classRoom : String
- studentCount : int
+ Class(className : String, classRoom : String, studentCount : int)
+ getClassName() : String
+ getClassRoom() : String
+ getStudentCount() : int
+ equals(obj : Object) : boolean
+ toString() : String
MyDate
- day : int
- month : int
- year : int
+ MyDate(day : int, month : int, year : int)
+ set(day : int, month : int, year : int) : void
+ getMonthName() : String
+ getDay() : int
+ getMonth() : int
+ getYear() : int
+ toString() : String
+ isLeapYear() : boolean
+ numberOfDaysInMonth() : int
+ dayOfWeek() : String
Exam
- isWritten : boolean
- hour : int
- minute : int
+ Exam(course : Course, theClass : Class, examRoom : Room, isWritten : boolean)
+ getRoom() : Room
+ setExamRoom(room : Room) : void
+ getMinute() : int
+ getHour() : int
+ setMinute(int minute : int) : void
+ setHour(int hour : int) : void
+ setExamDate(date : MyDate) : void
+ getCourse() : Course
+ isWritten() : boolean
+ getTheClass() : Class
+ getTeacher() : Teacher
+ getDate() : ListOfDates
+ removeDate(date : MyDate) : void
+ equals(obj : Object) : boolean
+ toString() : String
+ dateString() : String
ExamList
+ ExamList()
+ getExam(index : int) : Exam
+ getExam(theClass : Class, course : Course) : Exam
+ remove(exam : Exam) : void
+ size() : int
+ add(exam : Exam) : void
+ getAll() : ArrayList<Exam>
ListOfDates
+ ListOfDate()
+ allDates() : String
+ add(date : MyDate) : void
+ remove(date : MyDate) : void
+ getDates() : ArrayList<MyDate>
+ size() : int
+ toString() : String
TeacherList
+ TeacherList()
+ getTeacher(viaId : String) : Teacher
+ add(teacher : Teacher)
+ toString() : String
CourseList
+ CourseList()
+ getNumberOfCourses() : int
+ getCourses() : ArrayList<Course>
+ getCourseByName(name : String) : Course
+ add(course : Course) : void
+ size() : int
+ get(index : int) : Course
+ getUnscheduledCourses() : ArrayList<Course>
+ getCoursesWithTeacher(teacher : Teacher) : ArrayList<Course>
+ toString() : String
Course
- courseName : String
- isScheduled : boolean
+ Course(courseName : String, teacher : Teacher)
+ getCourseName() : String
+ getTeacher() : Teacher
+ getIsScheduled() : boolean
+ setScheduled(scheduled : boolean) : void
+ setIsScheduled() : void
+ equals(obj : Object) : boolean
+ toString() : String
Teacher
- firstName : String
- lastName : String
- viaId : String
+ Teacher(firstName : String, lastName : String, viaId : String)
+ getFirstName() : String
+ getLastName() : String
+ getViaId() : String
+ equals(obj : Object) : boolean
+ toString() : String

Figure 5

The domain consists of eleven classes, which ensure the base of the program. The most important classes are Room, Course, Teacher, Class, MyDate, and Exam.

Course class has a Teacher element, meaning when a course's exam is scheduled, the previously assigned teacher to the course is already set as the examiner. Class and Teacher classes hold general information about the class, respectively the teacher.

The ListOfDates class is responsible for holding MyDate objects so that it has the ability to schedule multiple days for exams if needed, relevant for the oral exam. Room, Class, Teacher, and Course have a corresponding List-classes, which gives the ExamModelManager the ability to access a list of all of these four objects. When creating an Exam object, a Course, Class, Room, and ListOfDates objects are needed. The teacher that will be assigned for the exam is the one who is assigned to the course and is automatically set with the selection of the course.

Mediator

The mediator package, which can be seen in Appendix 10, holds a class for file reading and writing, as well as a ModelManager class, which is responsible for creating a connection between the GUI, the FileManager and the Domain. File reading and writing is done mostly by using an external jar(parser), as it provides a simple solution for both. All of the files created and read are XML files for better manageability. Since the exam schedule has to be imported on a website and is not only used locally, another method had to be created which produces a simplified XML file in a predefined structure.

View

In the view package (shown in Appendix 11), the most important thing is simplicity, as the user should be able to utilize the program with ease. The view has four XML files: CheckScheduleview, DetailsView, ManageDataView and PlanAnExamView, which hold how the program should look when running. Each XML file has a corresponding Controller class which gives them functionality. The DetailViewController connects the windows, it is the simplest one. This is where the user is able to select what he/she would like to do in the program. There are two additional classes to be able to display the stored data in tables throughout the GUI: ExamViewModel and ExamLisViewModel. The ViewHandler class is responsible for opening, loading and closing views, ensuring the connection between the separate views.

3.2 Sequence diagram

Sequence diagrams are created in order to show how objects are working together. For this reason, a sequence diagram was created about a method that is used to get rooms that are suitable for written examinations (Figure 5).

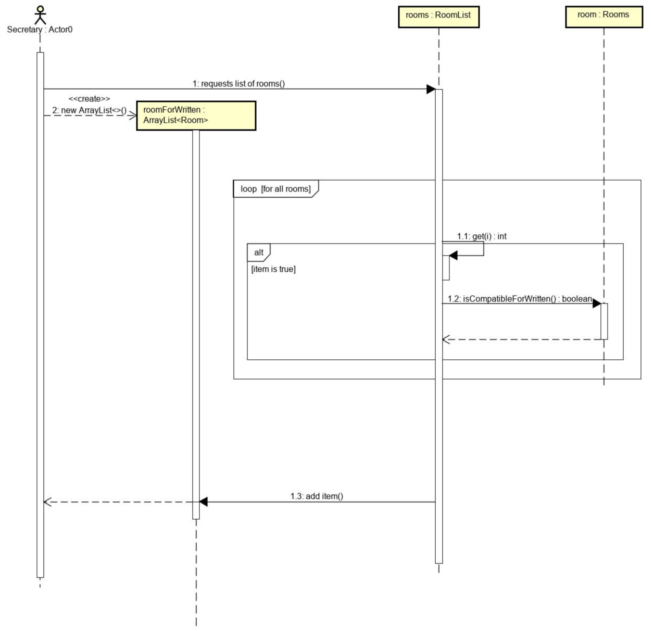
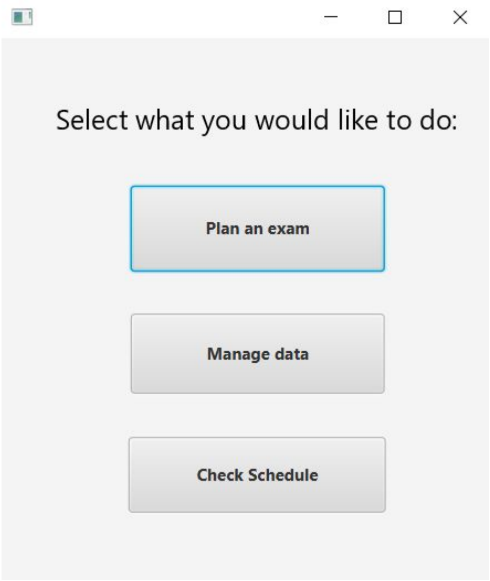


Figure 6

3.3 User interface



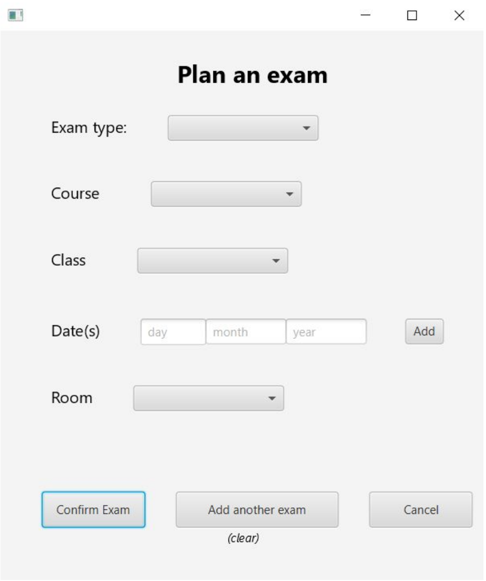


Figure 7

Figure 8

The goal of the user interface is to create an output for the user that is simple and easy to use and enables to follow the predetermined activity diagrams. It was made using the Scene Builder to allow visual editing of the interface.

As can be seen above in Figure 7, the main view was needed in order to give structure to the program and to be able to switch between windows.

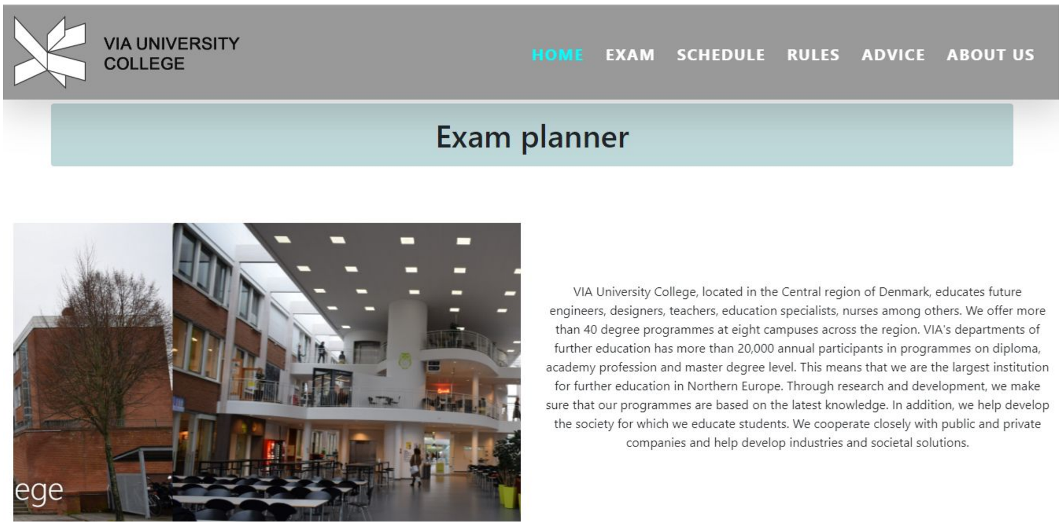
In Figure 8 is the Plan an exam view, which is the main functionality of the program. From the top to the bottom, following the corresponding activity diagram, the process of planning an exam is easy to follow and should be followed sequentially.

The other views can be seen in appendix 7-8.

3.4 Website

The final design and outlook are made to be simple and straightforward so that it is easy to navigate and use, while its purpose is still fulfilled. The colour scheme was chosen to match the colour palette of the university's official website and also be easy on the eyes.

It features a Home page (Figure 9) where a general overview of the school and the purpose of the site is presented, an Exam page that details the types of exams students can have, the main focus of the website, the Schedule page which consists of a table where the exams are displayed, a Rules page where the school rules related to the exams are presented, Advice page which contains advice and tips for students that could help them before the exam and an About Us page where the developers' contact information can be found.



Exam planner

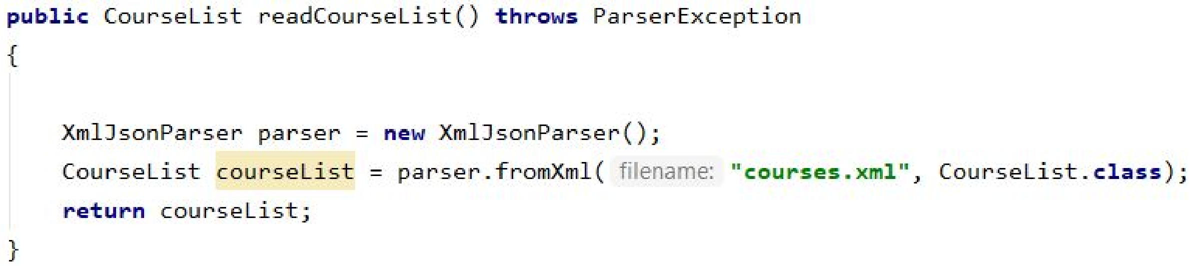
VIA University College, located in the Central region of Denmark, educates future engineers, designers, teachers, education specialists, nurses among others. We offer more than 40 degree programmes at eight campuses across the region. VIA's departments of further education has more than 20,000 annual participants in programmes on diploma, academy profession and master degree level. This means that we are the largest institution for further education in Northern Europe. Through research and development, we make sure that our programmes are based on the latest knowledge. In addition, we help develop the society for which we educate students. We cooperate closely with public and private companies and help develop industries and societal solutions.

Figure 9

4. Implementation

4.1 JAVA Program

The program was implemented based on the analysis using object-oriented programming. The idea of making a separate class for reading and writing files makes file management convenient. Methods are created in order to read the necessary data that is required for scheduling exams. For this purpose an external jar (parser) was used (ICT-engineering.dk, 2019), making the methods short and not overcomplicating.

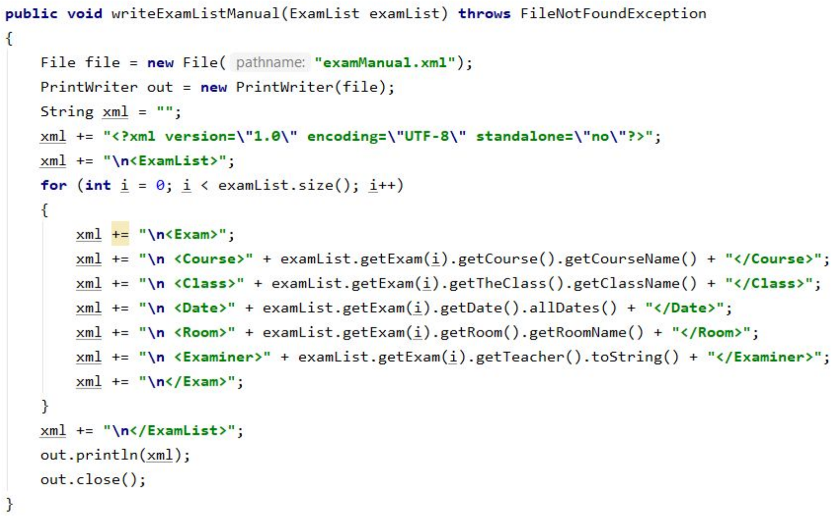


```
public CourseList readCourseList() throws ParserException
{

    XmlJsonParser parser = new XmlJsonParser();
    CourseList courseList = parser.fromXml( filename: "courses.xml", CourseList.class);
    return courseList;
}
```

Figure 10

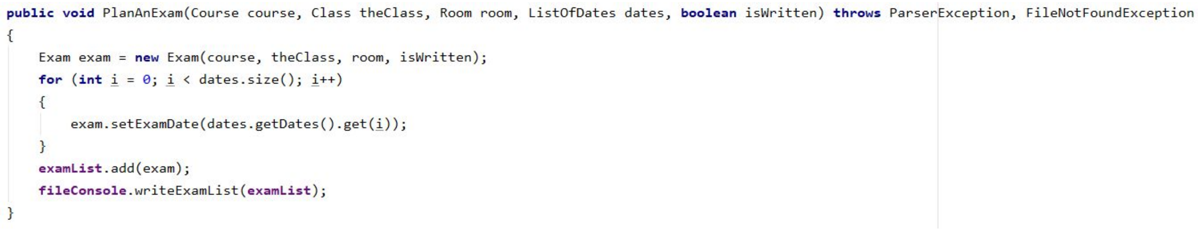
Although the XML parser is simple to use, for the XML that is imported on the website a method had to be created (writeExamListManual(ExamList examList)), so the result of writing to the XML is predefined and only the most important information is stored. By making it manually, it allows making a prefered structure in contrast with the parser, where there is less control over the outcome. The method takes an ExamList object, which is written to an XML. First, a File object is created and then PrintWriter is used to enable writing to a file. In a for loop the necessary data is acquired from the examList object then the file is written to and closed.



```
public void writeExamListManual(ExamList examList) throws FileNotFoundException
{
    File file = new File( pathname: "examManual.xml");
    PrintWriter out = new PrintWriter(file);
    String xml = "";
    xml += "<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"no\"?>";
    xml += "\n<ExamList>";
    for (int i = 0; i < examList.size(); i++)
    {
        xml += "\n<Exam>";
        xml += "\n <Course>" + examList.getExam(i).getCourse().getCourseName() + "</Course>";
        xml += "\n <Class>" + examList.getExam(i).getTheClass().getClassName() + "</Class>";
        xml += "\n <Date>" + examList.getExam(i).getDate().allDates() + "</Date>";
        xml += "\n <Room>" + examList.getExam(i).getRoom().getRoomName() + "</Room>";
        xml += "\n <Examiner>" + examList.getExam(i).getTeacher().toString() + "</Examiner>";
        xml += "\n</Exam>";
    }
    xml += "\n</ExamList>";
    out.println(xml);
    out.close();
}
```

Figure 11

In the ExamModelManager a PlanAnExam method was created to enable the planning of an exam (Figure 12). The method takes five variables, that are needed to create an Exam object. An Exam is created using Course, Class, Room objects and a boolean which determines whether the exam is written. After, exam dates are set for the exam in a for-loop.

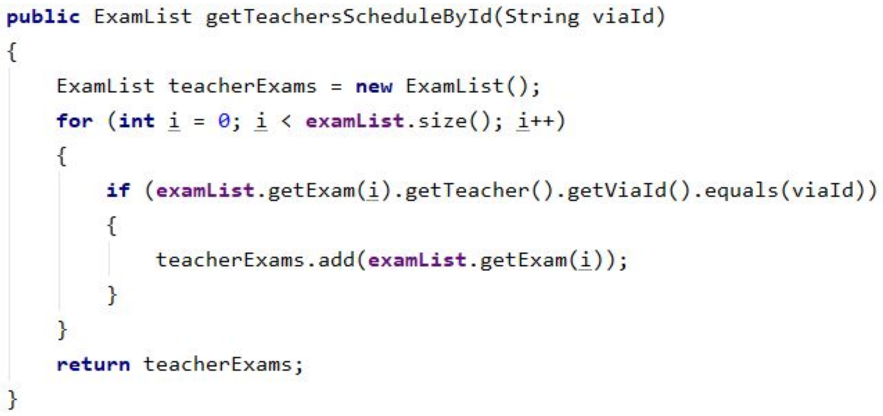


```
public void PlanAnExam(Course course, Class theClass, Room room, ListOfDates dates, boolean isWritten) throws ParserException, FileNotFoundException
{
    Exam exam = new Exam(course, theClass, room, isWritten);
    for (int i = 0; i < dates.size(); i++)
    {
        exam.setExamDate(dates.getDates().get(i));
    }
    examList.add(exam);
    fileConsole.writeExamList(examList);
}
```

Figure 12

Finally, the exam created gets added to the examList and the XML file that holds the exams is overwritten with the updated ExamList.

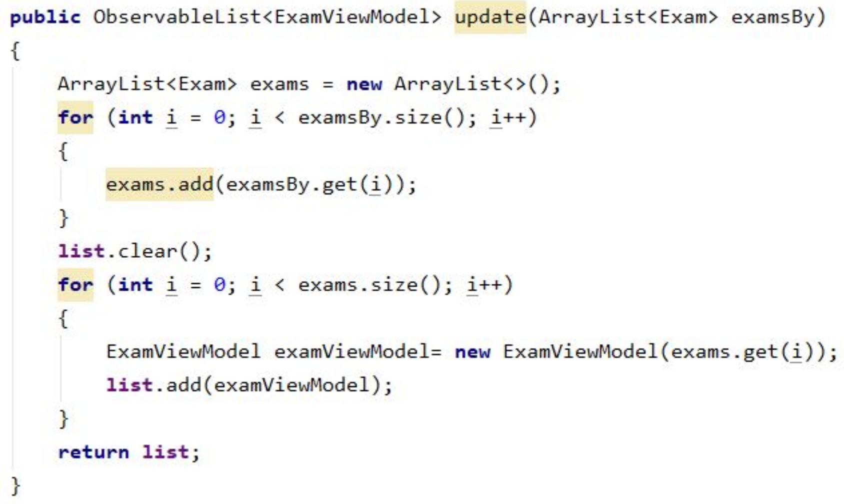
In order to have the ability to filter the list of exams based on different parameters in the CheckScheduleView, three methods were created, one of which is Figure 13. The getTeachersScheduleById method checks all of the exams stored in the examList and in case of a match to the parameter, the exam is added to the ArrayList created at the beginning of the method. The other two methods created for this purpose work in a similar way.



```
public ExamList getTeachersScheduleById(String viaId)
{
    ExamList teacherExams = new ExamList();
    for (int i = 0; i < examList.size(); i++)
    {
        if (examList.getExam(i).getTeacher().getViaId().equals(viaId))
        {
            teacherExams.add(examList.getExam(i));
        }
    }
    return teacherExams;
}
```

Figure 13

The Update method seen in Figure 14, has the purpose to update the String Properties in the ExamViewModel to make the new, filtered schedule be seen in the tables in the CheckScheduleView. It updates by creating a new ArrayList with all the new data and clears the existing list with all the properties and then adds the new ones.



```
public ObservableList<ExamViewModel> update(ArrayList<Exam> examsBy)
{
    ArrayList<Exam> exams = new ArrayList<>();
    for (int i = 0; i < examsBy.size(); i++)
    {
        exams.add(examsBy.get(i));
    }
    list.clear();
    for (int i = 0; i < exams.size(); i++)
    {
        ExamViewModel examViewModel= new ExamViewModel(exams.get(i));
        list.add(examViewModel);
    }
    return list;
}
```

Figure 14

4.2 Website

The implementation of the website's design has been done before the implementation of the program.

HTML, CSS, and JavaScript have been used to implement the website.

Balsamiq Mockup has been used to create an outline of the website before coding (Figure 15).

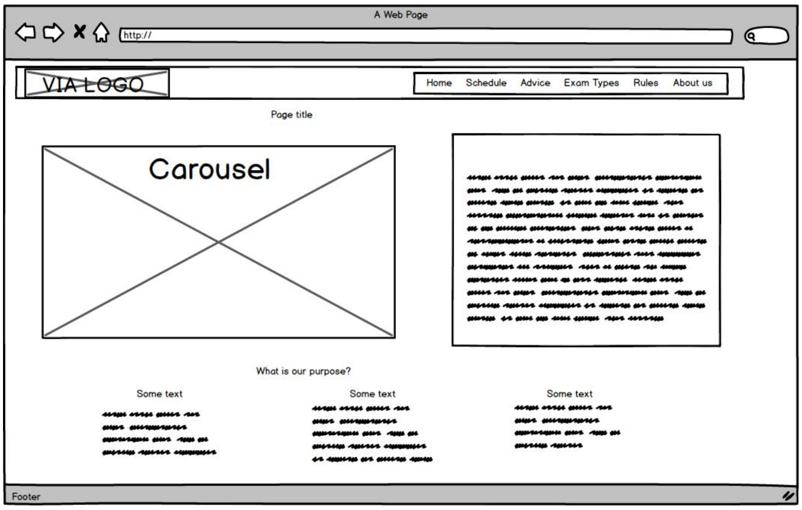
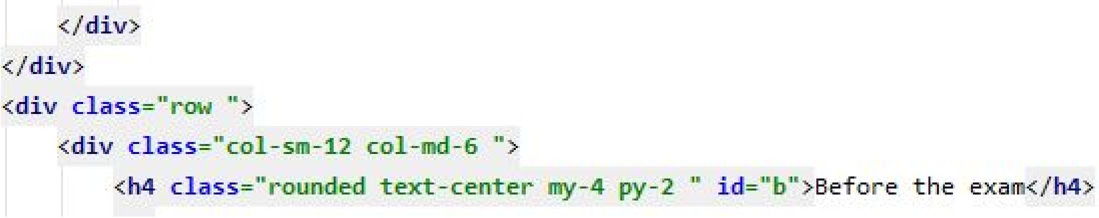


Figure 15

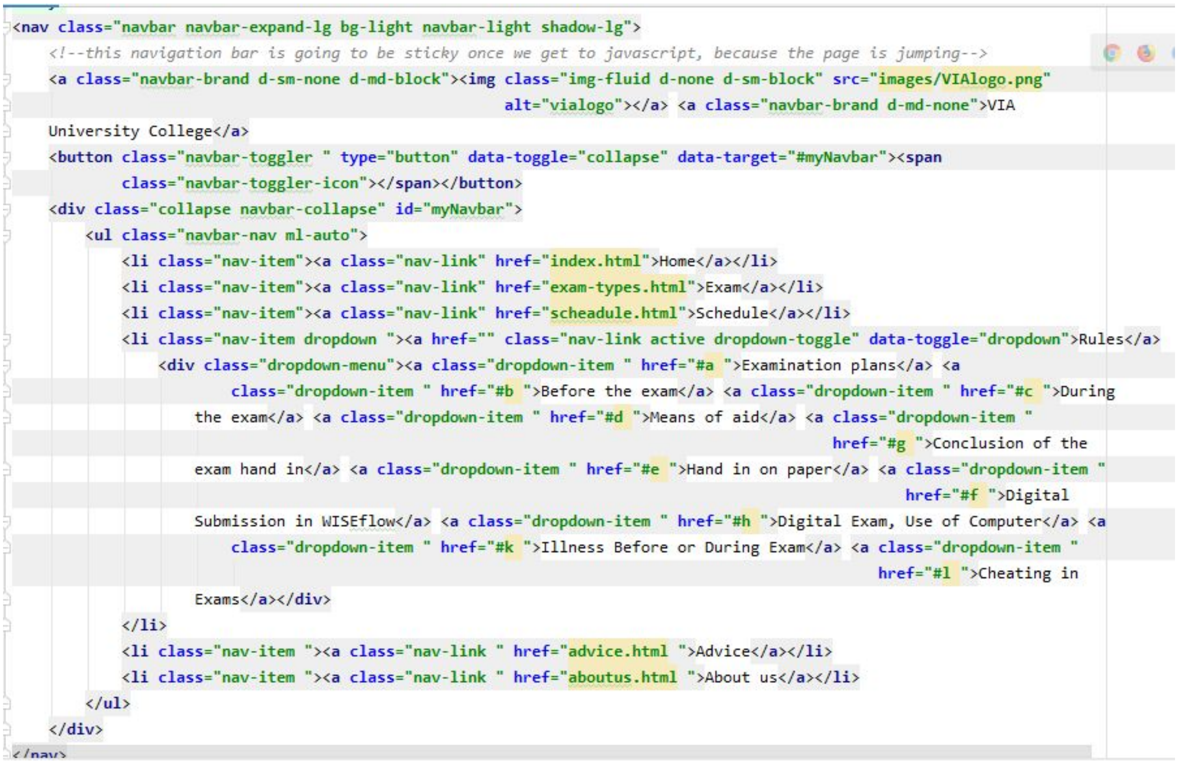
Bootstrap was added to make it responsive and its keyword class names have been used to reduce and improve CSS styling code.



```
        </div>
</div>
<div class="row ">
        <div class="col-sm-12 col-md-6 ">
                <h4 class="rounded text-center my-4 py-2 " id="b">Before the exam</h4>
```

Figure 16

In order to make it according to the planned design, the navigation bar needed specific targeting in the CSS style file as there were no bootstrap classes that could be used to achieve the desired result (Figure 17).



```
<nav class="navbar navbar-expand-lg bg-light navbar-light shadow-lg">
    <!--this navigation bar is going to be sticky once we get to javascript, because the page is jumping-->
    <a class="navbar-brand d-sm-none d-md-block"><img class="img-fluid d-none d-sm-block" src="images/VIAlogo.png"
                                                      alt="vialogo"></a> <a class="navbar-brand d-md-none">VIA
    University College</a>
    <button class="navbar-toggler " type="button" data-toggle="collapse" data-target="#myNavbar"><span
            class="navbar-toggler-icon"></span></button>
    <div class="collapse navbar-collapse" id="myNavbar">
        <ul class="navbar-nav ml-auto">
            <li class="nav-item"><a class="nav-link" href="index.html">Home</a></li>
            <li class="nav-item"><a class="nav-link" href="exam-types.html">Exam</a></li>
            <li class="nav-item"><a class="nav-link" href="scheadule.html">Schedule</a></li>
            <li class="nav-item dropdown "><a href="" class="nav-link active dropdown-toggle" data-toggle="dropdown">Rules</a>
                <div class="dropdown-menu"><a class="dropdown-item " href="#a ">Examination plans</a> <a
                        class="dropdown-item " href="#b ">Before the exam</a> <a class="dropdown-item " href="#c ">During
                    the exam</a> <a class="dropdown-item " href="#d ">Means of aid</a> <a class="dropdown-item "
                                                                                          href="#g ">Conclusion of the
                    exam hand in</a> <a class="dropdown-item " href="#e ">Hand in on paper</a> <a class="dropdown-item "
                                                                                                  href="#f ">Digital
                    Submission in WISEflow</a> <a class="dropdown-item " href="#h ">Digital Exam, Use of Computer</a> <a
                        class="dropdown-item " href="#k ">Illness Before or During Exam</a> <a class="dropdown-item "
                                                                                               href="#l ">Cheating in
                    Exams</a></div>
            </li>
            <li class="nav-item "><a class="nav-link " href="advice.html ">Advice</a></li>
            <li class="nav-item "><a class="nav-link " href="aboutus.html ">About us</a></li>
        </ul>
    </div>
</nav>
```

Figure 17

5. Test

This section presents result from testing, performed at the end of the implementation of each feature, to check correct functionality. In order to declare a use case implemented, the tests must have been satisfactorily run. The purpose of testing functionalities is to find bugs that are present and ensure that they remain fixed. To declare a test successful, the result from the test should be the same as the expected results.

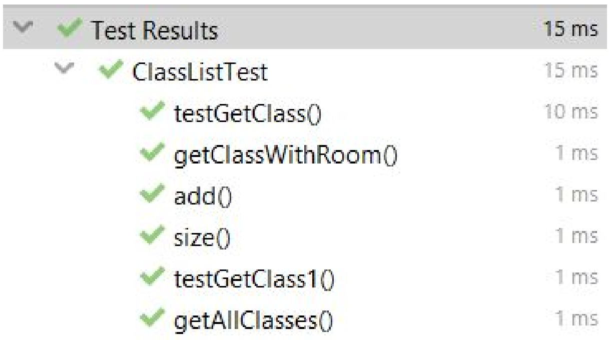
Testing was performed by using test cases and JUnit. Not all parts of the program were tested because of time constraint. The table below shows the test of the most important use case: Plan an exam.

Pre-condition	The system has the necessary data for the schedule. The system includes a list of courses. The list of courses includes the teacher for each course. The list of rooms includes capacity and connector availability, with an identifier of written or oral.				
Plan an exam					
Step	Test steps	Expected Result	Actual Result	Status	Note
1	Select exam type	Exam type selected	As expected	Pass	
2	Select a course	Course selected	As expected	Pass	
3	Select a class	Class selected	As expected	Pass	
5	Add a date or multiple	Date(s) is added	As expected	Pass	An exam can be planned without dates.
6	Rooms in dropdown updated	Rooms suitable displayed	As expected	Pass	
7	Select a room	Room selected	As expected	Pass	No option to select a different room other than the filtered ones
8	Confirm exam	Exam created	As expected	Pass	

More possible test scenarios are:

- the secretary cancels the planning of the exam - no exam planned - as expected
- the secretary tries to plan an exam without entering the data - error message - as expected
- the secretary wants to plan more than one exam - possible with clearing the selections

Due to time restriction and the number of classes considered, it was not possible to test each method individually, but a few tests have been run. The table in Figure 18 represents the test results of ClassList:



```
✓ Test Results                     15 ms
  ✓ ClassListTest                  15 ms
    ✓ testGetClass()               10 ms
    ✓ getClassWithRoom()            1 ms
    ✓ add()                         1 ms
    ✓ size()                        1 ms
    ✓ testGetClass1()               1 ms
    ✓ getAllClasses()               1 ms
```

Figure 18

The JUnit test can be found in Appendices. More tests when the system is completely implemented are required.

6. Results

The result of the system is a program that can schedule and modify exams and a website where the schedule is displayed.

In the table below, an overview of the requirements and their progress is presented.

Requirements	Status	Solution if not implemented
1. As a secretary, I want to be able to schedule an exam using the following information: room and room equipment, course, class, and date.	Done	
2. As a secretary, I should have the ability to edit and remove exams in order to accommodate necessary changes.	Done	
3. As a student, I want to be able to access my exam schedule on the website by my class, so that I know when to show up for my exam.	Done	
4. As a teacher, I want to be able to access the schedule by my name on the website so that I know where and when I am supposed to supervise exams.	Done	-
5. As a secretary, I want to select rooms by the availability of functioning HDMI and VGA connectors so that the students and teachers can connect their computers for presentations.	Done	-
6. As a secretary, I want the system to create a file of the scheduled exams, so it can be uploaded to the website.	Done	-
7. As a secretary, I want to type using a graphical user interface, so that it is easier to enter and overview the data inside the system	Done	-
8. As a secretary, I want to be able to check the schedule and each exam's data, so that I can see the already scheduled exams.	Done	-

9. As a secretary, I want to limit oral exams to 20 people per day because that is the maximum number of students that can take an oral exam.	Done	-
10. As secretary, I would like to filter the schedules by the teacher's id, courses or classes so that it is easier to find the needed information.	Done	-
11. As a student, I want to be able to filter the information on the website by course, so that it's easier to access the specific information that I need.	Done	-
12. As a teacher, I want to be able to filter information on the website by my VIA ID, so that I can see in which exams I have to be present.	Done	-
13. As a secretary, I would like to schedule the oral exams in the classes' designated classrooms, so that their testing experience is optimal.	Not Done	the classes have a preassigned classroom which should be automatically assigned to the Exam if available
14. As a secretary, I want the schedule of exams by the teacher and by classes to be spread out throughout the exam period so that each teacher or class is not scheduled an exam on three consecutive days.	Not Done	needs conflict checking for dates with a method which checks if there are exams for a selected class/teacher on previous or following days
15. As a secretary, I want the system to be able to identify scheduling conflicts by the teacher, room number and by room size, date, and time so that exams are not double booked by room, teacher or time.	Not Done	with methods that contain several if-statements to check each conflict with the already booked exams
16. As a student, I want to be able to access the examiners' contact information, so that in case of emergency I can quickly contact them.	Not Done	based on the VIA ID, emails can be created and added to the website
17. As a secretary, I want to be able to delete the exam schedule, so that the website contains only the most recent schedule.	Not Done	it can only be done manually

18. The system must support Windows 10 and the website must support Google Chrome 78, Microsoft Edge 20, Opera 60	Done	-

7. Discussion

The filtering option on both the website and the program works well even though the filtration in the program could be improved by adding options for the date. The website's outlook is easy to follow, well structured and it provides information about the exams in general.

The Manage data window fulfils its purpose, although data editing is not precise as the removal of dates is not an option. This feature could be added as an improvement in a future update. In order to have all the dates modified, the exam has to be deleted and remade with the correct date.

Although the reading of the exams.xml file was implemented, when calling the method the file was not read. Even though other options were tried, like manual and binary reading, they proved to be unsuccessful.

As far as courses are considered, in the program, only one teacher can be assigned to a Course object. As an update, an ArrayList of Teacher object could be assigned to a course, instead of a single one, as some of the courses have more than one teacher.

The requirements had to be edited and rearranged in order to provide a functional system and the lower requirements have not been implemented due to time restrictions.

Concerning the Exam class, there were two approaches. The first approach uses an abstract Exam class with two subclasses, Oral and Written (Figure 19). The Oral subclass contains an ArrayList to store multiple MyDate objects while the Written subclass has only one MyDate object. This method was the preferred one as it was more performant but due to lack of time, it was decided to implement a more general solution. The second approach (Figure 20) includes an Exam class with a boolean that would check the type: if true the exam is written, if false - oral. The exam list has an ArrayList of dates and in the written exam type case, only one date is in the ArrayList. This approach was chosen as it was considered to be more of a universal approach. If the project is carried on, the first approach would be implemented.

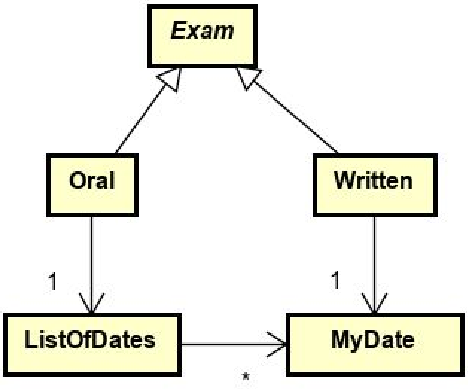


Figure 19

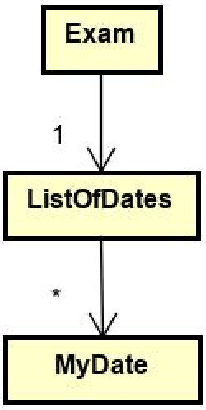


Figure 20

8. Conclusions

With the analyzation of the requirements, the user’s needs were translated into design and partially implemented and tested.

The analysis resulted in use cases with descriptions, activity diagrams and a domain model, describing the relations between the classes.

The systems GUI was made by using a java framework called JavaFX, which proved to be useful as the user interface could be done by visual editing in Scene Builder instead of hard-coding. This way editing the view was noticeably easier and allowed making a more user-friendly interface.

Although not all requirements were implemented, only the critical and high ones, the planning system still has basic functionality and the overall outlook is easy and convenient to use.

The website is a multi-purpose website which has information about exams and the exam table itself. Bootstrap, a CSS framework, proved to be very useful as it helped to make the website more responsive with various devices.

The final product was partly tested with JUnit and test case scenarios. The software is functional and meets all critical and high requirements.

9. Project future

As possibilities are concerned, there are still a lot of improvements that could be done in order to make the exam scheduling easier and better.

Firstly, now the program uses manual date entry, but using a Date Picker could be a good improvement, as it is easier and more convenient to use.

Also, having the option of adding external examiners could be a feature, as it might be important for either students and teachers to know their information.

Currently, the program has default styling, but a custom theme could be added in order to make the program more fit with the university's design.

The website currently uses a table to show the exams: in the future the exams could be displayed in a calendar.

10. Sources of information

Studyindenmark.dk. (2019). *Campus Horsens*. [online] Available at: https://studyindenmark.dk/portal/via-university-college/campus-horsens [Accessed 20 Sep. 2019].

Openjfx.io. (2019). *JavaFX*. [online] Available at: https://openjfx.io/ [Accessed 19 Dec. 2019].

Ict-engineering.dk. (2019). XmlJsonParser [online] Available at: https://ict-engineering.dk/javadoc/XmlJsonParser/ [Accessed 19 Dec. 2019].

11. Appendices

1 Check schedule use case description

2 Manage data use case description

3 Check schedule on the website use case description

4 Check schedule activity diagram

5 Check schedule on the website activity diagram

6 Manage data activity diagram

7 Manage Data View

8 Check Schedule View

9 Complete Class diagram

10 Mediator package

11 View package