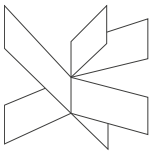


Client/Server System

Process Report – Language Learning Platform

Second Semester – Software Engineering

Sprogcenter midt
En verden til forskel

Supervisors:
Troels Mortensen
Ib Havn

Students:
Gais El-AAsi – 279910
Lyons Leviticus – 275171
Deivydas Zibkus – 280133
Lukas Vaisnoras – 280107
Marcel Notenboom - 279963

Number of characters incl. spaces 34,347 characters

Table of content

List of Figures

1. Introduction

We were introduced to the project on 7th of February when we were offered information in regards to how the project should be developed over the course of the semester as well as minimum requirements and milestones. On 13th of February after researching a few different proposals, we have decided to develop a learning platform for the Sprogcenter Midt.

During the following weeks the project started taking shape, with the progress in defining the functionalities and capabilities for the future project combined with parallel knowledge gained during the Software Engineering (SWE) course the process started quickly to accelerate up to Analysis when more concrete concepts where figured out and summarized in the Domain Model.

SCRUM (Schwaber & Southerland, 2017) was our main framework to structure the work and also it has kept us from losing focus and provided boundaries in how to operate in an efficient way, it made sure that we are in a constant state of hustle (in a good way). The knowledge gained during the SWE course helped us improve each sprint, and as we look back an incremental improvement can be observed.

The Semester Project work during the period when we have had courses was scheduled on Thursdays. In addition the last two – two and a half weeks were reserved solely for the Semester Project which helped us apply the theories that were presented and developed during the SWE classes.

Due to the nature of the project being a Multi – User System (VIA University College, 2019) the Software Development with Java (SDJ) course gave us the necessary knowledge for developing a client – system and the Semester Project was perfect for applying and experimenting with the gained knowledge.

During the whole project we have had several meetings with the supervisor (both scheduled and break/class non-scheduled) that helped us stay focused on what must be done as well as have an outside perspective on the development.

2. Group Description

Having experience working with each other and knowing what to expect helped us a lot when working as a group. In addition having a new (experienced) member in the team made us adopt some work-principles that were beneficial and made the overall team-work more productive and cohesive.

2.1 Group Contract

The Group Contract was developed based on the last semester's contract with necessary addition and modifications. The contract was not used much (for solving major issues) during the semester as the majority of the obstacles encountered were solved with the use of common sense. Even so, it provided the necessary work-ethics agreements and rules that helped us know what to expect from each other. The Group Contract can be seen in the Appendix A.

2.2 Project Experience

All members had some project experience prior to this project which granted a common ground for establishing the general rules on how to work. Moreover it offered the necessary understanding for setting up the expectations from the outcome and the effort that will be put into the project.

2.3 Cultural Background

Even if when analyzing Hofstede's Dimensions for the cultural differences between the members of the group, its relevance can be questioned for the reason that most members having been living in a Danish culture for at least one year (some much more). This caused the cultural differences to be attenuated and diluted combined with the group contract the differences did not impact in a way that can be observed or analyzed (Hofstede, 2019). The Hofstede's chart is presented in the Appendix B.

2.4 Group Roles

Even if it was not formally established each member had a certain role in the group. It is significant to mention that during SCRUM these roles were overwritten and/or separated from the SCRUM roles which will be discussed further in the report.

The group roles helped us work more efficient especially that these roles were naturally developed during the work rather than assigned and/or enforced as a formal responsibility.

- Gais' role in the group was of an overall Project Coordinator, he took over the communication and management responsibilities including ensuring a proper communication between team-members, proper communication with the outside of the group (supervisors), setting up meetings, etc.;
- Levi had the responsibility of ensuring that the team stay's focused on the main direction of the project. He was constantly assessing if the relevance of the project is on track and that the group does not go sideways;
- Marcel has been responsible with the overall assessment of the work that was done. Moreover he made sure that everything is documented and tested before being considered as finished;
- Deivydas was in charge with the development and overall direction for the implementation. His prior experience in Software Development helped us understand new techniques;
- Lukas had the role of ensuring that everyone is motivated and focused towards the project, he had a support role for the group members ensuring a more cohesive work;

In general we had a good collaboration in the developing the project and group assignments with small inevitable obstacles that did not strongly affected the overall work of the group.

4. Project Initiation

When selecting the project's topic we have made a few proposals that were than filtered so they are relevant for the minimum requirements for the project. Next based on an

internal group vote 3 main proposal were selected. The proposal can be seen in Appendix C. The favorite was the Language Learning Platform for a few reasons:

- We had an actual client which offered a good background for the project as well as helped us be more motivated due to the fact that it was not only an abstract idea but an actual needed product;
- It was highly relevant with the requirements of the project, it included Graphical User Interface, Database, Server, Client-Server communication, etc.;
- At the same time it was not too broad and unachievable, meaning that the scope of the project had to be very little delimited;

After some consultations with the supervisors and internal discussions the Language Learning Platform was selected as our main topic for the project.

3. Project Description

The Project Description was our first milestone to achieve before starting the actual project. It had offered us some knowledge in regards to what will need some adjustments in regards to how everyone works before jumping into the project work. The Project Description can be assessed in the Appendix D.

At the same time the project helped us understand better what is to be done as well as guided us through the whole project period to ensure the relevance of the project.

Some additions and modification needed to be done during the project development for some unseen issues that were encountered.

First of all some of the tools that we predicted that will be used were unavailable therefore we needed to adjust (Axosoft – Combination of MiesterTask and Microsoft Excel). Moreover, some tools proved to be inefficient and redundant, therefore it was decided to not use them at all (Microsoft Project).

From another perspective, some milestones have been wrongly estimated, some being unachievable (Minimum Viable Product incl. Basic GUI) within the estimation.

All in all, the Project Description has provided us the necessary guidelines when developing the project and served as a handbook for how everything should work and what is the expected outcome. The problem formulation helped us to stay on track and focus on what is relevant for the project.

5. Project Execution

When describing the project execution it is beneficial to mention that it was not a linear execution and/or development, rather it has fluctuated with ups and downs that has shaped the learning curve of the group as an overall.

5.1 Project Development and Methods

The project was developed during the second semester with an additional of two – two and a half weeks in the end of the semester reserved only for the semester project. During the project we have had as our main platform of communication #Slack which helped us on reducing the number of "physically" present meetings that required unnecessary effort and generated logistical obstacles.
As our main work framework we have used SCRUM combined with the Unified Process and Work Breakdown Structure.

5.2 SCRUM

When talking about SCRUM it is important to mention one common thing to each sprint, that in the beginning of each sprint the general opinion of the group was that "We know exactly how it works and what to do" and in the Sprint Retrospective realizing that everything needs to be changed or improved. To be pointed out that the discrepancy between the beginning of the sprint and sprint retrospective has diminished gradually as we went through the sprints.
A common ground for the scrum is needed to be underlined before going into the specifics.
The roles of the group team was split as follows:

- Levi – Product Owner – he was responsible with ensuring that the Product Backlog is understood and that each User Story is achieved fully before considered done. He was the one preparing the Spring Backlog and ordering user stories for each sprint to be achieved. He also ensured and was responsible for a proper communication between the client and the group;

- Gais – SCRUM Master – he provided support for both the Product Owner and the Development Team in different forms, as well as took care of the Burndown chart to ensure a proper time management ;
- All members of the group – Development Team;

The length of a sprint was agreed of being of 3 days, with a few exceptions, the amount of work done per day was agreed of being 8 hours/member. The meetings were done before each sprint sometimes meeting in person other times on #Slack depending on the needs and possibilities

5.2.1 Sprint Planning

Each sprint planning, even if not expected, was different. In the beginning we had the tendency to overestimate our work capabilities, meaning that a good chunk of the tasks were not achieved. This was an issue as it was putting more stress on the next sprint, everything needed to be pushed as well as affecting the overall motivation of our group.

It was decided in the third sprint to assign more time per tasks to ensure that it was done. The result of that was that we have underestimated and had to have a small planning in the daily meeting which took unnecessary time. In the fourth sprint we spent some more time into the planning and it paid off, with only a small difference between what was planned and what was actually done.
In the end there was a general agreement that even if usually in the planning phase everyone would be “existed” and would rather rush to do some work than plan, spending some addition time on planning has improved greatly the efficiency of the work that was done over a sprint. The planning for each sprint can be seen in the Appendix E.

Figure 1 - Sprint Planning Example

Third Sprint 15 – 20 May

Planning

Keeping in mind the recommendations from the previous sprint’s retrospective more time was allocated per task. It was agreed that in this sprint the user story 7 will be finished up as well as user story 8 and a part of the user story 9 (6/11 tasks).

5.2.2 Product Backlog

The product backlog was developed just before starting the first sprint, at during the project development has served as a main guideline of what needs to be done and approximate time that it requires.

When deciding on how much weight each user story we have used for some of the user stories the agile planning poker. It was used for the user stories that presented difficulties in deciding how much weight it should have.

It is important to specify that even from the beginning of our development using sprint, when the product backlog was developed, it was decided that we will not be able to complete the entire product backlog. Therefore we agreed that our goal is to complete first 23 user stories that would grant the software possibility to create quizzes and it fit more or less the available time.

The product backlog is included in the Appendix F.

Figure 2 - Product Backlog Example

18.	As a teacher I want to be able to select a specific question displayed	45	Done
19.	As a teacher I want be able to accept/reject a question and give feedback if needed	45	Started
20.	The system must save the accepted question to a pool ready for committing	35	Not started

5.2.3 Sprint Backlog

For each sprint we would take the next in line user story (if nothing was remaining from the previous sprint) and split it into the main tasks. Most tasks are the same for each user story, it was decided to do that because it provided a clearer overview of how much is needed to be achieved. Each task would have its subtask and descriptions specific to the user stories that were only discussed and sometimes noted on *MiesterTask*, but not registered in the sprint backlog.

Figure 3 - Sprint Backlog Example

ID	Sub task	Item	People	Time	Weight (hours)	Actual Time	Actual Weight	Status
16	As a student I want to be able to submit the question for review			5	8	5	8	
	9	Testing	2	2	4	2	4	Done
	10	Documentation & Comments	1	2	2	2	2	Done
	11	Project / Process Report	2	1	2	1	2	Done
17	As a teacher I want to be able to see a list of submitted questions ready for review(live)			17	45	21	54	
	1	Class Diagram	5	3	15	3	15	Started

To be as precise as possible when calculating the Weight (People * Time) of each task we took into consideration the number of people needed to complete that task. Meaning that for example a small task that requires only half of an hour if all five members must participate consumes two and a half hours on our Burndown Chart. It was decided to calculate in this manner as this is the most realistic way (in our opinion).

On the other hand, we also have a column called Actual Weight (People * Actual Time). The relationship between Weight and Actual Weight were always kept in balance, and are crucial for our Burndown Chart as it describes the amount of work done.
For example in the bellow figure it can be seen that both Wight and Actual Weight (grey columns) are in the end equal to 60, what that means that it was planned to work 60 hours and we have physically worked 60 hours.

	2	GUI (fxml)	2	1	2	2	4	Done
	3	View Implementation	1	1	1	2	2	Done
	4	View Model Implementation	2	2	4	0	0	Not Started
	5	Model Impelemntation	1	1	1	0	0	Not Started
	6	Server Implementation	1	2	2	0	0	Not Started
	7	Database Implementation	2	2	4	0	0	Not Started
Total				**30**	**60**	**28**	**60**	

But because not everything from what was planned was achieved (the tasks 4, 5, 6, 7 marked with 0 hours of actual work/time spent), it would have not been a good practice to register as 60 hours worked in the Burdown Chart. If we would have done so, it would have distorted the actual results that the Burndown Chart represents – Remaining Work.

To avoid this issue it was decided that in the Burndown Chart, in the Planned we would mark the amount of work planned, in this case 60 hours. But in the Actual work, even though we physically worked 60 hours, we will register the difference between what was planned the sum of the weights of the task completed.

Figure 4 - Actual Time Formula

$$Actual\ work\ (Burndown\ Chart) = Planned\ Work - \sum_{Last\ Task\ Done/Started}^{First\ Task\ Done} Weight\ (task)$$

In this way there was a clear overview of how much of the sprint was achieved and how much work is remaining. The Sprint backlogs are included together with the Burndown Charts in the Appendix G.

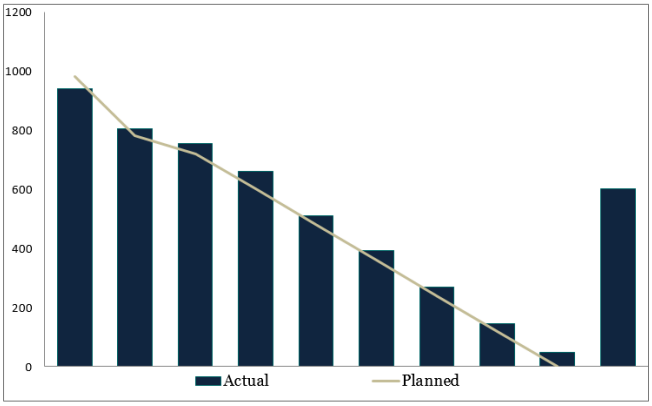
5.2.4 Burndown Chart

Each sprint had its own Burndown Chart which would update the main overview chart. It was decided to work with a separate Burndown Chart for each sprint because it was easier to track and update as needed without modifying the main chart that provided the overall picture. The last column of the Overview Burndown Chart is the approximation of how much work was left to complete the entire software.

Figure 5 - Burndown Chart Overview Example

Burndown Chart Overview

Srpint	Period	Burned down		Balance	
		Planned	Actual	Planned	Actual
	04 of April - 03 of June	980	893	980	940
1st Sprint	04 - 18 April	200	136	780	804
Low Activity	19 April - 01 May	60	49	720	755
2nd Sprint	02 - 15 May	120	95	600	660
3rd Sprint	15 - 20 May	120	150	480	510
4th Sprint	20 -23 May	120	117	360	393
5th Sprint	23 - 26 May	120	123	240	270
6th Sprint	27 - 30 May	120	125	120	145
7th Sprint	31 May - 3 June	120	98	0	47
Approximative Remaining Work					600



5.2.5 Sprint Review and Retrospective

The sprint reviews helped us keep track of what is done and what needs to be added as a priority to the next sprint, if needed. In the sprint review usually everyone would present and summarize what he developed and make a demonstration on the main system, it also served as a good motivation tool to observe the incremental increase in the functionalities of the system.

Figure 6 - Sprint Review Example

Review

Due to the fact that the sprint was cut short for the reasons discussed in the retrospective, only a bit over half was achieved from what was planned. The Login and Register functionality (User Stories 1 and 2) was added and fully functional. It was agreed to continue in the same order and to assign the remaining unfinished tasks for the next sprint. For the User Story 3 only the class diagram was developed in this sprint the user story 4 was not started.

The retrospective was an overview of what was good and bad during the sprint, what needed to be improved and a discussion on how can it be improved. During the sprint retrospective we have agreed on spending more time in the planning phase which was beneficial for the development of the project.
The sprint reviews and retrospectives for each sprint can be found in the Appendix E.

Figure 7 - Sprint Retrospective Example

Retrospective

We have realized that having a 5 day sprint is not efficient and have decided that a 3 day sprint (as recommended by the supervisors) will be more adequate. In general our approximations for the time/task was a bit smaller than the reality therefore adjustments are needed in the next sprint planning for some of the tasks.

There were two “tasks” that weren’t present in the initial planning. First there wasn’t allocated time for discussions and informing about how SCRUM works and time for reviewing and combining all the code. It was agreed that it needs to be fixed in the next planning.

5.3 Unified Process

Each sprint was a done in a unified process manner, meaning that in each sprint we would go through the Elaboration, Construction and Transition phases. Compared to last

semester it felt as we have more control over what it is done and helped us focus more on delivering all round functionalities. It ended in us completing less requirements, but being sure that each finished requirements if fully working and testing, as well as documented and prepared for the project report (Larman, 2005).

5.4 Critique to the project

There are a few elements that were not as good as would have wished to, therefore we consider that it is a priority to mention them so that the future developers of the project are aware of the possible obstacles they may encounter.

5.4.1 SOLID Principles

A good example of violation of the **Single Responsibility Principle** (even if there are more) is the *Model Class*, because it handles a wide range of functionalities for different *View Models*. The main reason that made us violate this principles was not making the class correctly from the beginning and rather saying *let's do it for now as one class and then we are going to separate it*. Unfortunately, when trying to actually split the responsibilities of the class we realized that it is very rigid and fragile, therefore it would take much more time and effort, and as the time was limited we were not able to fix.

Another principle that was violated is the **Open Close Principle**, when designing the software we kept in mind this principle as we were aware that we will not be able to fully finish our product, and therefore should leave space for extension. The general feel of the system is that is very closed and immobile, making extremely difficult to extend without modifying the existing code.

An example can be the very basic functionality of the system *Login*, if for example the management of the Sprogcenter Midt decides that the teachers should be able to log in even when the connection to the server and/or database is lost. It can be done by having a copy of the login information on each machine.

To add the above specified functionality, one will have to adapt/modify the *getCredentials()* class from the model so that it acts as desired.

The reason behind violating this principle, is little experience when it comes to design, as one should be able to anticipate and adapt software for future extension from the design phase.

Because we did not use much Abstraction classes, the **Liskov Substitution Principle** was not violated, but this is not necessarily a good thing, as abstract classes would have greatly improved the software and helped not to violate other principles (e.g. S.R.P violation example).

The **Interface-Segregation Principle** was less violated than others as we used it greatly to be able to work separately by not depending on each other, it is hard to find a concrete example of the ISP being violated.

The **Dependency Inversion Principle** was again violated in a lesser amount due to the fact that ISP was less violated and by the use of the MVVM Design Pattern which ensured a proper flow of control and a counter-directional source code dependency.

5.4.2 DRY Rule

Unfortunately, as simple as this rule may be, in many places of the software it was violated. The Don't Repeat Yourself has a very basic message behind it, whereas when it comes to actual implementation it is very easy to break the rule. A good example where the rule is broken is when loading the *Scenes* for different Graphical User Interfaces, the code is mostly similar for each scene. A good solution would have been making a static class responsible for this.

The reason behind this is again absence of experience in designing the software and the fact that it was designed and/or implemented with an iterative approach.

5.4.3 Database Critique

It was agreed in unanimity that the currently functioning and included database has a poor design and can be improved. The reasons, in our opinion, for a poor database is the fact that when working in an iterative way, we added and improved the database piece by piece. This offered us temporarily functioning database, which is not operating at the full potential and has many design flaws.

To solve this issue a newly design of a database was made, but because it would have taken too much time and effort, and because if the system's rigidity and fragility it was decided not to implement it right now, to be able to offer a demo of the software.
This fact that "it works" breaks is our greatest violation and it more or less is similar with the issues encountered when designing the Model class. A lesson learned from this is described in the conclusions chapter.

5.4.4 General assessment and critique

One fact that needs to be pointed out is the fact that we did not only not fully finish the system, but also did not finish what we assumed we would be able to do (User Story 20). A few reasons can be attributed to this:

- Absence of experience in working with SCRUM – we needed at least 3 – 4 Sprints to start understanding how to work in a scrum way so that it helps us rather than slows down;
- Absence of experience in software development – during the development process many things needed to be researched and understood before we were able to apply, which takes time;
- A relatively small amount of time;

5.5 Group Reflections

When considering the overall reflection on how the process went for the semester project and compared to the previous semester we will point out a few differences in our experience of the semester project.
First we would like to underline that in the beginning we chose to use a Gant Chart (from first semester) for our project management. Nevertheless, as we started working we realized that is it redundant and does not provide valuable solutions for our project development as it fits better a waterfall approach rather than an agile approach, so we did not use it through our semester project.
When comparing the differences between SCRUM and Waterfall approaches of software development it was generally agreed that SCRUM requires more work in implementing it.

On the other hand, the outcomes are very different, SCRUM allowing to adjust easier to the progress as well as provide more all-round solutions for each requirement.
Compared to previous semester, where in the end we have generally cover more requirements, only a few of them were fully developed, most being only partially developed and requiring more work for them perform as desired. With SCRUM we have achieved a lower number of requirements, but all the requirements are fully developed and tested, which contributes to a better overall system.

6. Personal Reflections

In this chapter personal opinions of each member of the group will be presented in regards to the Semester Project in general, to working within the group and other relevant matters.

6.1 Gais El-AAsi

Having in experience the first semester project, I was set from the beginning of the project not to underestimate it, especially which compared to the previous semester this project from the beginning seemed as it requires a lot of work and focus.
In the beginning I was feeling frustrated because everything moved very slowly, with overstretched deadlines and little to do in the allocated timeframe. But I trusted the supervisors which pointed out that the time spent in this phases will pay off in the future of the project.
The analysis was much more in-depth when compared to the previous semester, it required to actually imagine the software and try to interact with it from the outside. It beneficial when developing the Use Cases and Descriptions as it was easier to visualize how the system should perform.
On the other hand, as the SCRUM started everything started to accelerate rapidly. A lot of focus was need to stay on track and keep up with everything going on around. As mention previously, indeed the fact that we invested a good amount of work in the requirements and analysis started to pay off.
When comes to how the group worked I am very satisfied. During the entire process we did not have any major conflicts or disagreements (with the exception of a few when it

came to Sprint planning), which led to a good overall collaboration between me and the group. Similarly to the previous semester, everyone took upon certain responsibilities that felt more natural. It helped a lot that the responsibilities were taken over, rather than assigned as everyone was doing what they preferred and were good at.
Overall, even if not everything that I had in mind for this project was achieved, I am happy with its result. Moreover I am happy with what I have learned during the semester project and I feel like the gained knowledge is applicable for the future projects which is a good thing.

6.2 Leviticus Lyons

Having previously participated in a SEP2 project I came into the project with a clearer idea of what would be required from the group in order to successfully complete the project. However, being new to a group that had previously worked together, what was less clear and perhaps somewhat daunting, was how we would function as a team and where/if I would fit into this. My fears were allayed quite quickly however, as from the first few meetings it was obvious that the group had a solid structure in place from their work together in the previous semester and were also open to the input and perspective of a new member i.e. me.
This paid dividends from the outset of the Inception phase when the group adopted my suggestion of a project for the language center.
Having a client for the project added an extra dimension that afforded us a greater insight into the challenges that would need to be met in 'real world' scenarios. In particular in my role as Product Owner, the process of overcoming difficulties in translating language and concepts from an engineering point of view to that which a customer can understand and also the reverse of interpreting a client's wishes into engineering terms, is something I feel will be invaluable for future projects.
In terms of SCRUM, I found it a much more constructive process than I had on my previous experience, during which it was perceived by the group as more of a hindrance than a help. This time round however, despite some initial skepticism with the slow start of the project, the group embraced it as a useful tool, which I feel is reflected in the product that has been delivered. Moreover, whilst I find it somewhat disappointing that

the project was not completed to the level that had originally been planned, I feel that this is outweighed by the valuable lessons learned and the knowledge gained during the process.

6.3 Marcel Notenboom

The fact that we already had experience from the first semester in working with a project and in the same group gave me (and the group to some extent) the feeling of having a good beginning for the SEP2 (which was not wrong). As we started working, during the first weeks of the project we got introduced to a series of design processes as well as to agile development and SCRUM framework.

Even if in the beginning the group was interested and curious in regards to SCRUM although I had mixed feelings about it also due to the fact that in my previous work experience I have never used this tool.

When we started progressing on the project's development, but still in the beginning phases, I (and some members of the team) had the impression that it had none to little progress. Everything was moving very slowly, mostly that we were working on the concept (Domain Model) of the software and more or less only talked and wrote about what we could possible do.

Our supervisor (SWE lecturer), that also introduced us to the SCRUM framework, underlined that the extra work in the beginning of the project will pay off in the later stages. He was emphasizing that when we will start the sprints, it will help for a more smother progress during the high intensity periods.

This high detailed half abstract way of creating a software system was very tedious and demanding in matter of trying to consider all possible scenarios, it made me feel like no actual work was getting done. Nevertheless when we eventually began on the sprints it became more apparent what the whole analysis (everything up to Domain Model) was helping. Even if the first sprints were very confusing, it was very handy to find many answers in the analysis that was done.

When looking back at our collaboration, we did not have any major conflicts in the group and we seemed to work well together.

6.4 Lukas Vaisnoras

Since the beginning I was confident in regards to the project, as we had the same format as previous semester. We also were joined by another member which had more experience and knowledge that helped us during the semester project but as well with assignments.

This semester we were introduced and asked to use SCRUM for structuring our workflow. When we were introduced it seemed like a relatively simple concept both to understand and apply. But as we started working with it, it turned out to be quite hard and confusing in the beginning but as we were understanding it better it made our workflow more structured. It also helped us plan and delegate what needs to be done as well as made us be more efficient, it helped me a lot to stay focused and be more demanding with myself.

All in all the collaboration with the group members was good, we were able to efficiently communicate with each other and I consider that contributed a lot for a *conflictless* work through the entire project.

6.5 Deivydas Zibkus

I was generally satisfied with the development of the semester project, the SCRUM gave us some headache in beginning but as we figure it out it started to be very demanding but efficient.

The communication and collaboration of the team members was nice, even during the debates or “negotiation” times, everyone acted in good faith and with common sense.

All in all I am more than satisfied with the development of the semester project.

7. Supervision

When it comes to supervision, we have decided from the beginning that consulting with the supervisors is a priority, therefore every time we have had confusions, uncertainties and obstacles, we would consult with the supervisors.

Most of the times it was not an official, schedule meeting but rather a break discussion during the courses which was very beneficial as it was easy to communicate and it did not involve any preparation (scheduling, etc.).
On the other hand, we have also had two scheduled meetings in regards to the SCRUM process and before entering the timeframe designated solely for working on the project. It was decided for a scheduled meeting as it involved many questions that required for use to prepare and more time for the supervisors.
All in all the help from the supervisor was invaluable, without their guidance both the process and the project would have been much harder, and most probably unachievable.

8. Conclusions

When looking back at the semester project development there are a few main points that need to be underlined before making a general conclusion on the how the semester project went. The mini realizations are both related to group work, time management as well as the used frameworks.

- **SCRUM!** – SCRUM is a very powerful tool, it gives a structure to the work flow, and it helps monitor and assess the progress as well as forces the team to provide finished functionalities. On the other hand it gives the impression that you are doing it properly, after each sprint we would realize that many things were done wrong, or not as good as possible. Fortunately, it provides with a tool – Sprint Retrospective – that allows one to contemplate on how the sprint was, what was good and what can be improved, and it is a priority to use this tool to constantly improve;
- **Unified Process** – compared to the last semester, where everything went in a waterfall manner, working in an incremental way felt to be more natural. It allowed the team to adjust to the encountered obstacles much easier, resulting in solution that fits better the requirements;
- **Version Control and Interfaces** – compared to previous semester, this semester we relied much more on the version control and combined with using the interfaces it was much easier to delegate the work and to work separately;

- **Code still needs fitting!** – this point comes in a contradiction to the previous one, even if we used interfaces and version control to allow multiple members to work at the same time, it is important to understand that the lack of experience in software engineering still makes it hard for the code from different members to fit from the first time, therefore it is important to allocate time for refactoring and fitting different code;
- **Understand that Semester Project is a learning tool** – this point is present also in the previous semester and is very powerful, understanding from the beginning what is the purpose of the semester project helps focusing on the right thing – LEARNING. Even if sometimes we have the impression that the semester project is for providing fully developed solutions for the stated problem statement, the focus is on learning and gaining experience from the process rather than solely focusing on the solution;
- **Act local, think global and do it properly from the beginning! –** this point is a combination of a Marketing Principle (think global, act local) with a common sense principle (do it properly from the beginning). When adding a new functionality or designing a new improvement, even if it is a local addition to the current design or implementation it is important to consider what implications it will have on the entire software. At the same time, only thinking is not enough, clear and final solutions are needed to be done from the beginning, without relying on temporary solutions that will only add more work in the future. We learned it the hard way when deskinning the Model and the Database, as we relied on temporary implementations with the idea that it will be fixed in the future, and when it came to fixing it we realized that it is very hard. The difficulty of the work is not only generated by this issue, but also by the overall rigidity and fragility as well as immobility of the software.

All in all, we consider that the main purpose of this semester project was achieved, we have gained knowledge and experience in working with the SCRUM framework, in understanding the Unified Process and Work Breakdown Structure.
At the same time, we have obtained a deeper understanding of the links between the analysis and design, as well as the design principles and rules that make a better

implementation. Even if we have made mistakes the very fact that we can observe and contemplate upon those mistakes denotes that we have enriched our overall understanding of the software engineering.

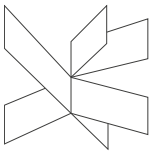
9. Sources

Hofstede, G., 2019. *Hofstede Insights.* [Online]
Available at: https://www.hofstede-insights.com/product/compare-countries/
[Accessed 2019].

Larman, C., 2005. *Applying UML and Patterns.* 3rd ed. s.l.:John Wait.

Schwaber, K. & Southerland, J., 2017. *The Definitive Guide to SCRUM: The Rules of the Game,* s.l.: Creative Commons.

VIA University College, 2019. *Semester Project: Client/Server System Course Description,* s.l.: VIAUC.



Appendices