VIA University College

Project Title - Process Report

Semester Project 3

Group 3

Students

Guillermo Sanchez Martinez (355442)

Piotr Wiktor Junosz (355502)

Halil Ibrahim Aygun (355770)

Alexandru Savin (354790)

Eduard Fekete (355323)

Supervisors

Joseph Chukwudi Okika (JOOK)

Jakob Trigger Knop (JKNR)

Character Count: 18871

Word Count: 3018

Software Technology Engineering

3rd Semester

December 16, 2025

Contents

1 Introduction

The project of a Learning Platform was initiated with the goal of creating an accessible and user-friendly online environment for learners. This project focused on developing a distributed system in a team environment, leveraging various distinct technologies and methodologies to ensure effective collaboration and successful delivery.

---

2 Group Work

Many of our group processes stemmed as a continuation from our previous project experience. Some of the core values that drove our collaboration included open communication, mutual respect, fairness and democratic decision-making.

We based the group contract (TODO: REFERENCE) for the project on our previous contract with slight changes, for example to the point system. However, we did not have to make any major changes as our previous contract had already established a strong foundation for teamwork. This group contract outlined our core values, purpose, expectations, conflict resolution strategies, and accountability measures in the team. In the true agile and democratic spirit, we ensured to have mechanisms in place to be able to revisit the contract and modify it appropriately as the project evolved.

An important feature of the project's process was the role of a Product Owner (PO). The PO was responsible for managing the user stories, prioritizing the backlog, and ensuring that the team was aligned with the project goals. This role was crucial in maintaining a clear vision and direction for the project. Contrary to our previous experience, this role was set to be more settled and less rotating among team members. This was done to ensure consistency and a clear point of contact for the team regarding project requirements and priorities. The role of the PO was assigned democratically at the start of the project, and the team (TODO: COMPLETE LATER)

---

3 Project Initiation

The initial ideas we worked with included a Health Assistant, Bank System, and the Learning Platform. These were chosen as relevant candidates from a brainstorming session based on their potential impact and feasibility. After discussing the vision and scope of each idea, we held a democratic vote to select the final project. The Learning Platform was chosen due to its clarity of purpose and alignment with our skills and interests.

The alignment on the actual vision was more complicated than the initial idea selection. We had to ensure that everyone in the team agreed on the project's goals, target audience, and key features. This involved several discussions and compromises to reach a consensus that satisfied everyone.

One of the problems we faced during the initiation phase was deciding on the actual scope of the project. We were all juggling between thinking about it as an actual ambitious product versus a simpler prototype suitable for the course requirements. This problem was resolved by agreeing on scalable and flexible solutions that allow for both simple approaches and potential complex upgrades.

Another challenge was approaching the problem in a data-driven manner. Considering our mostly European backgrounds, we had to ensure that the project's aims were relevant and applicable on a global scale by researching, and not relying solely on our personal experiences and assumptions. This also included exploring various perspectives and shifting some of the focus on minorities and underrepresented groups in education. This aligned with our mission to improve the education, rather than just having business goals focusing on making profit off of profitable majorities.

---

4 Project Execution

The way of working in the project was a radical shift from our previous experiences. We adopted various aspects of Agile methodologies that fit our desired workflow. This included sprints, regular meetings, pair programming, Kanban but also our own adaptations to these practices.

Our main workflow was initially communicated and written down in the (Appendix A: Project Guidelines). This document served to outline our processes and roles but also aligned us on semantics and definitions of key concepts. Even though this document served as our aim and theoretical perfection we could aim for, perfect adherence was not always possible and advantageous.

One of crucial aspects of our work was embracing asynchronicity, which might contradict with our vision of pair programming but was seen collectively as the means to achieving better productivity and work-life balance. Asynchronicity was also implemented on a level of subgroups - fx. different feature groups would collaborate within the group in real time while asynchronously reviewing and agreeing with the rest of the team on other issues.

4.1 Before Project Period

Because of daily school and work commitments, we mostly worked asynchronously in this period. Our sprints were one week long, starting and ending on Wednesdays. Each sprint would start around lunchtime as we wanted to have time to end the previous sprint collaboratively. Each sprint would start with a planning meeting where we would discuss the goals and tasks for the sprint. At the end of each sprint, we would review all features and tasks, and discuss any blockers or challenges faced during the sprint.

4.2 Project Period

Our work during the project period looked as follows:

4.2.1 Week perspective

We met every working day, weekends voluntary individually.

4.2.2 Day perspective

We preferred to work remotely as this reduced commuting time, allowed for more flexible working hours, but also prepared us for future remote work.

Each day started at 8:20 with a daily standup meeting where we discussed for each: - What was done - What will be done - Any blockers

After the standup, we would plan the day's work, which we considered one sprint in this dense period. Based on our framework, the purpose and focus of a sprint is to merely label and agree on which aspects of the Kanban board we focus on during the sprint. Contrary to Scrum, not all focus was put on putting tasks from TODO to DONE, but rather on deciding how far to push each task (allowing reviews to be done in a different sprint and similar).

Depending on the approach of each sprint, we would either be in a group call or split into channels for pair programming. With more iterations, we realized that collective calls were more effective as they allowed for quicker communication and synchronization on key issues; often problems would reappear for multiple people and having everyone listen to the solution saved time in the future.

Each day the calls lasted until around 16:30 with breaks for lunch and short breaks in between. Depending on everyone's situation, some days the work would continue later into the evening/night for those who preferred that.

4.2.3 Feature perspective

Working on a feature was usually for 1-3 people subgroups. The feature was started by discussing the definition of done, breaking down the tasks, and creating a branch for it. Each feature was practically a vertical slice of the product and the work was also often split into vertical slices (in order to reduce interpersonal dependencies). The aim of each feature team was to deliver a working feature compatible with main (which was evolving in parallel) by the end of the feature work. Each feature would have to be reviewed by someone least biased before merging to main.

Working with the team usually involved calling, sketching ideas, and drawing UML diagrams, while researching domain context and processing data from users. On a practical level, we used Visual Studio Code with Live Share for pair programming, GitHub for version control, Figma/FigJam for brainstorming and unrestricted diagramming, PlantUML inside VSCode for quick UML sketches, and Discord for screen sharing and communication.

---

5 Personal Reflections

5.1 Guillermo Sánchez Martínez

(TODO: Write a reflection for yourself)

5.2 Piotr Junosz

This semester, our group decided to explore new methods and frameworks so that we can learn more about different types of software development processes. The idea was to maintain working in the iterative, agile approach, transiting from strictly using Scrum with Unified Process (UP) to Kanban workflow while retaining Product Owner role which was found useful in the previous semester. For me, the most significant aspect resulted from changing the way of working, more precisely planning and splitting the tasks within the group. We moved from a push-based planning model, in which the work is assigned upfront in the beginning of each sprint, to a pull-based system using our Kanban board. The efficiency of completing and splitting the tasks improved greatly due to the fact that after each group member finished their work, they could immediately pull next tasks from “To Do” column. Those experiences can be explain by comparing push and pull theories, which confirms the success of our Kanban approach. A Push System relies on predetermined scheduling, where “work items are scheduled based on deterministic planning” (Kanban University, 2022). In comparison, Kanban uses a Pull system, where the process is controlled by available capacity. The Kanban Guide explicitly states that in a pull system, “completed work is regarded as more valuable than starting new work” (Kanban University, 2022). Our new team rule - that everyone has to commit all changes before taking a break - was designed to support the flow and improve efficiency of completing the tasks even more. Because of this policy, unnecessary waiting for a group member to finish they assigned tasks was avoided and everyone could just take over the unfinished work. Having experienced the fluency of the pull system, I now realise that self-organized pulling work based on capacity is far more efficient than the push system in the Scrum + UP setup. This successful transition showed us that there are a lot of different frameworks and methodologies to try out in the future development work. Understanding each new process method while comparing to the ones already learned is essential when it comes to choosing your own best and most efficient approach. As a next step, I would like to know more about defining clear process rules as well as learning any new agile frameworks that can improve group work efficiency even further.

5.3 Alexandru Savin

(TODO: Write a reflection for yourself)

5.4 Halil Ibrahim Aygun

(TODO: Write a reflection for yourself)

5.5 Eduard Fekete

This semester brought the best and the worst of experiences. To start on the positive side, I believe that the way of our working was an extremely powerful blend of industry standards that was created after our team debates. I dreamt of something like this for a long time, every time listening to Uncle Bob, Kent Beck, Martin Fowler and other great guys, I just thought about how exactly it could work for me, how could I arrive at some satisfactory way of functioning that would be easy to get people into, that would not make VIA complain, and that would actually make sense.

And here it was, the nice chaos of XP with the structure of Kanban and benefits of Scrum. Although, not to idealize - many people absolutely avoided pair programming. Once when I suggested "let's do it together, can you share screen?", the answer was "Nah, it's super easy, I'll just do it and you check it afterwards". And yes, that defeats the purpose, this wasn't 100%. And it wasn't super easy.

It also really got me when someone would "review their own changes", and in an attempt to redeem "refactored the code" and funnily enough, all this without even running or contributing to the tests.

However, claiming that the process failed because of a lack of adherence and expertise puts us back to the waterfall times, where an argument of type "You should have thought about it in the Analysis phase" would be the way to justify why waterfall works rather than addressing its flaws.

And so the question arises - *what was it that failed and what was it that worked*?

In order to look why things worked, let's look at what worked. We were able to efficiently and quickly deliver features, with not much bureaucracy and sweet sweet agility. Disputes of what and how to do were replaced with getting data, shared code ownership, and a bunch of refactoring.

What didn't work? I believe that our bus factor was at most 2, if not 1. We absolutely lacked a shared understanding of crucial aspects despite tons of accessible documentation. And there it was - "tons" of documentation maybe failed us.

The group contract is another chapter of the story itself. Karaoke never happened and was not even a proper punishment to begin with. I never liked the idea of involving supervisors or any such external party in a position of authority in the SEP conflicts because it just felt like cheating to get conflicts resolved by someone else. Now I would not go into a project without signing something saying that if someone decides to not show up absolutely for anything, and if they lie about their work, they will have real consequences besides "singing karaoke". I cannot believe my *naïvety* of thinking that easing the rules from last semester would just work like magic, I think that freedom is something more complicated than just removing rules.

And this connects to the bus factor again - this system worked better than the previous Scrum + UP one, but was it because we were more productive or because suddenly without any fairness/equality concerns of sprint planning, there was full power for a couple of people to do the work while the rest could ease? Did the lack of code ownership mean a lack of responsibility over the system? Is it in any way fair if someone has more responsibility over the system *purely* because they put more effort? And actually, did I bite myself on this because I predicted a lot of this in my head and thought that having no constraint of taking on the work would somehow only improve my efficiency without having a negative impact on the rest of the people?

To bridge the gap between speculative illusions of reality and the more refined reflection of it, let's form some facts:

- the group contract was broken often, consequences were not enforced
- the methods were more of an inspiration rather than a strict framework
- promises were broken, assigned work was not done
- transparency and honesty were often lacking (e.g. AI usage)

And with a weak proof or rather subjective anecdotal evidence, if these facts hold true, then I claim that none of the previous/other processes would have worked. I believe that a team that does not have a good sense of responsibility, honesty, and transparency can not achieve good results with any normal process; with normal including nothing that would break the law, ethics, ideally common sense as well.

Nevertheless, I would not be able to claim that I am data-driven without proper objective measurements. Therefore, I propose that if the process of judgment is fair, then the grade should reflect the quality of my work on semester project, not my subjective talks in this report. And without such judgment, I could merely conclude and reflect that my contribution was not perfect but never expected to be perfect, though of high quality, consistent, and conducted in a professional and ethical manner.

And to reflect further, I would like to say that I haven't felt such pride in a long time as when seeing some of the team members just crushing the whole SEP aspect of the project. Conducting an hour-long interview on a call because we needed data, studying how kanban works, speaking up on big issues, and even going to "prison" just to lock on a task because that's what we agreed to do are in my opinion the moments that will shape this team into something more than regular Software Engineers. I hope that people who find themselves in this understand the transformation that happened during the semester in the same way I see it.

---

6 Reflect on Supervision

In the project team, we were often aligned on ideas and approaches, which made critical analysis and reflections on our work more challenging. The role of supervision thus became a crucial element of our project process. The supervisors often challenged our assumptions and decisions, ensuring that the project was kept improving based on proper processes and data, rather than just group consensus and momentum.

Each of our supervisor meetings had a clear agenda - acquiring external feedback and clarifying doubts. We did not focus on mere presenting of our progress but rather on seeking constructive criticism and guidance. This approach allowed us to identify potential pitfalls and areas for improvement that we might have overlooked without the supervisors.

Another aspect of the supervision was filling the technical gaps in our knowledge and project requirements. The supervisors always helped with long-term planning which could not have been done by the team alone due to the limited knowledge towards the beginning of the project.

(TODO: COMPLETE LATER, maybe something about frequency of meetings, communication outside meetings, etc.)

---

7 Conclusion

The project showed how flexible structured team collaboration works through our transition from Scrum to Kanban pull-system workflow. The experience we gained helped us develop a list of best practices and common mistakes which will guide our future group projects.

The team needs open communication and respect as fundamental values together with a Product Owner who manages the vision to achieve successful teamwork. We should keep using standup meetings to exchange information while supervisors need to provide essential feedback and help with technical issues and code submissions must happen regularly to avoid creating bottlenecks.

It would be a good idea to avoid three main mistakes which include making decisions based on assumptions when data exists, enforcing processes that block work and splitting into groups that do not collaborate well with each other. The team should focus on achieving fundamental goals before attempting to tackle complex problems.

The team needs to keep their group contract active while using Kanban pull-based workflow management and feature decisions must be based on data. The team should seek outside feedback to confirm their project's direction.

---

8 References

Kanban University. (2022). The official guide to the Kanban method (Version 2). Mauvius Group Inc. https://kanban.university/wp-content/uploads/2023/04/The-Official-Kanban-Guide_A4.pdf

8.1 Appendix A: Project Guidelines

Can be found as ProjectGuidelines.pdf