COURSE NAME COMP 2211

Deliverable 5: Final Report

Team 42

Introduction

For our Software Engineering Group Project we selected to complete the AD Dashboard task. This involved creating an application that could take data and make it easy to read for our clients. This task made sure we worked closely as a team and with our supervisor/client in order to make sure all the boxes were ticked and the client was satisfied with our final product and bi-weekly increments. In this report we will describe and evaluate how we worked together and made decisions like we did to produce the content that we did.

Evaluation of Teamwork

Teamwork is a very complex process and you can never be sure how it will end until you reach the finish line. Members of team number 42, which I was a part of, did not know each other before we started the project, so it was a mystery how we would adjust and whether it would give a positive result. Five complete strangers joined forces in a common cause - the success of the project.

In the beginning, even after we met on Facebook, we gathered for our first meeting in the Zepler laboratory to get acquainted and decide on which project we would work on. We made the unanimous decision to work on the "Ad auction dashboard" and arranged to see each other again in order to start the actual work.

The evaluation of teamwork is quite subjective and what determines whether it is a success or failure is the final result of the project work in general, but to be honest, the experience we gain and the steps we take down the road of completing project work, are a thousand times more important. Our team was united all the time and worked at full speed. We were

like a well-kept machine – there was a problem, someone was immediately there to assist. When some of us had difficulty, we didn't hesitate to ask for help and we knew that there was always someone from the team who was ready to help. Therefore, we were able to work quickly and efficiently. From the very beginning of one SCRUM meeting, we decided to get together and do our job. In this way, we became and felt like a real team.

To be successful in a job, you must know how to work with people who were not familiar to you before – from a different cultural background, with different interests and priorities in life. When you come across these things, you realize that every success requires compromises and hard work. In order for the work to go in the best way, the team must build a plan that works and assess the strengths of each member so that their contribution is optimal. That is exactly what we did. Everyone said what they felt most confident in and we used it skillfully. We never felt pressured by time. The approach, for each increment, was different, we considered it carefully. For our first submission, everyone did their part and then together we edited everything so that it could present our capacity to the maximum. When it was time for the coding part in team 42, two sub-teams were formed, one consisting of Dylan and Palak and the other by James and Vasilena, and William was the main link and support between the two. The whole process was conducted mostly on-ground as we reserved rooms on campus so we could all be together in one place and get our questions answered instantly. Of course, there was still work to do at home, but definitely not much. It is interesting that as a team of complete strangers, we did not have any disagreements or disputes - we sat and did our job without a drop of boredom.

This whole project for us was a very valuable experience that developed us not only at the educational level but also socially.

We can boldly say that everything went well. We fulfilled our goals, we worked in a pleasant, calm and inspiring environment. We are pleased with the end result and the fact that we found new friends. We are satisfied with everything we did. To the question of what could have been better, I think we cannot give an answer because everything worked according to plan. Communication between us was excellent, the feeling that we are responsible for the other people in the team made us more organized and motivated. Meetings with our supervisor Tom, helped us a lot to understand what was not clear. He, in his role as a client, gave us not only requirements but also constructive criticism. The only thing that could be

better, however, it did not depend on us or anyone else is the global pandemic situation, which scattered the united team around the world and made our work slower and less fun. Nevertheless, this did not affect its quality in any way.

The advantages of adapting an agile methodology were sufficiently many. For example, pair programming, which we already mentioned before, made the process faster, prevented any bugs and made sharing of knowledge across the team members easy. Burndown charts prevented procrastination and helped us stay on track. We got a better understanding of the requirements because of User stories combined with stakeholder analysis. Another advantage is that you can get immediate feedback, which also improves team spirits. Unfortunately, like everything else, agile methods also have downsides. The biggest that we experienced was the time-consuming one because every method requires more time and energy from all team members that are involved.

We certainly used XP values. They are not rules but rather guiding points for achieving harmony in the team. We will pay attention to the main 5:

Simplicity was not something we particularly stuck to since we decided to add additional tasks to the ones already given in the specification of the course assignments, but on the other hand, we took small and safe steps to protect us from bugs and make us proud.

The communication we certainly strictly followed. We held daily meetings and we all worked together on everything from requirements to code. Everyone gave an idea, we discussed each proposal and chose the best option.

Feedback - we listened to what our supervisor was telling us at our weekly meeting. During our presentations, one of us, James, usually kept notes of what the second supervisor noted so we could follow up on it. We also carefully read, analyzed and drew up an action plan for each email sent with feedback.

Respect. As we have already mentioned in our team, tolerance and support were the driving force. No one ever feared the reaction of others and we treated each other with the appropriate respect and readiness to immediately help if needed.

Courage was the value we adhered to, striving for success and achieving our goals. We refused to accept anything other than success. In case of any unexpected change, we always looked at the positive side of things and turned it into our plus.

Time Expenditure

Time management is a key part of any project. Over the course of this project, it was especially important to keep organised and to make sure we had a clear idea of what we were doing and how long each part of the project would take.

Average Expenditure

It is difficult to say how long each person spent on average as it is not entirely consistent. Earlier on in a given increment we would slowly work on the biggest tasks for that increment. After that, we may find ourselves completing the smaller tasks whenever we would find time. We obviously needed to balance our workload to be able to complete our group project and any other university work we had to do. Sometimes that may result in days where we don't do as much on the group project, and other days where we complete many tasks at once. If we were forced to estimate our average expenditure, 10 hours a week would seem a good approximation.

Expense of Tasks

For the most part, our expenditure was fairly equal. However some tasks obviously took more work than others, potentially due to a roadblock for a given task that may take time to overcome.

One notably expensive task was designing the SQL statements. There was a lot we needed to account for in the long run when designing our queries (E.g filters, bounce rate, graph types etc). Because of this, it took time to get them to work for the many different things we needed them to do all at once.

Another expensive task was directly comparing tasks from multiple campaigns, as this required a lot of work to already be in place before we could correctly begin. Because of

this fact, we were very careful to design our system to work with this functionality long before we actually started properly implementing it. Things we did included designing the GUI and parts of the backend to work with multiple campaigns. This greatly helped us when we came to add this feature.

We ran into roadblocks for many of the tasks which increased their expense. However, with a bit of extra work, we were able to successfully stay on schedule.

Success of Our Estimations

Looking back at our sprint backlogs for each increment and comparing them with the final spring burndown chart, we were fairly close to a correct approximation for what tasks were going to take the most time. Tasks we deemed to be “large” normally were not completed until at least halfway through the increment. Smaller tasks on the other hand tended to be completed very soon after we started them.

We knew that we were not going to be able to predict every task's expenditure perfectly. We made sure to use keywords to describe the time it would take to complete a task so as not to be too specific. Our use of keywords instead of numbers (small, medium, large) meant we were able to be more flexible with our predictions, prioritisation and task distribution.

Balancing the Workload

Balancing the workload was always tricky as we could never be too certain how long things were going to take, and whether or not someone would get stuck completing a task or not.

It was also sometimes difficult to split larger tasks. Sometimes it may simply be easier and more efficient to have one person working on one thing, no matter how big that thing is. Because of this, our individual workloads were not always perfectly balanced.

For the most part however, we did a good job of making sure to all chip in when we felt like we should do more. Because of this, none of us ever felt like we had far too much to do. Our method of task distribution proved to work, so we stuck with it for all 3 increments and were able to successfully meet our deadlines this way.

Tools and Communication

In order to aid with the development of our application, our group adopted a number of software tools helping to streamline the development process and help us better take up agile methods of group working.

Design and Development

Our Java application was coded by all group members using IntelliJ as our IDE. We used a number of Java libraries in our code, which we managed with Maven, a library management tool based on the POM model. This enabled easy “plug and play” functionality, reducing need for repetitive configuration of every library whenever a group member would import the project to a new system, something we found extremely useful given the current situation where a lot of us moved from using primarily labs computers and laptops to our home desktop setups when we were placed under lockdown. Scene builder was utilised by the frontend team for editing of the design and layout of our application, providing an easy way to ensure a unified style over coding the design without a scene builder.

Collaboration

The main tool we used for our collaborative effort in coding as a group was our project’s Git integration with Github via IntelliJ’s version control. This proved to be invaluable as a means of ensuring that all group members were working on as recently updated version of the code as possible, helping us to waste as little time possible on fixing and merging differences on our code so that we could spend more time adding features. Even when members were working on the application at the same time, our use of Git made merging changes simpler. Linked to our Github repository was our GitKraken Glo board, a task management tool that was very useful as it allowed every group member to have access to an easy to understand list of tasks and who was assigned to completing each one, even giving members the functionality to tick off tasks when completed. This had the added bonus of providing useful screenshots showcasing our task distribution within the group that we could show off in our reports for each increment. Finally, Google Docs was used as our storage and sharing centre for all other documentation and files that our Github

repository did not make sense for, including increment reports, screenshots and charts for said reports, and even our demonstration videos that became necessary for the post-Easter-break online presentations.

Communication

Perhaps the most important part of this whole process was our communication as a group, which we facilitated with certain software tools. The first of which, Facebook Messenger, proved extremely useful for arranging the majority of our quick scrum meetings and ironing out smaller issues that did not require us to meet in person. The mobile app enabled quick replies from all of our members, so that if we wanted to arrange a quick meeting that was very possible by quickly checking if everyone was free and arranging somewhere to meet. Our other main means of communicating as a team was using Slack a team communication platform with several features we found quite useful. Slack acted as not only another place to transfer files and documents, but somewhere to schedule more formal meetings with the built in calendar functionality. It also gave us somewhere to easily communicate with our supervisor, who was invited to our Slack group, where we formed specific channels, one of which for communicating only with group members, and another where our supervisor was included, so that a separation between group work and supervisor help could be made.

Meetings

As mentioned previously, several of these tools helped our team's strategy for physical meetings. We aimed for two meetings a week at least, the most essential type of which being our short scrum meetings which helped getting everyone up to date on our project and on the same page for the next steps. We tried to also have at least one longer working meeting a week where we would sit down together, sometimes splitting off into our front and back-end teams and sometimes working on stuff that needed attention as a full group. Scrums were generally organised via Facebook Messenger due to ease and speed of replies enabling a level of flexibility important for arranging as regular as possible meetings. Sprint planning and working meetings were generally planned in group work Slack channel, and review meetings were organised with our supervisor using a mix of Slack and email correspondence with our supervisor. We found meetings with our supervisor to be very important in order to clear up confusions we had with our client's requirements.

Unfortunately however, as is to be expected of a group of students, due to members' other commitments and work responsibilities this schedule was not always possible, which is why we found it important to be flexible about this in order not to put too much pressure on any group member. To make up for when regular meetings were not possible, we made sure to try to communicate effectively via our communication channels such as Messenger and Slack to make sure each increment's requirements were completed in a timely manner. This became extremely important in light of the current lockdown situation, where online communication became our only means of doing so. We made sure to have weekly Messenger calls to check in on how everyone's work was going. We found that frequent meetings helped us a group to keep everyone on the same page throughout our design and development process.

Advice to Next Year's Students

Upon having successfully completed the software engineering group project, there is a range of advice we can give to next year's students. The most important advice that we can give is to plan ahead of time. Due to the fast-paced nature of the course and limited time, it is essential to keep track of the tasks that have been completed and have realistic assumptions about the average time expenditure required on the remaining tasks. Make sure that all your design artifacts including the UML use case diagram, class diagram, storyboards, scenarios are well-developed and clear prior to the programming phase for every increment; jumping right into coding without planning can be a waste of time. It is however important to make sure that the diagrams are not explored in too much detail when presented in the report, as this can create confusion for the reader. When generating the sprint backlog for each increment, split the number of user stories relatively equally so that the same load of work needs to be completed at each iteration.

In order to be prepared ahead of time for submission dates, we set our own internal deadlines a few days earlier to the external deadlines and we would recommend other groups to do the same, as this allowed for code review, discussion and integration of our individual sections prior to our meeting with our supervisor. Moreover, we would recommend groups to organize at least 2-3 scrum meetings per week to come together

and discuss their progress, decide which stories are of highest priority and would generate the highest value to the product at that particular point in time.

During the programming phase, it is always wise to design the front-end considering the ways that would enhance user experience the most, such as adding colour highlights to buttons, greying out buttons etc. Moreover, it is important to incorporate testing techniques such as unit testing, code coverage and scenario testing from early on, possibly even the first increment. For effective unit testing, we recommend carrying out tests for all the Model functionalities (in the MVC model) including querying, filtering, defining bounce rate, and calculating outlier values, instead of running the entire code at once to figure out exactly which part of the running code is throwing the error. Additionally, we would advise for students to incorporate user acceptance testing by incorporating hypothetical scenarios involving personas.

It is essential for group members to decide and use the most effective and efficient way of communication to inform other members of their progress. We would advise students to use software such as GitKraken Glo and Trello for task and issue tracking, and a group chat in social media to communicate with others about their progress regularly.

Work together and meet regularly for time to be utilised most efficiently and to create software of good quality. Regular meeting strategies give everyone an opportunity to voice ideas, concerns and share updates. We would also advise for the meetings to be well-planned in advance - choose a member of the team to draft a meeting schedule which highlights all the points that need to be discussed in the increment, and keep some time in hand to discuss (programming/newly-emerged) questions towards the end. During the meeting, keep a record of all the points that have been discussed. Lastly, we would recommend teams to always keep a customer-oriented approach; work with your supervisor to make sure you are delivering what they want and expect, and on schedule. Each customer has different requirements as to how the system must work so you should modify your application accordingly.

To close off, here are a few general tips about working in a group:

- Remember to always show respect for fellow team members, even when your opinions may differ; allow for everyone to communicate their ideas freely

- Contribute equally; don’t rely on a few members to carry the team.
- Do not wait too long for a team member if he/she is struggling; make sure to come forward and help.
- Always accept tasks that you can deliver/choose tasks that will give you the opportunity to challenge yourself; don’t under/overestimate your abilities.

Conclusion

In conclusion we used the above time-keeping and team-working skills to make a positive working environment for everyone so that we could perform to our very best. This was important as we could rely on each other to finish their retrospective parts and report any issues or problems they had immediately so they could be problem solved as a team. This provided an efficient and fast way of knocking through tasks and staying on track with our goals for this project.

Overall we must say that this project was a resounding success due to the fact we met all of our targets that we set out to complete in the envisioning stage and finished up with an impressive project that we can all be proud of. Completing the project however wasn't the only thing we gained out of this module, along the way as a whole team collectively we have picked up some very important skills that can be applied to future group work.