Reviewing the Development of a Flight-Ticket Sales Java Application

PRJ2-Report

Group 11

Jonas Brockmöller

4.Juni.2021

Fontys Venlo

Introduction

During this project, a complex information system was developed from scratch. The project started with the analysis phase and documentation of core business processes. Subsequently, software requirements had to be gathered and specified, under the guidance of the project coach. After the analysis phase, an information system was designed and implemented using common software engineering techniques. This includes an agile test driven way of developing as well as creating a multi layered application with the separation of logic, user interface and database connections.
During the whole project, the focus was on the delivery of high quality and consistent artefacts that are in line with each other. Besides the integration of technical skills and knowledge, the process of working well together in a team is a key requirement for a successful project group work.
The work was always divided in blocks of two week sprints to ensure a well performing agile development style. To keep track of the work the GitHub project board was used and the items were assigned to the group members. After a sprint came to an end, the group reviewed the results and depending on the new knowledge the next sprint was planned.
In the analysis phase the group gained a general overview of the application and diagrams were created to visualize the relations of the entities. The goal was to detect as many problems as possible very early in the project to take measures against it before the implementation began.

GROUP WORK

This project is the result of a group work of in theory six students. One student never joined any group meetings and was kicked out of the group very early. When the analysis phase ended the group shrank down again to a core of three students working reliably. This was the first huge issue for the group. Additionally a slow communication on MS Teams and a lack of leadership set back the project over and over. Because I did not experience this in a project group before I was not prepared for the situation. I learned a lot about group work and the importance of leadership and communication. Because this was the biggest problem I encountered in PRJ2, it will be discussed further in the following paragraphs.

Reflection

LEARNING GOAL 1: ANALYSING THE FUNCTIONAL REQUIREMENTS RESULTING IN A CONSISTENT SET OF DIAGRAMS

The firsts few weeks of the project went great with total participation of all members and great progress each week. The application was analysed over and over and models were adapted to the new discoveries (as it can be seen here).

For the complete analysis phase (LG1) all group members were motivated and excited. We always kept up with the plannings and prepared for possible presentations in the coaching hour. As the first phase was coming to an end the groups' progress slowed down and the motivation was declining. It turned out by the beginning of the design phase, where critical details needed to be discussed and decided for the implementation, that the group was lacking a real leader. Since no-one wanted to take the responsibility for decisions and the consequences, the group faced a big problem. In the first sprint in week six, the task was to create a class diagram and deal with the presentation, application and data logic. By this time the group only consisted of a core of four students working regularly and reliable. The lack of communication complicated the situation even further. All meetings were held online and it was hard to keep an overview of everyones progress. The separation and assignment of the work was not the issue, but most work items are connected to each other and the group members should have communicated more and worked together closer. In this time period I learned that a group with no leader cannot keep up with the progress of the competitors. Also it is better to take a decision, even when it is wrong. Diagrams can and will always be edited before a project is completed. I learned that as long as there are good and valid arguments for something, one should just go with it.

LEARNING GOAL 5: AGILE PROJECT MANAGEMENT TECHNIQUES IN A PROJECT WITH CHANGING REQUIREMENTS

For this project the groups were advised to work in an agile way, by planning the development in two week blocks called “sprints”. Also a project board with all open tasks in the backlog and columns for the current sprint, items in progress, in review or done were recommended. The group payed attention to all of these suggestions. Even when the new theory of CI - Continuous integration was explained in a DVPR lesson, the knowledge was used to perform the same actions in this project. This was especially helpful, because we had a lot of problems within the group with code working on one machine but not on others. The project board in Combination with prioritization labels helped keeping an overview of the current status. The time estimations for each work items were a big challenge for me and I know that this is something I have to improve. I constantly underestimated the work that a task contains or overestimated how fast the group progresses. For future projects I should stick to a more conventional time budget.

LEARNING GOAL 2: THE DESIGN AND THE LAYERED ARCHITECTURE

Because the database was a huge issue in the development, the group agreed to use a temporary Manager class and started working on the layered architecture. This was completely new to all students in the group. Because of this, it took some time and many iterations of the class diagram to get it right. To this day the project consists of Java packages instead of modules, but the layers are clearly separated from each other. Whereas in the beginning I thought this is unnecessary and did not help the project at all, I understand by now how important a separation of layers might be. It makes an application much easier to test and read. This is something I will use for future projects again, especially in combination with the test-driven-development approach. It took a lot of time to get used to the separation of logic and sometimes the correct layer is debatable, but this makes future updates to the software a lot easier. Even though the single responsibility principle was only explained in week twelve, the group quickly adapted to it, because of the multi layered architecture the principle was used in most cases naturally.

THE PROJECT WEEK

In the project week a desktop application with a database had to be build. We understood very early that the way we used to work until then did not bring the expected results. By the end of the second sprint we realised, that we are falling back on the schedule, because the database creation took much longer than expected. Even though the results of the analysis phase were very helpful and showed us the correct way to go, it just took too long to make decisions. After the project week we came together and discussed this issue. It was clear, that everybody had a good understanding of the application and what we want to achieve. Since no one was feeling ready to lead the group we determined that complete use cases should be assigned instead of small work items. This way we hoped to increase the group productivity. As it turns out the use cases had to be split up between the group members anyway a few weeks later. This was a learning process the group had to go through and we learned how important responsibility and leadership can be for a project. This is something that I will definitely try to improve for future projects. Luckily this approach was much more promising, because everyone was implementing their own use case and every week the progress was presented to the group.

LEARNING GOAL 3: THE TDD-APPROACH BASED ON THE OUTCOME OF THE ANALYSIS

In the analysis and design phase usecase descriptions, diagrams, test scenarios and class diagrams were created. These are especially important for the implementation phase. The results from the class diagram are used to build a first template application. Then the test scenarios are needed for each usecase. Based on the test scenarios automatic JUnit tests were written and executed on the project. By writing the tests first we can ensure nothing was forgotten to be implemented and everything works as intended. Many bugs and edge cases are tested automatically this way. For example in the Business Logic tests were written to verify that overlapping routes cannot be used on connecting flight. Moreover the destination airport of the first route and the origin of the second route need to match. Additionally a 45 min time period has to be between landing and takeoff for refuelling and boarding of the aircraft. All of these checks are verified by the tests. Another example is the encryption of the employee passwords. Tests were written first to check if the encryption and decryption works as planned. In this example the tests were also useful later in the project when changes were made by many contributors and suddenly some tests failed. This was a good reminder for us that something is not working properly anymore. Working test driven is a big adjustment and brings a lot of work with it. To be fair I believe that the tests could be much more in quantity and also better in quality. I think that even though there was a big difference in the collaboration of the group members, we tried to do our best. I am aware that there are other groups that tested more and better, but all user interfaces and essential business logic parts have tests in our project as well. In total we wrote more than 175 tests. I can not say that I am very pleased with the amount of tests, but concerning the circumstances it is sufficient.

LEARNING GOAL 4 BUILDING A MULTITIER APPLICATION

An optional requirement was to build a module based application. Because of the time pressure this requirement was not fulfilled by the group. Nevertheless the separation of layers was implemented. The only disadvantage is that modules hide unwanted and internal details very safely for better security, which is missing in the current project structure. For future projects I will use the module functionality, but since it was only optional we decided not to implement it.

SECURITY FEATURES

When talking about security I remember the last PRC2 lecture was about security measurements. Because I had some capacity left, I created roles for the database to limit the table access to the minimum required. In addition a SHA-254 encryption algorithm was implemented for the employee passwords. The encryption was enhanced by adding a SALT. Because of the time pressure the employee-id is used at the moment which is not the most secure way. I had a lot of fun implementing these features and learned a bit about cyber security. This shows that security was also important to the group and to me personally. I think this is a very interesting topic in computer science.