Lesson Learned

Software-Project 2024

Team Members:

Muazzam Bin Aqeel – (Scrum Master)
Aida Zhumiyeva – (Quality Architect)
Wesley – (Product Owner)
Cosmin Constantin Andronescu – (Software Architect)
Keti – (Software Architect)
Ebtehal Aldheeshy– (Software Architect)
Citlalin Reyes Soto – (Software Architect)
Ruth Jesulayomi Mautin Amosu – (Software Architect)

Muazzam Bin Aqeel

"Software Project", I must say, is one of the most important subjects I have experienced in this degree because all the things that I have learned so far in Computer Science were mostly applied in this course.

1. As my main role in the project was being a Scrum Master, this position made me understand the importance of Scrum process and why it is so systematic.
    a. **Applying Knowledge from outside the scope of the university:**
    With my current Working Student Job experience as a Strategic Analyst, I saw how i can apply scrum mythologies in this project which brought a new level of experience for me
        i. For example, why daily short meetings are very important, as due to the fact that the whole team stays connected throughout the Sprint, where I was able to understand the blockers of every team member, and this brought the team closer to each other.
        ii. Sprint Reviews and Sprint Retrospective made me understand in which areas as a Scrum master i need to put more work in for the team to have a more productive Sprint round.
            1. Organizing the Sprint Retrospective made me understand that to have a healthy and transparent relationship within the team, it's very important to get some feedback from the team as well, to increase their motivation and to help them understand the whole Scrum process
            2. As every team faces problems over the projects, having a group of 8 members gave me an opportunity to understand the dynamics and multiple areas of how one should work as team member. In the start we had some problems overall with our processes but i think that is normal, as eventually the whole team grew so much over the period. Most to the team members trusted me with the process and asked for guidelines, that increased my motivation as well to put more than ever for the success of this project.
            3. Team Calibration: With that experience, i was able to give quite some good understanding about the process to the team, and learnt a lot myself how i need to adapt things in different teams

    b. Full Stack Development
    In the project, I also had a strong involvement in the front and back-end development, which i will now elaborate:
        i. Front End
            1. For me QT Creator tool was a big exposure in UI development
                a. Not only was it the tool that i had to deeply get a hold on but i learnt as a student how to read long documentations to solve problems with a more systematic approach.
                b. Applying my C++ knowledge was very satisfying because Programming 3 (A subject taught at the Technische Hochschule Ulm) has been one of my favorite subjects to learn so far and then having a chance to apply that theoretically knowledge in practice was something that I have never experienced before.

i. Concepts like creating my own delegate classes from the existing QT based classes was very satisfying
ii. Learned how to embed Python code inside the cpp code, which made me understand, programming in C++ brings a lot of flexibility
iii. For every Widget that I or my team members had to use, gave us another chance to learn much more in depth how to operate with that specific widget and access it that in different classes while following the trivial concepts of C++
iv. Was able to also optimize the code over the period, learnt how to use Smart Pointers and Move Semantics concepts to increase the overall performance of the software itself
v. Leart how to apply SQL queries in the code and retrieve information
vi. Used STL libraries offered by QT for example QMAP, to make a good structure

c. **Back-End Development**

i. I learned how to integrate drivers, for example, MySQL drivers, into our version of Qt.
ii. I was able to integrate the SQL queries with the correct business logic.
iii. I also learned how to execute multiple SQL queries using transactions to achieve optimum performance.
iv. We observed that sometimes we were using the same queries across different classes, which was creating redundant code. That's why in future projects, it would also make sense to create a class from which we can retrieve those queries that repeat quite often throughout the code.

d. Software Architect / Documentation

i. I learned how to be clearer and more precise with my explanations to the team members and understood the importance of commenting code. Those comments were helpful to my team members in recreating things faster by learning through that code, naturally increasing the effectiveness and the horizon of our development team.
ii. I also learned the importance of having a Quality Architect, as the quality of the project's process and requirements improves.

2. Conclusion

a. Like I said before, this course enabled me to achieve those things in such a short period that I had never experienced before. Due to this experience, I now fully understand that choosing Computer Science as my major was one of the best decisions I ever made. I believe that every team member faces ups and downs in a project, but if one has the motivation to learn more and sets clear goals, then they can achieve wonders.

Muazzam Bin Aqeel
CTS 5"

Aida Zhumiyeva

This project was the largest, both in terms of content and the number of team members, during my university years. It was quite interesting and challenging. Additionally, it allowed me to apply what I had learned before. So, the lessons I learned from this project are as follows:

Importance of Scrum Tools: Recognizing the vital role of Scrum tools in team organization, progress tracking, and meeting project deadlines was a key takeaway. The development of a Scrum tool underscored the significance of our project, enhancing my understanding of its importance.

Collaboration and Teamwork: A fundamental lesson involved learning to collaborate effectively within the team—helping when needed and being open to receiving help from others. This collaborative mindset fostered a more cohesive and productive team environment.

Product Backlog Creation: Contributed to creating an initial Product Backlog, outlining a preliminary and prioritized list of User Stories. This process was instrumental in shaping the project roadmap and setting clear priorities for development.

Diverse Developer Tasks: Undertook various development tasks, including the design and development of UI windows, implementing and editing button functionalities, and designing pages, where I gained basic experience in C++ programming.

Designing Tasks and Utilizing Skills: Engaged in multiple design-related tasks, leveraging my design skills and creativity. This included defining a color palette (including button design), and creating a visually appealing color scheme for our software.

Test Case Writing: Acquired the skill of writing test cases, contributing to the quality assurance process. This step proved important in identifying and addressing potential issues.

Continuous Learning: Embraced a continuous learning mindset throughout the project, adapting to new challenges and expanding my skill set. This flexibility allowed me to take on diverse roles and responsibilities, contributing to the overall success of the project.

Aida Zhumiyeva

CTS 5

Wesley Dos Santos Barisch

This class has provided lots of learning experiences, some of which go beyond what I can explain in this document, but I will try to list the ones that certainly made an impact:

- At the first three weeks of the class, I've learned the importance of software planning/engineering in a software project:
  - By reading the pdfs provided for guidance on the project and the goals of the class, there was an instant feeling of need to revisit the software engineering concepts learned in the previous semester. Since the project would follow a OpenUP + scrum software process development, it was clear to me that those would have to be fully understood to achieve the learning goals of this class. By reading on the topics of OpenUp and scrum, and trying to fit the 2 together, I've learned a lot about what it means to engineer a software. The skills acquired by doing that were very helpful for organization and management of the phases in the software development
  - I've learned how to use, manage and customize a software development tool (JIRA). In fact, I was already introduced to the tool in my workplace, but I never really understood it until I was using it on the project. I've learned so much of how to manage projects in JIRA and how to set workflows, edit data objects, manage sprints, priorities, test tabs that I've managed to help out at work when colleges had questions about JIRA.
- I've learned that creating software goes beyond just tying code in an editor. Understanding the client's requirements, the tools needed, team's needs and process following discipline are some key factors in software development that I would rarely have learned if it wasn't for this class.
  - During the early and middle phases of our project development, there were a lot of challenges regarding those things mentioned above. But the way the team worked together to find alternatives and resolve issues was amazing and provided joy in having those experiences.
  - To be able to develop the software, there was a battle in tearms of preparing the environment and adjusting the workflows. That in it self took some time, but after it was done, work seemed to be very intuitive and the issue rate started to decrease as I got familiar with everything.
  - Much could also be said about the communication skills learned during the class. Our Scrum-Master was definately inspirational in keeping the team together and helping everyone get familiar with the tools. Therefore, by observing how he acted as a Scrum-Master, I've also learned a lot about that role as well.
- As a team, we've voted for the roles of Scrum-Master, Product Owner, Software Architect and Quality Architect. The lessons I've learned from the team members representing those roles, and from taking the role of product owner/software/quality architect myself, were invaluable.
  - In my role as Software Architect, I've learned how much developers can depend of each other to finish the work. In an unorganized process, this would have been a disaster. But since our team was very focused on having our dailies, doing the sprint planning/reviews, the blocker were identified very early and communicated which improved the time it took to solve them.
  - This role also helped me practice the knowledge learned in Programming 3 with c++, as well as understanding the perks for a powerful language like c++. By coding the classes, functions, and projecting the relationships between the classes, I see much improvement in my programming skills.
  - As a product owner I've learned how to plan a software and understand how the requirements connect with the implementation. There is a lot that's expected from this role, and I've learned that documentation of meetings, requirements, user-flows, cases and any relevant information needs to be documented for clarity. Tasks and their descriptions on JIRA need constant maintenance and update. Development of anything in the software requires the person

developing it to understand the domain and how this work fits into the project. Also, requirements change, so everyone had to train themselves to let go of everything we fear to lose.

- Another great lesson on that role is good communication is key. Even though the ability to speak does not make one intelligent, it is still important. In our team we have people from all over the globe and some, as me, do not have English as their main language, so some miscommunication was bound to happen. Since there is expectation on this role to communicate clearly and precisely, I had to focus drastically on improving my communication skills.
- I've became very interested in understanding the product on a deeper level and growing attached to the software. This helped me strive for a better quality in the development, reviewing my and teammate's code to improve and find bugs/memory leaks that were present in the software (thanks programming 3 classes!).

- General lessons
  - By becoming familiar with the software and how it is built, there were some lessons learned on the database topic as well. Improving the tables and relationships, as well as testing the queries were tasks that made me a bit more interested in the database field (which I was traumatized after working in a hospital database software). In the end, I'm confident future database programming working opportunities won't be just immediately turned down by me.
  - I don't like QT. Even though I had a very bad feeling about this, it turned out okay in the end. After having worked with it in this project, having understood at least the basics of QT and how to set the environment, I still find it to be a very unintuitive and non-user-friendly tool. Knowing this, I will do my best to stay away from anything involving QT, I'd just as soon kiss a wookiee.
  - Practical classes are where the gold is at. Nothing beats getting your hands in the mud. Okay, enough of metaphors. I've learned that practicing the knowledge acquired in classes is the way to really learn. I can't remember most of the theory of my best graded classes so far, but I feel like everything I've applied in this project will stick with me, as it made me realize the importance of it, how it fits together!
  - I also learned that it is okay to put some Star Wars references hidden within the code (maybe other places too), people will probably not notice it. Since coding is going to be present in most of my professional life, I probably should have some fun while doing it.
  - Working on a project was stressful and fun. The team was engaged and working hard to achieve the end goal. Dividing work properly and understanding the capabilities/needs of each team member is very important and this lesson I also will carry with me.

Cosmin Constantin Andronescu

The Software Project has been a difficult challenge for me. I have used everything I learned in previous semesters, as well as things I learned outside of the university. It has also pushed me to learn new things and improve my skills.

My main role in the project as a Software Architect made me realize the importance of chemistry and communication in a team, for which the scrum process helped tremendously.

1. **Applying Knowledge from outside the scope of the university:**
   Having some experience with Photoshop, I was tasked with the creation of the Scrummy Logo, Application Icon, Poster, as well as designing the initial Wireframes for the application and creating the User-Flow.
2. **Full Stack Development:**
   In the project, I had a strong involvement in the front and back-end development:
   a. Front End:
   The QT Creator tool was the biggest helping hand in UI development. There were many things done in C++ for the UI, and this meant applying everything I learned and understanding the new things Qt brought.
      i. Understanding and using many different classes provided by Qt makes it possible to create UI elements in C++.
      ii. Creating interactions for the UI, like button connections, as well as other interactive functionalities.
   b. Back-End Development
      i. Most of the hard work was done in the program's classes, in C++.
      ii. Integrating MySQL queries inside C++, to retrieve and update the database.

3. **Software Architect / Documentation**
   a. Learned to be clearer and more precise with the code and using a lot of comments.
   b. Learned about the Scrum Process and Qt framework.
   c. Improved my knowledge in C++ and MySQL.
   d. Learned to schedule work more efficiently and accordingly with other team members.
4. **Conclusion**
   The Software Project course has taught me a lot about team projects and the scrum process, and introduced me to new tools like Qt. I managed to include skills and knowledge I had and improved them.
   This project has been a valuable experience for me, and I am thankful to everyone for their support.

   Cosmin Constantin Andronescu
   CTS 5

Keti Hazbiu

This project was challenging for me and also it was an impetus to learn many new things. Apart from learning I was able to put into practice what I was learning during those two years of studying. I will mention some lessons I learned from this project.

Implementing the Scrum process emphasized collaborative teamwork, adaptability, and continuous improvement. Regular communication and iterative development allowed us to prioritize tasks effectively, leading to disciplined and efficient time management.

Wireframe Color Palette and Button Design: Understanding the importance of early collaboration between design and development teams for a cohesive user interface. Balancing aesthetics with functionality is key.

Design and Development of UI Windows: Participating in the design and development of UI windows highlighted the necessity of a seamless and intuitive user interface. Learning to balance functionality with user-centric design was crucial, emphasizing the importance of user feedback in refining and optimizing UI elements.

Full Stack Development: Using Qt Creator tools for the front end, integrated with C++, made the initial phase relatively straightforward. Working with C++ and SQL helped me a lot to revise things I know and to learn new things.

Lists of Non-functional, Technical Constraints: Identifying and addressing non-functional constraints early in the project lifecycle. The importance of anticipating challenges and mitigating risks.

Documentation: I had several tasks related to documentation and this helped me to learn and at the same time to practice how to explain in a more exact way without going on too much.

Researching: I also had a lot of research tasks which helped me a lot to learn a lot of things I never knew before. In summary, the project taught the importance of collaboration, adaptability, security, and customization in software development. It emphasized the need for clear communication, thorough testing, and continuous improvement throughout the development process. Each task contributed to a holistic understanding of effective project management and execution. These lessons will undoubtedly influence my approach in future projects, emphasizing a holistic understanding of both the front-end and back-end aspects of software development and also emphasizing Scrum process.

Keti Hazbiu
CTS 5

Ebtehal Aldheeshy

1. **Introduction:** This project aimed to develop a robust, scalable, performance-optimized database system tailored for a complex application. The focus was on efficient data management and security, with a seamless integration into a Qt C++ application. From concept to execution, the goal was to build this database from scratch, ensuring it met the high demands of both functionality and reliability.

2. **Database Design:** The project underscored the critical importance of detailed database planning. I enhanced database efficiency by analyzing query patterns and carefully selecting data types, especially under high-load conditions. This approach not only improved performance but also ensured scalability.

3. **Building Relationships Between Tables:** I learned that adequately defining relationships between tables is crucial for maintaining data integrity and enabling complex queries. By strategically using foreign keys and joining tables, we eliminated data redundancy and improved retrieval accuracy. This was pivotal in ensuring our database could handle complex data interactions while maintaining consistency.

4. **Learning to Use Workbench:** The experience highlighted the value of database tools like MySQL Workbench. This tool was instrumental in the design and management phases, offering a graphical interface that simplified schema modifications and expedited troubleshooting processes.

5. **ER Models and Database Engineering: Employing** ER models was a game-changer in visualizing and conceptualizing the database structure. These models served as a blueprint, making our database development process more structured, efficient, and aligned with our goals.

6. **Database Testing:** Testing was vital in ensuring the database's reliability and performance. We implemented manual tests and conducted regular load testing, which helped us maintain a high database performance and reliability standard throughout the project's lifecycle.

7. **Server Integration and Connection to Qt C++:** One of the project's significant achievements was successfully integrating the database with the server and the Qt C++ application. This step was crucial for creating a fully functional system and posed several challenges, particularly in establishing a stable connection for real-time data processing.

8. **Challenges and Solutions:** We faced a significant challenge in ensuring data consistency during concurrent access. To address this, we implemented effective transaction management and locking mechanisms essential in managing simultaneous operations and maintaining data integrity.

9. **Key Takeaways and Future Improvements:** The project reinforced the importance of continuous learning, especially in areas like database optimization and security. We plan to explore more cloud-based database solutions, aiming for enhanced scalability and performance in future projects. This experience has laid a solid foundation for our database design and implementation team, equipping us with valuable insights and skills for future endeavors.

Citlalin Reyes Soto

1. Database Design and Construction Using MySQL Workbench:

Learning: I gained valuable hands-on experience building a database from scratch using MySQL Workbench.

Application: I was responsible for successfully designing the database structure, which involved defining tables, fields, and data types.

Reflection: Through this process, I realized the significance of careful planning in database design to ensure scalability and efficiency. In future projects, I aim to employ even more detailed initial planning.

2. Understanding Relationships and Dependencies in Database Tables:

Learning: I developed a deeper understanding of establishing relationships and dependencies among tables.

Application: I implemented these concepts in our database, which was crucial for maintaining data integrity and optimizing queries.

Reflection: This experience highlighted to me the vital role of database normalization in reducing redundancy and enhancing data consistency.

3. Utilizing Reverse Engineering for Entity Relationship Models:

Learning: I acquired and applied skills in using reverse engineering to construct and modify the entity-relationship model.

Application: This enabled me to adopt a more dynamic approach to database design, adjusting as our project requirements evolved.

Reflection: I found this approach extremely valuable for visualizing and refining database structures, and I plan to use it in future projects.

4. Creating and Connecting a Server on AWS:

Learning: I learned how to set up and manage an Amazon Web Services (AWS) server and connect it to our database.

Application: I successfully hosted the database on this reliable and scalable cloud platform.

Reflection: This experience gave me insights into the advantages of cloud computing for database hosting, particularly regarding scalability and accessibility.

5. Integration with QT for Database Connectivity:

Learning: I mastered connecting our database with the QT framework.

Application: My work enabled efficient communication between the front-end and back-end systems.

Reflection: I realized how crucial this integration is for the functionality of an application and the importance of understanding the interplay between different software system components.

6. Database Maintenance and Backups:

Learning: I acquired skills in performing regular database backups and maintenance tasks.

Application: I was responsible for ensuring data safety and database health throughout the project.

Reflection: I now recognize the critical importance of regular backups and maintenance in preventing data loss and maintaining data integrity.

Ruth

This project has been one of the most interesting things in the semester. It was a great change from applying all what I have learned in almost two and the half years in this project.

My main responsibility was System Architect, which at first, I thought i knew was it was, however after intense work in this area, i can say I have more experience than ever.

1. Application of knowledge from outside the scope of the University

It was more or less of work management, being constantly accountable for what I did. Working with teammates and trying to be understanding from their perspective. Being the voice for some team member and always taking the intense to question thing that i didn't understand rather than shying away from it.

2. Full Stack Development:

Working in this aspect had its ups and down, while they were constant change of requirement, they were also the aspect of even trying to comprehend the changes to be able to apply which usually takes hours:

a. Frond End:

The Qt creator tools was the main basic for the front end, which was integrated with C++, which almost i had to understand the Ui which I must was the easiest part because it was more or less dragging widget tools in the right position and renaming proper, which was an interesting way to start to the project because afterward it was downhill.

b. Back-End development:

Working with C++ with multiple classes and understanding the need for FURPS + was a change gamer for me, because at the beginning I was so lost within the classes. It also allowed me to understand C++ as a language better and power it is. The use of SQL was like a wakeup call, because I haven't touched it for like 2 years.

3. Documentation:

The constant need for documentation was very important because it gave everyone an idea of what it was, being precise was something to learn while documenting and not just writing long words. Code been comment help elevate more coding that i could always reuse code and understand what i was reusing.

4. Conclusion:

In Conclusion, having to work with the scrum methodology from working alone was quite difficult at the beginning but with time it got easier and interesting that in another project of mine, scrum methodology was also used. This project has taught me a lot, even better than what I have learned in theory. It taught me how to work with time and how to be able to defend my work. In all in all, i will say working on a project with team members is more effective than learning theory.