Student Name: Masoumeh Farokhpourshekalgourabi

Student ID: 40309733

Course: SOEN 6841 - Software Project Management

Journal URL: https://github.com/MasoumehF/SOEN-6841-Software-Project-Management

Week: Final Reflection

Date: 30th March, 2025

Final Reflections:

Overall Course Impact:

Before taking this course, I had a general idea of what software project management involved, but I didn't realize how structured and detailed the process really is. This course opened my eyes to the many stages and components that are essential for a successful software project. From initiating a project to releasing and maintaining the final product, I gained a much clearer picture of how everything fits together in real-world scenarios. Software project management is an important part of the IT and software industry, shaping modern practices and influencing the global economy. Through this course, I gained a solid understanding of the key areas involved in managing software projects effectively.

Topics Covered throughout this course:

- Introduction to Software Project Management
- Project Initiation Management
- Software Project Effort and Cost Estimation
- Risk Management
- Configuration Management
- Project Planning
- Project Monitoring and Control
- Project Closure
- Introduction to Software Life-Cycle Management
- Software Requirement Management
- Software Design Management
- Software Construction
- Software Testing
- Product Release and Maintenance

Summary of key topics I learned:

- Chapter 1: Introduction to Software Project Management - Introduced the definition and nature of projects and how software projects differ from other types. Emphasized the importance of planning, scope definition, and structured phases like initiation, monitoring, and closure. It also highlighted the responsibilities of project managers and the difference between roles such as leader, manager, and scrum master.

- Chapter 2: Project Initiation - Taught how to formally begin a software project using a project charter and scope statement. This chapter focused on defining SMART objectives, estimating initial budgets, and scheduling. It also presented project division methods for more accurate planning and stakeholder engagement.
- Chapter 3: Effort & Cost Estimation - Provided multiple estimation techniques including Function Point Analysis, Wideband Delphi, and COCOMO. I learned how to use past data, judgment, and algorithmic models to predict labor hours, costs, and schedules. It also taught me the challenges of uncertainty and how iterative estimation can improve accuracy.
- Chapter 4: Risk Management - Explained the process of identifying, analyzing, and prioritizing risks. Covered qualitative and quantitative assessments, and strategic responses such as mitigation, avoidance, transference, and acceptance. Emphasized the need for continuous risk monitoring and use of buffer time and contingency plans.
- Chapter 5: Configuration Management - Focused on managing software changes and versions. Introduced key components like configuration identification, version control, status accounting, and auditing. I now understand how to ensure traceability between requirements, design, code, and test cases—reducing rework and defects.
- Chapter 6: Project Planning - Explained how to structure a full project plan including WBS (Work Breakdown Structure), scheduling (top-down vs. bottom-up), resource allocation, budgeting, and communication planning. Tools like Gantt charts, PERT diagrams, and Critical Path Method (CPM) were introduced.
- Chapter 7: Project Monitoring & Control - Taught methods for tracking project progress using performance metrics. Introduced Earned Value Management (EVM) to assess budget and schedule variances. Discussed strategies for taking corrective action, managing resource loading, and handling unexpected project deviations.
- Chapter 8: Project Closure - Emphasized the importance of completing all project work and deliverables. Discussed the archiving of documents, analyzing metrics, and documenting lessons learned to benefit future projects. Introduced the practice of final audits and formal closure activities.
- Chapter 9: Introduction to Software Lifecycle Management - Introduced various software lifecycle models—Waterfall, SCRUM, XP—and discussed their suitability based on project needs. Explained concurrent engineering, work products at each stage, and quality gates. Also introduced Capability Maturity Model (CMM) to assess process maturity.
- Chapter 10: Software Requirement Management - Focused on capturing and managing customer requirements. Differentiated between functional and non-functional requirements and stressed the importance of validation and change management. Covered the full requirement lifecycle and common flaws like ambiguity and omission.
- Chapter 11: Software Design Management - Described how high-level and low-level designs are created to meet requirements. Covered design strategies like modularity, abstraction, and architectural patterns. Stressed the role of design documentation for guiding development and testing phases.

- Chapter 12: Software Construction - Focused on actual coding practices, use of programming standards, and integration of code modules. Covered version control, code review practices, and the importance of maintainability and readability.
- Chapter 13: Software Testing - Explored testing methods such as unit testing, integration testing, system testing, and acceptance testing. Covered test planning, defect tracking, and the role of automation in improving software reliability and reducing time-to-market.
- Chapter 14: Product Release and Maintenance - Addressed post-deployment activities including user training, bug fixing, performance monitoring, and long-term maintenance strategies. Emphasized the importance of customer feedback loops and adaptability in maintaining product relevance.

This course focused on the need for a structured and collaborative approach to software project management. It demonstrated how aligning project management with software engineering practices leads to better outcomes. The course also highlighted the importance of effectively managing teams, technologies, client expectations, and vendor relationships to ensure project success.

Application in Professional Life:

- Our topic discussion on “The Role of Continuous Kindness in Projects” really stood out to me. It showed me that being kind, respectful, and understanding can make a big difference in teamwork. Kindness keeps people motivated and makes it easier to solve problems together. I try to be more patient and positive when working with others, and I think it’s made me a better teammate.
- Our class project—FlexAI, the virtual fitness trainer app—was a great way to practice everything we learned. I helped with planning, assigning tasks, and solving problems as a team. It felt like working on a real software project. This experience taught me how to handle pressure, communicate better, and keep a project moving forward.
- This course helped me understand what it really means to manage software projects in the real world. Before, I mostly thought project management was about sticking to deadlines and getting tasks done. Now, I know it’s also about planning, teamwork, communication, and being flexible when things change.
- One big thing I learned is how important it is to have a clear plan from the start. Managing a software project isn’t just about writing code—it also means working with people, dealing with risks, and making sure everyone is on the same page. I’ve already started using these ideas in my university projects and job.
- The part about risk management was especially helpful. I now know that spotting possible problems early can help avoid big issues later. Planning for risks makes me feel more in control and ready to handle challenges.
- I also now understand the value of working closely with stakeholders—the people who care about the project. Talking to them early, listening to their needs, and keeping them informed helps avoid confusion and makes the project more successful.

- Another useful idea was adaptive planning. Things don't always go as planned, especially in software projects. Now, instead of sticking to the original plan no matter what, I'm more open to changing things based on feedback or new needs. This flexible way of thinking has made me more confident in leading projects.
- Finally, this course taught me the importance of always learning and improving. I now take time to think about what went well and what didn't after each project. That way, I can do better next time.

Peer Collaboration Insights:

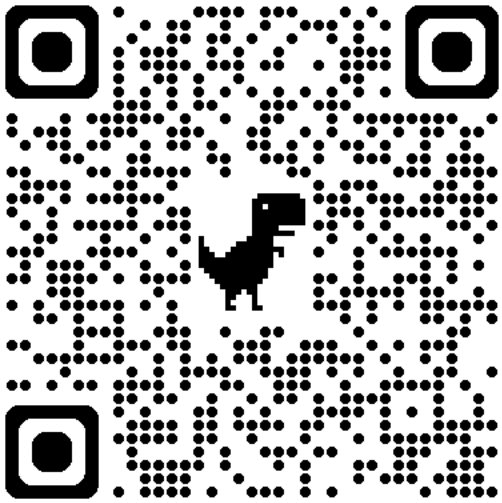
- Our main project, FlexAI – a virtual fitness trainer app – gave us the opportunity to apply software project management concepts in a real-world context, making the learning experience more practical and immersive.
- Working in a group setting allowed us to share ideas openly, divide responsibilities effectively, and support each other's learning throughout the project lifecycle.
- Our group dynamic helped me improve my communication skills, especially in clearly explaining ideas, actively listening, and providing useful feedback.
- Collaboratively solving challenges during the project taught me how essential teamwork is, especially when dealing with deadlines, technical decisions, and evolving requirements.
- Managing time across different tasks and aligning our schedules helped me develop stronger time management and organizational skills.
- Being part of the project from initiation to closure boosted my confidence in applying project management tools and techniques, and in navigating the phases of a real project.
- I learned a lot from observing how my teammates approached problems differently, which expanded my understanding of diverse thinking styles and technical approaches.
- A significant part of our coursework involved a topic analysis on the role of continuous kindness in projects, which really shaped how I view team collaboration.
- This topic highlighted how empathy, support, and kindness create a healthier and more productive work environment, which I witnessed firsthand within our team.
- Understanding the value of kindness helped me become more patient and thoughtful in my interactions, especially during times of stress or disagreement.
- Through continuous feedback loops, we improved both our individual contributions and the overall quality of our project. Feedback was not just about improvement—it was also a form of encouragement.
- There was a strong sense of support in our group. Whenever something was confusing or overwhelming, we helped each other stay calm, focused, and motivated.
- Applying theories from lectures to the FlexAI project made the subject matter more tangible, and I now better understand the practical applications of planning, execution, and team management.

Overall, working on this course project helped me grow both professionally and personally, equipping me with technical knowledge, soft skills, and a deeper appreciation for collaboration and kindness in project teams.

Personal Growth:

- Balancing the course workload with other responsibilities was initially a challenge, but I gradually developed stronger time management strategies tailored to project work.
- Breaking large assignments into smaller tasks, assigning internal deadlines, and using tools like Gantt charts and Trello boards helped me become more efficient and organized.
- The professor truly uniquely combined books, slides, interactive discussions, and real-life examples, which improved my understanding and made the learning process engaging and effective.
- Exploring real-world examples and analyzing case studies helped me understand how these concepts are applied in actual software projects, making abstract ideas feel practical and relevant.
- Throughout the software project management course, I experienced significant personal and academic growth, both in technical knowledge and soft skills.
- Applying the concepts we learned directly to our group project, FlexAI (a virtual fitness trainer app), was a valuable opportunity that bridged theory and practice in a meaningful way.
- The concepts like estimations, risk management, and budgeting seemed quite difficult. However, thanks to lectures, hands-on activities, and discussions with classmates, these topics gradually became much clearer.
- One of the most impactful practices was maintaining a learning journal. Regularly reflecting on what I learned helped me reinforce my understanding, prepare better for exams, and stay engaged with the course content.
- Facing complex project scenarios taught me how to approach problem-solving more strategically and systematically, especially when dealing with uncertainties or shifting requirements.
- Group collaboration played a major role in my growth. Brainstorming, receiving peer feedback, and making joint decisions taught me how to consider multiple viewpoints and work toward shared goals.
- These experiences strengthened my adaptability and teamwork skills the two key components of successful software project management.
- Writing these journals became more than just an academic task; it became a habit of self-reflection and improvement that I plan to carry forward in future projects.
- I've documented all my learning journals on GitHub to track my learning journey throughout the course.

This course has been a transformational experience. It has improved my technical knowledge, enhanced my personal discipline, and helped me become more confident in managing projects and working collaboratively.



https://github.com/MasoumehF/SOEN-6841-Software-Project-Management

Overall, the software project management course has helped me grow a lot—both in terms of what I know and how I work as a person. At the beginning of the course, I found many topics like risk management, cost estimation, scheduling, and stakeholder communication quite difficult to understand. But as the weeks went by, things started to make more sense. The way the professor explained the lessons through lectures, class discussions, and real-life examples really helped me understand how these concepts work in practice. Working on the group project also played a big role in my learning. It was a great chance to take the theories we learned in class and use them in a real project, which made everything more clear and useful. I was also doing an internship at the same time, so I had to balance my time between the course, the project, and my job. This helped me improve my time management and problem-solving skills because I had to plan my days carefully and deal with challenges in a smart way. During the internship, I got to see how project management actually works in real companies. I learned about Agile methods, how teams use two-week sprints, track progress with story points, and keep track of time with timesheets. Seeing these things in action made everything we learned in class feel more real and important. Now, I feel much more confident in working with teams, planning tasks, and handling problems when they come up. I've learned how to communicate better, stay organized, and be more flexible when things don't go as planned. All of these skills will definitely help me in the future, especially if I work in software development or become a project manager. This course has not only given me useful knowledge but has also helped me grow as a more capable and confident person.