Throughout my software engineering course, I gained a deep understanding of systematic design methods that are essential for building scalable and maintainable software. I learned how software development is not just about writing code, but about following structured methodologies such as Agile and Waterfall, each suited to different project needs. These design processes helped me approach problem-solving more efficiently and highlighted the importance of user requirements, planning, and iteration. By working through real-world scenarios and applying concepts like modular design, cohesion, and coupling, I developed a more disciplined and thoughtful approach to creating software systems.

A major focus of the course was also on the variety of diagrams used in software engineering to visualize and document systems. I explored use case diagrams, class diagrams, sequence diagrams, and activity diagrams—each offering a unique perspective on how software components interact and evolve. These visual tools allowed me to better communicate ideas with team members and stakeholders, and helped clarify the behavior and architecture of the systems I designed. Beyond diagrams, the course emphasized testing, version control, documentation, and the ethical responsibilities of a software engineer, rounding out my understanding of what it truly means to develop high-quality software in a professional setting.