Task 2
Group X

Two topics that have helped for software development:

1. <u>Traditional software engineering phases</u> (slide 06, L02_lifecycle_models)

Throughout this course, we were given five different tasks related to traditional software engineering phases: Requirements Engineering, Design, Implementation, Verification and Validation, and Maintenance.

Task 1: Project Description - We applied the theories discussed in lectures to define the requirements for a specific software project.

Task 2: Initial Implementation - We wrote the initial program code. This task was quite challenging because the previous group's requirements were unclear and incomplete.

Task 3: Evolution - We improved the code and added new features, which was part of the verification and validation phase.

Task 4: Testing - We wrote test cases for the software designed by another group. This was also part of verification and validation.

Task 5: Bug Fixing and Reflection - We are fixing bugs here which is part of the maintenance phase.

2. <u>Black-box and white-box testing</u> (slide 21, L07_Testing)

As mentioned, Task 4 involved testing. The worksheet for this task instructed us to perform both black box and white box testing. This slide helps us understand the differences between these two testing types and what each involves.

Two aspects of software development process that might be missing or can be include during lecture:

1. <u>Design Phase</u>

In the lecture, we discussed almost all aspects of the software development process, but one part that can be added is the design phase. In Lecture 3, we learned how to define requirements, which are simply what the system should do. However, I think we didn't discuss how to achieve those requirements, which is part of the design phase. This might be due to time limitations.

2. <u>Coding</u>

Since I was new to Java programming, it was challenging for me at the beginning to implement requirements. I believe discussing Java in our exercise classes would have been advantageous, especially for those of us who did not come from a strong programming background.