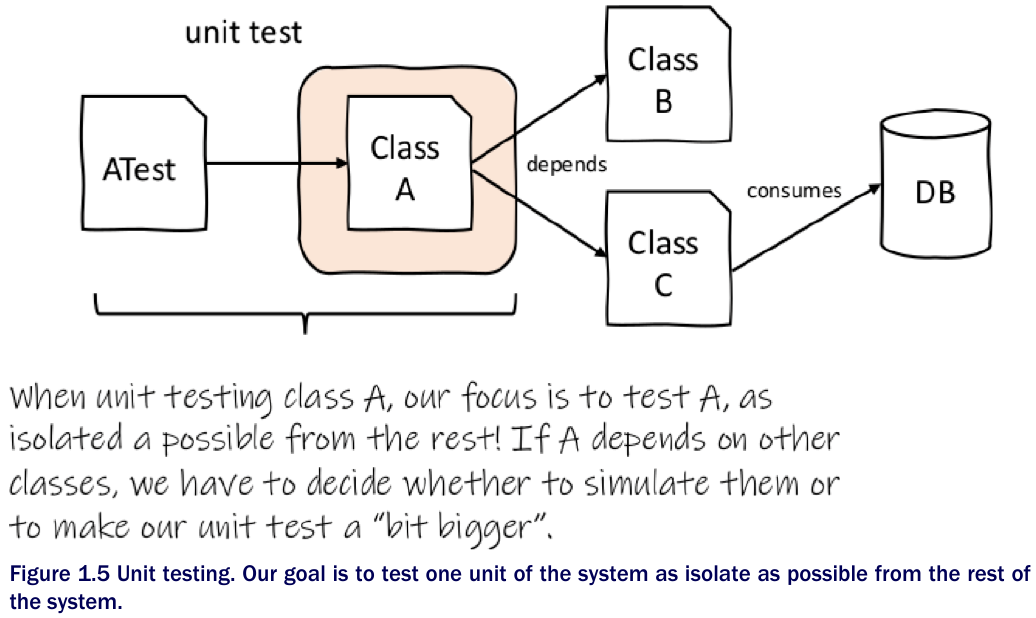
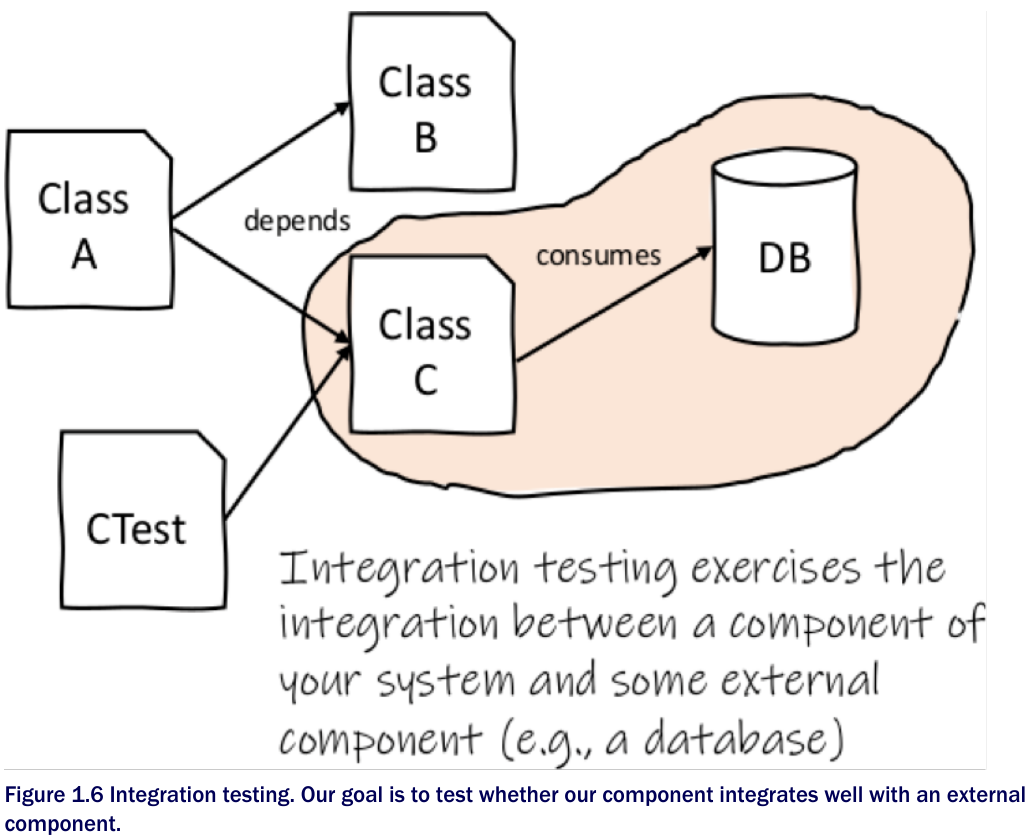
# SOEN390 - Testing Plan

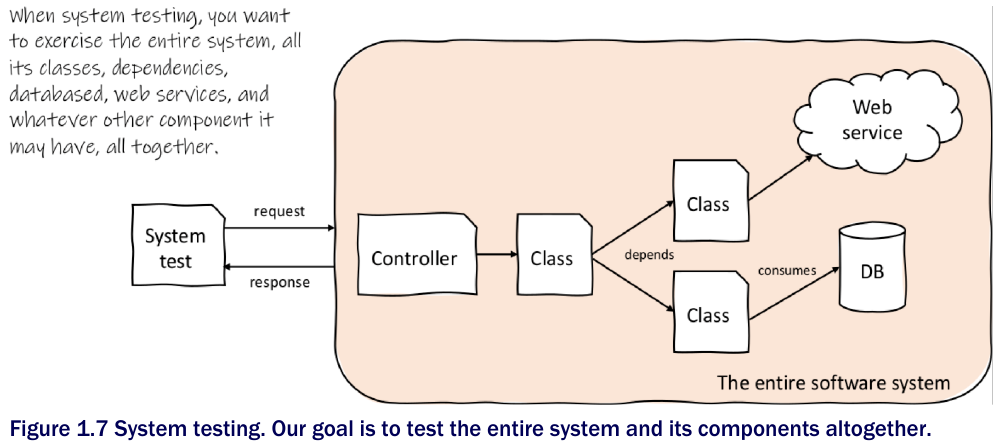
Using the languages Next.js, tailwindCSS and such we will construct a variety of tests, each with its own purpose. Further, an overview of the different types of approach and tools used for testing our application will be discussed below.

We will write three different types of tests which are unit testing, integration testing and finally system testing using Vitest, a testing tool in the Next.js library.

For unit testing, we will break our codes down into units like classes, methods or functions to conduct important tests. Then, we will write test cases to test its behaviour and make sure that under different circumstances they will pass. We will aim for high code coverage while still focusing on the critical functionalities. For unit testing, we will try to aim for 85% coverage. 

For integration testing, we will mainly be focusing on different interactions implemented in our system. These specific interactions to be tested are between different components and modules within AnaCondo. These tests are crucial because they will determine if different components can work together as needed and serve to be useful to the users. For integration testing, we will try to aim for 90% coverage.

For system testing, we will focus on the system as a whole, in other words making sure that AnaCondo works as per the requirements and serves the user's needs. These test cases will verify the behaviour and performance of the full system and simulate it as it will be in the real world. We will need the use cases to ensure that we have completed the tasks as required. We will also need to test the coverage on the usability, performance, security and other crucial components. In other words, for system testing we will try to aim for 100% coverage.



To add, we will also add a form of automation to execute these tests. We will be using a continuous integration pipeline. We will integrate unit tests into our pipeline to make sure that our tests will run automatically on our code with any change in the repository. By using this CI pipeline, we will be able to detect bugs quicker and provide higher quality software. As for the tool, we will be using Jenkins.

As for the acceptance testing, they will be performed by the stakeholder, and some end users as well. This type of testing will make sure the requirements in our developed application meets the needs of the clients. They will utilise the final version of our application and give us any type of feedback.