“Testing”, in the context of the Software Development Lifecycle, is the stage in which a product is deemed viable or not viable according to the requirements put forth on it during the planning stage. Much happens during this stage, such as review of the requirements and specifications, test design, analysis, and reporting. The term “testing” is also a fairly general term as this can mean several different types of testing such as unit testing, integration testing, or regression testing. The amount of effort that goes into testing a product or feature can be large, indeed.

The practice of testing is just as important as the development itself (maybe even more so) and should be treated as such. To not test the product is to potentially release a system that does not work to the use case of your users and thus defeats the purpose of designing something at all. Even worse, to test incompletely can cause major problems and cost major amounts of money. We rely on software to automate major parts of our lives, and in some cases, our lives even depend on it. Consider the software used in an airplane. If that were malfunctioning it could cost the lives of the passengers.

Testing can be done at various times depending on the development model you are using or the product that is being developed. Consider again the example of the airplane software. Since the testing for something such as this is very rigorous, testing during the normal iteration step, in say an agile workflow, may not make much business sense since mistakes need to be found as early as possible to avoid excess costs. For most other cases it would make the most business sense to do the opposite as it might be more costly to continuously ask for changes than it would be to find all mistakes at one time and give them to the developer to correct.