# Test Description

Testing was focused on happy path tests for the behavior (requirements) described in the case description. Normally when more time is given, I would also implement better error handling and test erroneous input.

## Testing The Service Layer

I was not sure if I could use third party mocking libraries (such as “Moq” for the xUnit library) to mock the repository interfaces. So instead, I simply provided real implementations of the repositories when testing the service layer. Usually these would contain more complicated dependencies and I would mock the interfaces to control the responses from them.

I added one happy path test for each service method as a form of sanity check that I implemented all operations specified in the case description.

## Testing The Repository Layer

Usually, the data storage is implemented by a third-party solution in which case the actual implementation shall already be unit tested. I added some simple tests for each repository method to verify my data storage implementation.

## Test Scenarios

I added some additional test scenarios like the one provided in the case description. I focused on testing the behavioral cases I could elicit from the case description, like state transition and issue filtering.