# Test Description

Testing was focused on happy path tests. Normally when more time is given, I would also test more erroneous input.

## Testing The Service Layer (IssueService & UserService)

I was not sure if I could use third party Mocking libraries such as “Moq” for xUnit. So instead of mocking the data layer repositories based on their interfaces, I provided real implementations of them when testing the service layer. This way I saved time and verified the functionality of the data layer at the same time.

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

To seed the “database” I used the service layer calls, normally I would Mock or Fake this layer to seed data necessary for each test. I skipped that due to time restraint.

## Testing The Data Layer (Repositories)

The data layer was indirectly tested through testing of the service layer. I did not unit test the data layer separately since I considered it out of scope and too time consuming. Normally this layer is implemented by a third-party solution in which case the actual implementation shall already be tested.

## 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 “an issue is allowed to transition to any issue state”.