Integration Tests

So, in this section, you learned that:

- Unit tests are easy to write, fast to execute and are ideal for testing functions with minimal or zero dependency on external resources.

- The more you use mock functions, the more your tests get coupled to the current implementation. If you change this implementation in the future, your tests will break. If you find yourself doing too much mocking, that's when you need to replace your unit test with an integration test.

- With integration tests, we test our application with a real database. As a best practice, separate your test database from the development or production databases.

- You should write each integration test as if it is the only test in the world. Start with a clean state (database). Populate the database only with the data required by the test. Nothing more, nothing less. Clean up after your test using the **afterEach** function.

- Run jest with **—coverage** flag to get a code coverage report.