DAO Testing

Integration Testing

- Integration testing verifies the connection between two different systems.

- In our case, the two systems are your Java application and the Postgres database.

Dummy Data

- We want our tests to have a minimum footprint on the database itself, as the existing tables might be in use by other users or applications.

- We therefore generate a temporary database with temporary tables and data.

What are we Testing?

- The tests will be focused on whether your DAO methods are doing what they should

- As such, our tests will probably need:
    - An instance of the DAO
    - Some test data (more on this later)

But What are we REALLY Testing?

- The goal of testing is to verify that the **application works as intended, not as its coded**.
  - We can always write passing tests by inspecting the code that is presented so that our tests pass, but doing so doesn’t make any sense.

- Normally, documentation is provided to describe what an application feature should do: **design specifications**, **Kanban cards**, pseudo-code provided by an architect, etc.

  - In the absence of such documentation, the method header of the test itself can serve as a description, i.e. get_employee_by_state_returns_correct_list_of_employee

Rollback

We use the @After annotation to define a method which rolls back any changes done by a specific test:

```
@After
public void rollback() throws SQLException {
   dataSource.getConnection().rollback();
}
```