Verificação e Validação - TDD + Mockito

Aluno: Saulo Bruno de Freitas Lino

Biblioteka - Sistema de Gestão de BIblioteca

Fiz um sistema web de gestão de biblioteca, que possui algumas entidades:

Autor,
Coleção,
Livro,
Leitor (que possui Usuário),
Empréstimo,
Reserva.

Utilizei o framework do Bean Validation para validar os valores passados pelo usuário do sistema para as classes da camada de modelo. Quanto aos testes dessas classes, testei se a validação estava funcionando corretamente para diferentes cenários (positivos e negativos), especialmente para as validações criadas por mim mesmo através da anotação @Pattern. Além disso, apesar de essas classes não terem muitos métodos diferentes de getters e setters, e desses métodos serem relativamente simples (uma vez que a camada de serviços ficou encarregada pela maior parte da lógica), testei os distintos cenários para os métodos diferentes de getters e setters das classes da camada de modelo.

Por outro lado, as classes da camada de serviço ficaram responsáveis pela maior parte da lógica, de forma que testei vários cenários positivos e negativos relacionados a lógica do sistema em cada uma dessas classes. Também realizei validações que necessitavam do repositório, como saber se um email ou cpf já existe no banco de dados.

Uma vez que as classes da camada de serviço utilizavam repositórios, mockei os repositórios de forma a não precisar acessar o banco de dados ao realizar os testes, o que pode ser bastante custoso, e tendo em vista que o objetivo era saber se as classes da camada de serviço se comportavam adequadamente para diferentes cenários relacionados a lógica do sistema, podendo-se utilizar objetos criados em memória para isso. Também mockei outras classes da camada de serviço ao testar uma classe específica, já que essas outras classes também utilizavam repositórios e sua lógica foi testada de forma individual na sua respectiva classe de teste.

Ao realizar integração, também não mockei outras classes da camada de modelo e as bibliotecas padrão do java, pois não era custoso utilizá-las, tornando mais fácil realizar os testes usando-as diretamente.

Os testes podem ser conferidos nos diretórios de “model” e “services” dentro do diretório de “test”.

Não realizei testes de endpoint nos controladores, já que o objetivo principal da disciplina é testar a lógica do sistema. Também não realizei testes de integração com o PostgreSQL pelo mesmo motivo.