Übungsgruppe 3 , Matrikelnummern: 3720921, 3737820, 3721399, 372687, 3730924
Emily Beck, Elia Soller, Moritz Mairle, Quentin Hadar, Nora Jasharaj

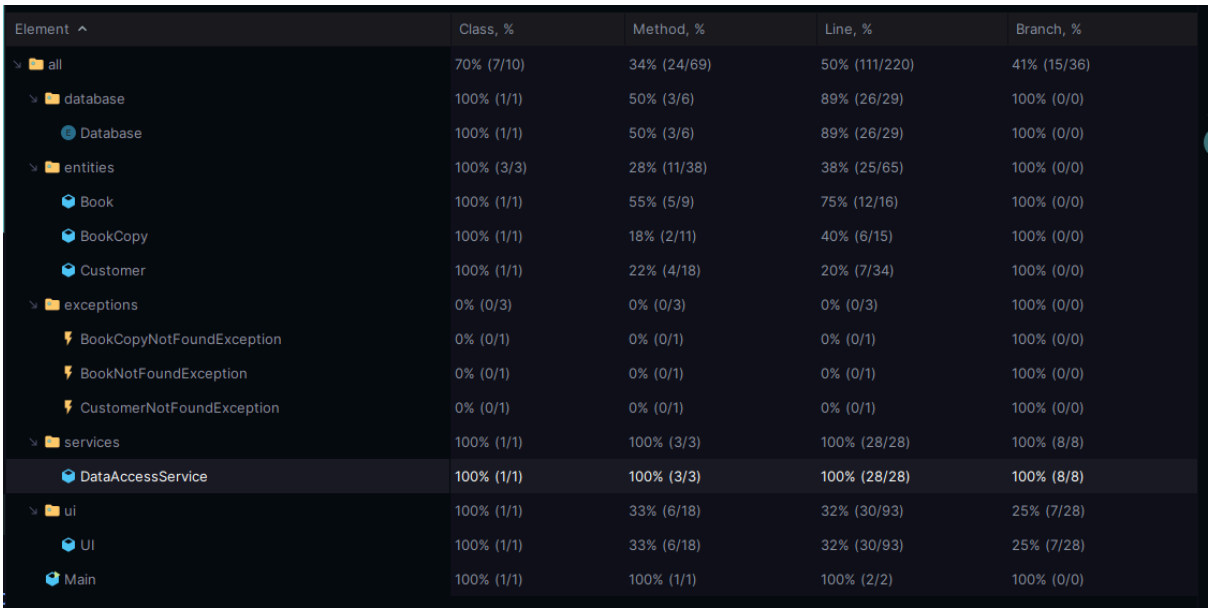
Reporting:

Testing- und Coverage-Infrastruktur
Testing-Framework: *JUnit 5*
In diesem Projekt verwenden wir JUnit 5 für das Unit-Testing unserer DataAccessService-Klasse.

Testing-Bibliotheken:
JUnit Jupiter API: Diese API bietet die Kernfunktionalität für das Schreiben von Tests in JUnit 5.
JUnit Jupiter Assertions: Dieses Modul enthält eine Vielzahl von Assertions, die für das Testen von Ergebnissen verwendet werden können.
Coverage-Tool: JaCoCo

Konfiguration:
JaCoCo Maven Plugin: JaCoCo ist Teil unseres Maven-Projektes, um die Testabdeckung während des Build-Vorgangs zu messen.
JUnit 5: Unsere Testklassen sind mit JUnit 5 geschrieben und verwenden die entsprechenden Annotationen für Testfälle und Setup-Methoden.

Test Abdeckungsanalyse:
Nachdem die Tests ausgeführt wurden, generieren wir einen JaCoCo-Report, um die Testabdeckung zu analysieren. Der Report zeigt, welche Teile des Codes durch Tests abgedeckt sind und welche noch nicht.



Element	Class, %	Method, %	Line, %	Branch, %
all	70% (7/10)	34% (24/69)	50% (111/220)	41% (15/36)
database	100% (1/1)	50% (3/6)	89% (26/29)	100% (0/0)
Database	100% (1/1)	50% (3/6)	89% (26/29)	100% (0/0)
entities	100% (3/3)	28% (11/38)	38% (25/65)	100% (0/0)
Book	100% (1/1)	55% (5/9)	75% (12/16)	100% (0/0)
BookCopy	100% (1/1)	18% (2/11)	40% (6/15)	100% (0/0)
Customer	100% (1/1)	22% (4/18)	20% (7/34)	100% (0/0)
exceptions	0% (0/3)	0% (0/3)	0% (0/3)	100% (0/0)
BookCopyNotFoundException	0% (0/1)	0% (0/1)	0% (0/1)	100% (0/0)
BookNotFoundException	0% (0/1)	0% (0/1)	0% (0/1)	100% (0/0)
CustomerNotFoundException	0% (0/1)	0% (0/1)	0% (0/1)	100% (0/0)
services	100% (1/1)	100% (3/3)	100% (28/28)	100% (8/8)
DataAccessService	100% (1/1)	100% (3/3)	100% (28/28)	100% (8/8)
ui	100% (1/1)	33% (6/18)	32% (30/93)	25% (7/28)
UI	100% (1/1)	33% (6/18)	32% (30/93)	25% (7/28)
Main	100% (1/1)	100% (1/1)	100% (2/2)	100% (0/0)

Erklärung:
Wie man erkennen kann, liegt die Testabdeckung (Coverage) für unseren DataAccessService bei 100%. Dies ist auf die DataAccessServiceTest Klasse zurückzuführen. Der DataAccessService ist bisher das Herzstück des Projekts, welches momentan für das Löschen von Büchern, Buchkopien und Kunden verantwortlich ist.
Die anderen Klassen werden teilweise auch durch den DataAccesServiceTest abgedeckt. Wir haben jedoch keine eigenen Tests für jede Klasse geschrieben, da es keinen Sinn macht für Entities, Database, den Exceptions, der UI, so wie Getter und Setter Tests zu schreiben.