Ergebnisse der Implementierungen

Ergebnisse:

Implementierung:	Korrekt:	Verstöße:
1	Ja	Keine
2	Nein	R4, R7
3	Nein	R2, R7
4	Ja	Keine

Sicherstellung der Korrektheit:

Um sicher zu stellen dass alle Requirements erfüllt sind, wurden in den Command-files Testfunktionen implementiert. Diese Funktionen verwenden einen Boolean-Datentyp als Return-Wert, der anzeigt ob ein Requirement verletzt wurde. Liefern sie True zurück ist kein Requirement verletzt - liefern sie hingegen False zurück wird ein Teststring mit dem verletzten Requirement an die ToString()-Methode des jeweiligen Commands angehängt.

In den 3 Commands, Publish, FindMessages und LikeMessage werden jeweils folgende Requirements überprüft:

- PublishCommand: R1, R2, R6, und R7
- FindMessagesCommand: R1, R2, R4 und R5
- LikeMessage: R3, R6 und R7

Die TestMethod1() Testklasse ruft jeweils, wie in den Tutorial-files gezeigt, eine neue Instanz der MessageBoardSpecification() auf und wiederholt alles so lange, bis entweder ein Gegenbeispiel von FsCheck gefunden wurde - oder aber die maximale Anzahl an Testläufen erreicht wurde.

Als Anzahl der maximalen Testläufen wurde aufgrund der Laufzeit 2.000 gewählt.

MessageBoardSpecification() erstellt pro Durchlauf 200 zufällige Test-Strings, welche aus sämtlichen Varianten der Groß- und Kleinbuchstaben bestehen können. Die Länge wird ebenso zufällig zwischen 1 und 20 gewählt, um sicherstellen zu können, dass sowohl erlaubte messages (Länge <= 10) als auch verbotene (Länge > 10) getestet werden.

Zusätzlich zu den zufällig generierten Testdaten verwendet die Spezifikation drei festgelegte Strings, um die Überprüfung auf doppelte Publish bzw. Like Commands zu garantieren.

Implementierung 1:

Implementierung 1 wurde als korrekt erkannt. Siehe oben. (Sicherstellung der Korrektheit)

Implementierung 2:

```
[Publish(Richard,Hello) [R2] Violated: Message already exists!;
 Like(Richard,Hello, liked by:Maria) ;
 Like(Richard,Hello, liked by:Maria) [R7] Violated: Model was unsuccessfull but no OperationFailed was
received!]
```

Bei Implementierung 2 akzeptiert das System offensichtlich, dass Messages von derselben Person mehrmals geliked werden können, das Model scheitert hier jedoch. Hierbei werden R4 (mehrfache likes) sowie R7 (Model fail - Actual ok) verletzt.

Implementierung 3:

```
[Publish(Richard,Hello) [R2] Violated: Message already exists!;
 Publish(Richard,Hello) [R7] Violated: Model was unsuccessfull but no OperationFailed was received!]
```

Bei Implementierung 3 akzeptiert das System das Speichern doppelter Nachrichten. Dadurch wird R2 (gleiche Nachrichten vom selben Autor) verletzt. Das Model scheitert wiederum und erwartet eine OperationFailed Nachricht, das System hingegen sendet eine OperationAck Nachricht.

Implementierung 4:

Implementierung 4 wurde ebenfalls als korrekt erkannt. Siehe oben. (Sicherstellung der Korrektheit)