IT kompakt

Pascal Moll
Daniel Sonnet

Softwaretesting kompakt

Grundlagen von Tests und Testautomatisierung mit Java

Springer Vieweg

IT kompakt

Die Bücher der Reihe „IT kompakt“ zu wichtigen Konzepten und Technologien der IT:

- ermöglichen einen raschen Einstieg,
- bieten einen fundierten Überblick,
- eignen sich für Selbststudium und Lehre,
- sind praxisorientiert, aktuell und immer ihren Preis wert.

Pascal Moll · Daniel Sonnet

Softwaretesting kompakt

Grundlagen von Tests und Testautomatisierung mit Java

Springer Vieweg

Pascal Moll
Karben, Deutschland

Daniel Sonnet
Hochschule Fresenius Hamburg
Hamburg, Deutschland

ISSN 2195-3651 ISSN 2195-366X (electronic)
IT kompakt
ISBN 978-3-658-46104-1 ISBN 978-3-658-46105-8 (eBook)
https://doi.org/10.1007/978-3-658-46105-8

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://portal.dnb.de abrufbar.

Planung/Lektorat: David Imgrund
Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.
Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Danksagung

Mein besonderer Dank gilt **Alex Schuler** fürs sorgfältige Korrekturlesen, Fehlerfinden und den entscheidenden Anstoß, dieses Werk zu verfassen. Ohne dich hätte ich dieses Buch vermutlich noch immer nicht geschrieben.

Ebenso möchte ich mich bei **Moritz Wittig** fürs ausführliche Korrekturlesen und Fehlerfinden bedanken. Deine wertvollen Vorschläge haben dieses Buch besser gemacht und du hast mit den gefundenen Bugs in Screenshots und Texten die Qualität gesteigert.

Herzliches Dankeschön auch an **Patrick Eschenbach** und **Stefan Ludwig,** die ständig zum Korrekturlesen und Fehlerfinden für mich da waren.

Besonders möchte ich meiner Frau **Julia Beck** und meinen Kindern **Zoe Moll** und **Jan Moll** danken. Ohne eure Unterstützung wäre es mir nicht möglich gewesen, dieses Werk zu vollenden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Grundlagen des Softwaretestings

1

„Etwas zu testen bedeutet, eine Annahme kritisch auf Wahrheit zu prüfen“

Eigene Definition von Testing

Inhaltsverzeichnis

Einleitung

Vor mehr als einem Jahrzehnt startete ich mein erstes Projekt als Softwaretester. Zu diesem Zeitpunkt hatte ich nahezu keine Erfahrungen in diesem Bereich. Klar war das Thema im Studium prä-

P. Moll und D. Sonnet, *Softwaretesting kompakt,* IT kompakt,
https://doi.org/10.1007/978-3-658-46105-8_1

sent, aber die verschiedenen Testarten oder die Erstellung einer Teststrategie waren kein Bestandteil der Vorlesung. Überhaupt fehlte mir der erste Einstieg in die Thematik. Ich begann daher bei null und musste mir sämtliches Wissen zunächst aneignen.

Für das Projekt eines großen Kunden war es notwendig, auf Oberflächentests zurückzugreifen; dies war daher auch mein Fokus. Würde ich allerdings noch mal beginnen, wäre der erste Schritt, die Grundlagen kennenzulernen. Diese Grundlagen sind wichtig, um Fehler zu vermeiden, die richtige Testart für die richtige Problemstellung zu nutzen und ein gutes Fundament für die anstehenden Inhalte zu schaffen. Dieser Einstieg hilft auch dabei, komplexe Sachverhalte leicht zu verstehen, sodass aufbauendes Wissen schneller und leichter möglich ist.

In diesem Buch werde ich daher zuerst besagte Grundlagen des Softwaretestens erläutern, gängige Testarten vorstellen und die Programmiersprache Java aus Testsicht vermitteln. Es sind keine Vorkenntnisse notwendig, um den Erklärungen zu folgen. Die weiteren Kapitel umfassen dann Testautomatisierung und Testframeworks, Mocking und Oberflächentests. Den Abschluss bilden Behaviour Driven Development und exploratives Testen.

Ausgangslage – Das Projekt

Ein neues System einführen ohne zu testen? Was soll schon schiefgehen…

Vor langer Zeit war ich als Entwickler daran beteiligt, ein neues System für eine Onlinehandelsplattform aufzubauen. Grundlegend sollte ein Onlineshop eingeführt werden, der verschiedenste Schnittstellen zu Ebay, Amazon und diversen Zahlungsdienstleistern bietet. Das ganze Rechnungswesen wurde über SAP abgewickelt. Da der Onlineshop zur damaligen Zeit zwar eingekauft wurde, aber verschiedenste Features nicht mitlieferte, wurden diese von Entwicklern nach implementiert.

Beispielsweise das Gutscheinsystem erhielt zahlreiche neue Funktionen. All das erfolgte ohne Teststrategie und ohne ausreichende Tests.

Dieses Fehlen hatte weitreichende Folgen.

Folgen der fehlenden Tests

Die Angst vor Codeoptimierungen

Während der Entwicklung neuer Features wird vorhandener Programmcode oft verändert, so zum Beispiel bei der erwähnten Gutscheinfunktion. Eine Erweiterung sollte das Eingeben von mehreren Gutscheinen auf einmal ermöglichen. Der vorhandene Code wurde angepasst und dabei entstanden zahlreiche neue Zeilen mit mehr Funktionsinhalten. Nach der Featureentwicklung fanden aber keine Optimierung und keine Anpassungen mehr statt. Der Kunde klickte die Oberflächen durch und nahm die Funktion ab. Danach wurde dieses Feature nicht mehr verändert und der Code blieb in seiner Form bestehen mit allen unnötigen Programmzeilen und so chaotisch wie bisher. Schon eine Woche später fehlte jegliches Verständnis über die vorhandene Programmlogik.

Eigentlich entwickelt Programmcode sich stetig weiter, wird aufgeräumt und verbessert. Dieses sogenannte Refactoring findet immer wieder statt und dabei schleichen sich hin und wieder auch Fehler ein. Bei vorhandenen Tests fallen diese Bugs schnell und zeitnah auf, werden behoben und das Refactoring wird abgeschlossen. Sind keine Tests vorhanden, bleibt immer ein Risiko bestehen und eine damit verbundene Unsicherheit. Niemand möchte für Fehler verantwortlich sein und dadurch werden Codeoptimierungen wenig bis gar nicht mehr durchgeführt.

Fehler frühzeitig erkennen und beheben

Wird bei einer Abfragebedingung nur ein Größerzeichen statt einem Größer-gleich verwendet, liefert die Funktion schon ganz andere Ergebnisse. Solche Probleme bleiben ohne Tests zunächst unerkannt, da sie erst bei der genaueren Ergebnisbetrachtung auffallen. So war es auch beim Shop. Besagter Fehler trat auf und schon war der Minimalwert für die Anwendung des Gutscheins plötzlich falsch.

Für einen Gutschein, der einen Mindesteinlösewert von 10,00 oder mehr Euro hatte, sollte ein Rabatt von 10 % gewährt werden. Hat die einkaufende Person einen Warenkorbwert von 10,01 € geht die Funktion durch, bei 10,00 € erscheint eine Fehlermeldung. Ein kleiner Fehler, der zunächst unerkannt blieb, aber einige Einkäufe abbrechen ließ und somit für Umsatzverluste sorgte. Während des Entwickelns trat einfach kein gezielter Fall auf, bei dem der Gesamtwarenwert exakt 10 € betrug, alle waren weit drüber.

Nachdem die Meldung vom Kunden kam, musste zunächst der Fall rekonstruiert werden. Zunächst war nicht klar, wo genau die Fehlerursache lag. Erst nach längerer Analyse fiel das fehlende Gleichheitszeichen auf.

Menschen machen Fehler, so auch beim Softwareentwickeln. Aber diese Fehler werden erst zum Problem, wenn sie unerkannt bleiben. Tests in verschiedenen Varianten, mit verschiedensten Parametern helfen dabei, diese Defekte auch schon beim Entwickeln früh zu finden. Dadurch wird eine direkte Behebung auch gleich möglich und spart damit Zeit und Geld.

Kundenzufriedenheit und Firmenimage

Was passiert, wenn knapp hundert Personen gleichzeitig einen Shop besuchen und darauf einkaufen? Wie verhält es sich, wenn diese Zahl plötzlich auf tausend steigt?

Durch eine Marketingaktion mit verschiedenen Rabatten auf limitierte Sportschuhe wurden zahlreiche Besucher auf den Shop gelockt. Die Aktion war sehr erfolgreich, aus Serversicht zu erfolgreich. Um 24:00 Uhr nachts wurden die Gutscheine freigeschaltet und somit begann auch der Ansturm auf den Shop.

Die gleichzeitige Besucherzahl war für das Shopsystem ein großes Problem. Im Hintergrund fungierte eine MySQL-Datenbank als Datenhaltungsschicht, während verschiedene Schnittstellen mit SAP und weiteren Systemen kommunizierten. Irgendwann war der Webserver überlastet. Die Performance des Systems ging immer weiter zurück, bis schließlich der ganze Server zusammenbrach. Welche Komponente der Auslöser war, blieb

zunächst unerkannt und benötigte einige Zeit der Nachforschung. Nach zahlreichen Telefonaten mit dem Serverprovider und diverser Ermittlungsarbeit konnte die Datenbank identifiziert werden. Zwar wurden notwendige Einstellungen rasch umgesetzt, dennoch deutlich zu spät für die teure Marketingkampagne.

Entsprechende Kritik hagelte es auch von Shopbesuchern, die ihre Bestellungen nicht abschließen konnten. Kunden behalten solche Situationen im Gedächtnis und verbreiten sie weiter. Dies kann schnell zu Reputationsschäden führen. Unbekannt bleibt, wie viel potenzielle Kunden dadurch zur Konkurrenz wechselten.

Mit meinem heutigen Wissen und der Perspektive eines Testers würde ich dieses Projekt komplett anders angehen.

Die Aufgaben eines Testers

Eine der wichtigsten Aufgaben eines Softwartesters, einer Softwaretesterin ist, die Qualität einer Software dauerhaft und nachhaltig zu verbessern. Dies gelingt durch die Entwicklung und Umsetzung einer Softwareteststrategie. Dabei werden Testkonzepte, Vorgehen und Methoden festgelegt. Meistens ist hierfür ein Testmanager verantwortlich. Außerdem gehören die Auswahl und Entwicklung von Testtools sowie Testframeworks in den Aufgabenbereich von Testern. Gerade Eigenentwicklungen sollten dabei wie ein Softwareprodukt behandelt werden, einer Architektur folgen und grundlegende Designvorgaben erfüllen. Häufig fällt dieser Aufgabenbereich in die Rolle des Testautomatisierers. Wichtig ist bei der Erstellung von Frameworks und Tools, dass dabei Flexibilität eingehalten und Komplexität minimiert wird. Mit genau diesen Themen beschäftigt sich das Kap. 4.

Zudem existiert die Rolle des Testanalysten. Dabei stehen Analyse, Planung und Erstellung von Tests im Vordergrund. Vorgaben werden analysiert und Testfälle mit unterschiedlichen Testmöglichkeiten sowie Testtypen hergeleitet. Diese können manuell oder automatisiert durchgeführt werden. Um eine Auswahl über die richtigen Testarten für die richtige Situation treffen zu können, werden diese grundlegend vorgestellt.