Das Grosse SQL-Spiel

The SQL-Alchemist

Software-Entwicklungspraktikum (SEP)
Sommersemester 2015

Testspezifikation

Auftraggeber:
Technische Universität Braunschweig
Institut für Informationssysteme
Prof. Dr. Wolf-Tilo Balke
Mühlenpfordtstraße 23, 2.OG
D-38106 Braunschweig

Betreuer: Jan-Christoph Kalo

Auftragnehmer:

Name	E-Mail-Adresse
Gabriel Ahlers	g.ahlers@tu-braunschweig.de
Majid Dashtiepielehroud	m.dashtiepielehroud@tu-braunschweig.de
Ronja Friebe	r.friebe@tu-braunschweig.de
Stefan Hanisch	stefan.hanisch@tu-braunschweig.de
Fabio Luigi Mazzone	f.mazzone@tu-braunschweig.de
Nicole Naczk	n.naczk@tu-braunschweig.de
Denis Nagel	denis.nagel@tu-braunschweig.de
Luca Porcello	l.porcello@tu-braunschweig.de
Christian Reineke	c.reineke@tu-braunschweig.de
Christian Sander	christian.sander@tu-braunschweig.de
Carl Schiller	c.schiller@tu-braunschweig.de
Levent Muzaffer Üner	l.uener@tu-braunschweig.de
Sören van der Wall	s.van-der-wall@tu-braunschweig.de
Daniel Wolfram	d.wolfram@tu-braunschweig.de

Braunschweig, 15. Juli 2015

Inhaltsverzeichnis

1 Einleitung

Einer der wichtigsten Aspekte bei der Entwicklung von Software ist das ausführliche Testen des Programms. Dabei müssen für diese Software alle Softwarebestandteile, auf ihre Funktionsweise, Ausfallsicherheit und Bedienbarkeit getestet werden. Gleichzeitig dient das Testen der Qualitätssicherung der Applikation. Dies ist speziell bei dieser Software wichtig, da sie den Lehrbetrieb der Veranstaltung „Relationale Datenbanksysteme 1 “ unterstützen und einen Teil des Hausaufgabenablaufs übernehmen soll.

Aus diesem Grund wird auch der SQL-Alchemist umfangreichen Tests unterzogen um einen reibungslosen Betrieb, während der späteren Nutzung zu gewährleisten. Insbesondere da das Programm später vorlesungsunterstützend eingesetzt werden soll und Einfluss auf die erfolgreiche Bearbeitung der Hausaufgaben der Studenten hat, ist es umso wichtiger, dass keine unvorhergesehenen Fehler auftreten, welche durch ein frühes Testen hätten verhindert werden können.

In diesem Dokument wird ausführlich auf die einzelnen Bereiche der Software eingegangen, welche verschiedenen Tests unterzogen werden müssen. Dabei wird auch die genaue Vorgehensweise der Testläufe näher beschrieben.

2 Testplan

Im Folgenden wird der Testplan für SQL-Alchemist beschrieben. Dabei wird sowohl auf die zu testenden Komponenten als auch die bereits im Pflichtenheft beschriebenen Funktionen und Merkmale eingegangen. Außerdem wird noch das genaue Vorgehen bei der Durchführung der Tests aufgezeigt.

2.1 Zu testende Komponenten

Die Sofware besteht grundsätzlich aus zwei Komponenten: Front-End und Back-End.
Da diese jedoch auf verschiedene Weise an verschiedenen Teilen der Software beteiligt sind, sind folgend die verschiedenen Kernpunkte aufgelistet, die zur Bewertung der Funktionstüchtigkeit getestet werden müssen:

- **SQL Modul:** Das SQL Modul ist der Kern der Software. Der Zweck des Programms, nämlich der Unterstützung beim Erlernen der Datenbankanfragesprache SQL, wird hierüber realisiert. Daher muss besonders die korrekte Anzeige der Aufgaben und die automatische Bewertung getestet werden. Weiterhin soll eine simple Bedienbarkeit gewährleistet werden.
- **Minispiel:** Das Minispiel stellt die zweite zentrale Komponente des Projektes dar und muss daher auch ausgiebig getestet werden. In diesem Fall liegt das Hauptaugenmerk auf der Spielbarkeit der einzelnen Level und die Einhaltung des Schwierigkeitsgrades.
- **GUI:** Die grafische Oberfläche bildet die Schnittstelle zwischen Benutzer und Applikation. Daher muss hier geprüft werden, ob die Menüs auf allen Endgeräten korrekt angezeigt werden und intuitiv bedienbar sind.
- **Datenbank:** Die Datenbank ist für die Speicherung aller Daten notwendig, die für die Erstellung der SQL-Abfragen benötigt werden. Demnach muss die Konsistenz und die Sicherung der Daten geprüft werden um einen einfachen Zugriff sicherzustellen.
- **Benutzerverwaltung:** Die Benutzerverwaltung sorgt für die Speicherung der Spieleraccounts. In diesem Fall muss insbesondere die korrekte Rechtevergabe getestet werden.
- **Ranglisten/Spielerprofile:** Die Profile halten den Fortschritt der Spieler fest, während die Ranglisten es ermöglichen, dass sich die Nutzer untereinander messen können. Daher

ist es wichtig, dass alle zu erfassenden Daten in gewünschter Weise abgespeichert und angezeigt werden.

- **Administrationstool:** Das Administrationstool ist das Werkzeug, welches benötigt wird damit die Dozenten und ihre Mitarbeiter das Spiel vorlesungsunterstützend einsetzen können. Die gesamte Hausaufgabenverwaltung läuft über dieses Tool. Demnach muss an dieser Stelle getestet werden, ob die Erstellung neuer Eingaben intuitiv und schnell machbar ist.

2.2 Zu testende Funktionen/Merkmale

Um die oben genannten Kernpunkte der Software effizient zu testen bietet sich an, die Produktfunktionen zu Hilfe zu nehmen. Zu diesem Zweck sind diese hier noch einmal aufgefürt und kurz umschrieben.

- ⟨**F10**⟩ **Nutzer registrieren:** Neue Benutzer lassen sich durch Angabe von E-Mail/y-Adresse, sowie Passwort registrieren. Dabei wird auf die Vollständigkeit der Angaben geachtet und Studenten werden automatisch anhand der Eingaben erkannt.
- ⟨**F20**⟩ **Nutzer anmelden:** Die Nutzer-Authentifizierung erkennt, ob es sich um einen registrierten Nutzer handelt und gewährt diesem Zugang.
- ⟨**F30**⟩ **Nutzer abmelden:** Der Nutzer wird ordnungsgemäß abgemeldet und kann, ohne erneute Anmeldung, nicht mehr auf das Spiel zugreifen.
- ⟨**F40**⟩ **Profil einsehen:** Das Profil wird auf allen Endgeräten korrekt und gut lesbar dargestellt. Alle angezeigten Daten sind richtig und aktuell.
- ⟨**F60**⟩ **Passwort ändern:** Das Passwort wird geändert, der Passworthash in der Datenbank aktualisiert sich dabei. Anschließend muss sich der Nutzer mit dem neuen Passwort anmelden können, das alte Passwort ist nicht mehr gültig.
- ⟨**F70**⟩ **Avatar ändern:** Die Spielfigur im Minispiel und das Icon das Nutzers werden geändert und korrekt dargestellt.
- ⟨**F90**⟩ **Audioeinstellungen bearbeiten:** Soundeffekte und Musik können unabhängig voneinander ein- und ausgeschaltet werden. Die Einstellung wird korrekt in der Datenbank gespeichert und bei der nächsten Anmeldung korrekt geladen.
- ⟨**F100**⟩ **Spielstand zurücksetzen:** Der Spielstand wird korrekt zurückgesetzt. Das Tutorial bleibt deaktiviert.
- ⟨**F110**⟩ **Tutorial spielen:** Das Tutorial ist standardmäßig bei einem neuen Nutzer aktiviert, bis dieser das Tutorial erfolgreich beendet hat. Dann wird das Tutorial automatisch deaktiviert, bis es im „Settings“-Menü erneut aktiviert wird.

- ⟨**F120**⟩ **Story spielen:** Der Ablauf der Handlung ist korrekt und das Zusammenspiel von Minispiel und SQL-Modul funktioniert wie geplant.
- ⟨**F130**⟩ **SQL-Trainer spielen:** Die Aufgaben wiederholen sich im Trivia-Modus nicht zu häufig. Im Story-Modus werden zu jedem Zeitpunkt die aktuellen Aufgaben gestellt. Zudem darf es nicht zu Fehlern kommen, die die Bearbeitung einschränken. Die Rückmeldung auf die Antwort der Nutzer ist korrekt.
- ⟨**F140**⟩ **Minispiel spielen:** Durch den zufälligen Aufruf der Level kommt es nicht zu Wiederholungen. Die Figur reagiert sofort auf Eingaben und das Spiel ist auf allen Endgeräten spielbar.
- ⟨**F150**⟩ **Hausaufgaben bearbeiten:** Es werden die geforderten Aufgabenpakete geladen und die Bewertung funktioniert wie erwünscht. Bei der Bearbeitung darf es nicht zu Fehlern kommen, die den Betrieb einschränken.
- ⟨**F160**⟩ **Ranglisten einsehen:** Die Ranglisten stellen alle geforderten Daten aktuell und richtig dar.
- ⟨**F170**⟩ **Spieler suchen:** Spieler sollen anhand ihres Benutzernamens gesucht und korrekt angezeigt werden.
- ⟨**F180**⟩ **Hausaufgabenergebnisse einsehen:** Die in den Hausaufgaben erzielten Ergebnisse werden ordnungsgemäß abgespeichert und für jeden Nutzer sind nur die eigenen Ergebnisse einsehbar.
- ⟨**F200**⟩ **Benutzer Adminrechte geben:** Die Option kann nur von einem Admin genutzt werden und ernennt nur den richtigen Benutzer zum Admin.
- ⟨**F210**⟩ **Trivia-Aufgabe erstellen:** Nur berechtigte Nutzer haben Zugriff auf diese Funktion und die Aufgaben werden korrekt und vollständig in die Datenbank eingepflegt.
- ⟨**F220**⟩ **Benutzeraufgaben bewerten:** Die Bewertung kann nur von Admins oder beförderten Nutzern vorgenommen werden und wird korrekt mit den bereits vorhandenen Bewertungen verrechnet.
- ⟨**F230**⟩ **Hausaufgaben erstellen:** Hausaufgaben können nur von Admins erstellt und nur von Studenten bearbeitet werden. Die Hausaufgaben sind nur innerhalb des angegebenen Bearbeitungszeitraums freigeschaltet.
- ⟨**Q10**⟩ **Benutzerfreundlichkeit:** Alle Funktionen sind für den Nutzer verständlich und können über möglichst kurze Wege durch das Menü erreicht werden.
- ⟨**Q20**⟩ **Zuverlässigkeit:** Das Spiel stürzt nicht ab und die Bearbeitung der Hausaufgaben ist technisch jederzeit möglich.
- ⟨**Q30**⟩ **Korrektheit:** Die Auswertung der Benutzereingaben ist korrekt.

- ⟨**Q40**⟩ **Plattformunabhängigkeit:** Das Spiel ist plattformunabhängig nutzbar.

2.3 Nicht zu testende Funktionen

Bei allen verwendeten Frameworks und Funktionen von Dritten werden ausreichende Testläufe vorausgesetzt. Daher werden keine weiteren Tests für die folgende Software vorgenommen:

- MelonJS (Framework)
- play (Framework)
- jQuery (Framework)
- AngularJS (Framework)
- Bootstrap (Framework)
- automatische Generierung von Schemata und Statements (Teamprojekt)

2.4 Vorgehen

a) Abnahme- und Funktionstests:
Für den Abnahmetest ist folgendes Vorgehen geplant: Damit der Kunde einen guten Gesamtüberblick über alle Funktionalitäten erhält, sollen alle Anwendungsfälle aus der Anforderungsspezifikation in einer Reihenfolge getestet werden, in der auch spätere Nutzer die Applikation nutzen könnten. Dazu wird sich der Kunde zuerst einen neuen Account erstellen, dem dann der Administrator-Status zugewiesen wird. Mit diesem Account kann er zunächst, unabhängig vom eigentlichen Programm, die verschiedenen Funktionen des Administrationstools testen. Danach wird er sich für die eigentliche Applikation anmelden um die unterschiedlichen zur Verfügung stehenden Funktionen auszuprobieren und auch die verschiedenen Spielmodi anzuspielen. Zuletzt wird der Kunde dann noch die Abmeldung durchführen. Während dieses Testlaufs kann sich der Kunde dann auch gleich ein Bild davon machen, inwieweit die im Pflichtenheft festgelegten Qualitätsstandards eingehalten wurden.

b) Integrationstest:
Im Integrationstest wird für diese Software die Komunikation der beiden Komponenten getestet. Da die Komponenten über verschiedene Schnittstellen miteinander kommunizieren, werden für den Integrationstest die Funktionen nochmal genauer beleuchtet, die die jeweilige Schnittstelle benutzen. Dabei werden die Funktionen manuell mehrfach ausgeführt um die stabilität der Daten zu gewährleisten. Desweiteren werden die angezeigten Daten dabei mit den Daten in der Datenbank verglichen.

c) Unittest:
Bei den Unit-Tests werden die einzelnen Komponenten für sich selbst überprüft. Dies geschieht für das Back-End schon während der Programmierung. Da das Back-End zum Großteil eine Java-Basierte Anwendung ist wird beim Testen Gebrauch von JUnit gemacht, es wird also für jede erstellte Klasse direkt ein Test erstellt. Da das Front-End eine Javascript-basierte Anwendung ist und sich Unittests dafür nur etwas zeitaufwendiger umsetzen lassen wurde von diesen, in Absprache mit dem Protokoll-abnehmenden Institut, abgesehen.

2.5 Testumgebung

Back-End Funktionen werden mit der JUnit getestet. Diese ist in das benutzte play!-Framework integriert. Tests für Front-End Funktionen werden manuell durchgeführt.

3 Abnahmetest

Der Abnahmetest für den SQL-Alchemist hat den Zweck die vom Auftragnehmer im Pflichtenheft festgelegten Anforderungen an das fertige Produkt auf ihre Vollständigkeit zu überprüfen. Dabei sollen sowohl die vom Auftraggeber geforderten technischen Funktionen als auch die eigentliche Ausführung des Produktes bewertet werden. Das Produkt soll dem Kunden samt aller Funktionen vorgestellt werden um einen Abgleich mit dessen Forderungen zu ermöglichen. Das Ziel ist es, dem Kunden das von ihm geforderte Produkt zu liefern, sodass dieser vollständig zufrieden ist und die Software abnehmen kann.

3.1 Zu testende Anforderungen

In diesem Abschnitt werden alle zu testenden Funktionen aufgeführt werden, die im Rahmen des Abnahmetests vom Kunden ausgeführt werden. Die Reihenfolge der aufgeführten Funktionen spiegelt eine beispielhafte Nutzungsabfolge wider, welche auch ein zukünftiger Nutzer bzw. Admin durchlaufen würde.

Nr	Anforderung	Testfälle	Kommentar
1	⟨**F10**⟩ **Nutzer registrieren**	⟨T300⟩	
2	⟨**F20**⟩ **Nutzer anmelden**	⟨T300⟩	
3	⟨**F110**⟩ **Tutorial spielen**	⟨T500⟩	
4	⟨**F120**⟩ **Story spielen**	⟨T500⟩	
5	⟨**F130**⟩ **SQL-Trainer spielen**	⟨T500⟩, ⟨T600⟩	
6	⟨**F140**⟩ **Minispiel spielen**	⟨T500⟩	
7	⟨**F100**⟩ **Spielstand zurücksetzen**	⟨T500⟩	
8	⟨**F90**⟩ **Audioeinstellungen bearbeiten**	⟨T400⟩	
9	⟨**F230**⟩ **Hausaufgaben erstellen**	⟨T100⟩	
10	⟨**F150**⟩ **Hausaufgaben bearbeiten**	⟨T600⟩	
11	⟨**F180**⟩ **Hausaufgabenergebnisse einsehen**	⟨T200⟩	
12	⟨**F40**⟩ **Profil einsehen**	⟨T400⟩	

13	⟨**F60**⟩ **Passwort ändern**	⟨T400⟩	
14	⟨**F70**⟩ **Avatar ändern**	⟨T400⟩	
15	⟨**F160**⟩ **Rangliste einsehen**	⟨T400⟩	
16	⟨**F190**⟩ **Benutzer befördern**	⟨T100⟩	
17	⟨**F200**⟩ **Benutzer Adminrechte geben**	⟨T100⟩	
18	⟨**F210**⟩ **Trivia-Aufgabe erstellen**	⟨T100⟩, ⟨T200⟩	
19	⟨**F220**⟩ **Benutzeraufgaben bewerten**	⟨T100⟩, ⟨T200⟩	
20	⟨**F30**⟩ **Nutzer abmelden**	⟨T300⟩	

3.2 Testverfahren

Für das Back-End werden in JUnit Test-Funktionen programmiert, die die Funktionen automatisiert überprüft.
Front-End-Funktionen, in denen es sich um Eingabe-Ausgabe-Abläufe handelt, werden auf Basis von Äquivalenz-Klassen manuell getestet.

3.3 Testfälle

Im Folgenden werden die Testfälle aufgelistet, die Reihenfolge der Durchführung ergibt sich nach der in Kapitel 3.1 genannten Reihenfolge.

3.3.1 Testfall $\langle T100 \rangle$ - Administrationstool - Adminoperationen

Ziel

Der Zweck des Tests ist es alle Funktionen, welche das Administrationstool den Admins zur Verfügung stellt, ausführlich zu testen. Dabei werden auch gleich die Funktionen Trivia-Aufgaben erstellen und Trivia-Aufgaben bewerten getestet, die allen beförderten Nutzern zur Verfügung stehen.

Objekte/Methoden/Funktionen

$\langle \mathbf{F190} \rangle, \langle \mathbf{F200} \rangle, \langle \mathbf{F210} \rangle, \langle \mathbf{F220} \rangle, \langle \mathbf{F230} \rangle$

Pass/Fail Kriterien

Es meldet sich ein Admin zum Administrationstool an. Dieser führt die beschriebenen Funktionen aus. Falls alle Operationen erfolgreich ausgeführt werden können, gilt der Test als erfolgreich. Sollte der Admin aufgrund falscher Rechtevergabe eine der Funktionen nicht ausführen können oder sollte eine Operation nicht erfolgreich beendet werden können, gilt der Test als fehlgeschlagen.

Vorbedingung

Um den Test durchzuführen muss der testende Nutzer Adminrechte haben. Außerdem muss für die Funktionen $\langle \mathbf{F190} \rangle$ und $\langle \mathbf{F200} \rangle$ bereits ein Benutzerdatensatz in der Datenbank vorhanden sein.

Einzelschritte

Eingabe:

1. Als Admin zum Administrationstool anmelden
2. Auf den Button „Promote User“klicken
3. Den zu befördernden Nutzer über Eingabe des Benutzernamens beziehungsweise der y-Nummer suchen.
4. Benutzer befördern
5. Benutzer Adminrechte geben
6. Auf den Button „Create Task “ klicken und eine Trivia-Aufgabe erstellen
7. Auf den Button „Rate Task“ klicken, die eben erstellte Aufgabe suchen und bewerten
8. Auf den Button „Create Homework“ klicken und Hausaufgabenpaket erstellen
9. vom Admintool abmelden

Beobachtungen / Log / Umgebung

Ausgabe: Zu den Ausgaben gehören die Bestätigungen, dass die Operationen erfolgreich ausgeführt wurden, sowie die erstellten Aufgaben samt Bewertung. Der beförderte Nutzer hat zudem seine neuen Rechte erhalten. Bei einem Fehlschlag werden Fehlermeldungen als Ausgaben gegeben.

Besonderheiten

Abhängigkeiten

Der Testfall ist von ⟨**T300**⟩ abhängig, da die Registrierung von Nutzern ordnungsgemäß funktionieren muss.

3.3.2 Testfall $\langle T200 \rangle$ - Administrationstool - Studentenoperationen

Ziel

Der Zweck des Tests ist es alle Funktionen, welche das Administrationstool den Studenten zur Verfügung stellt, ausführlich zu testen. Dabei werden auch gleich die Funktionen Trivia-Aufgaben erstellen und Trivia-Aufgaben bewerten getestet, die allen beförderten Nutzern zur Verfügung stehen.

Objekte/Methoden/Funktionen

$\langle \mathbf{F180} \rangle, \langle \mathbf{F210} \rangle, \langle \mathbf{F220} \rangle$

Pass/Fail Kriterien

Es meldet sich ein Student zum Administrationstool an. Dieser führt die beschriebenen Funktionen aus. Falls alle Operationen erfolgreich ausgeführt werden können, gilt der Test als erfolgreich. Sollte der Student aufgrund falscher Rechtevergabe eine der Funktionen nicht ausführen sowie Funktionen für die er nicht berechtigt ist ausführen können oder sollte eine Operation nicht erfolgreich beendet werden können, gilt der Test als fehlgeschlagen.

Vorbedingung

Um den Test durchzuführen muss der testende Nutzer als Student registriert sein. Außerdem muss für die Funktion $\langle \mathbf{F180} \rangle$ bereits ein Hausaufgabenpaket bearbeitet worden sein.

Einzelschritte

Eingabe:

1. Als Student zum Administrationstool anmelden
2. Auf den Button „View Results “klicken
3. Ergebnisse überprüfen
4. Auf den Button „Create Task“ klicken und eine Trivia-Aufgabe erstellen
5. Auf den Button „Rate Task “ klicken, die eben erstellte Aufgabe suchen und bewerten
6. Vom Admintool abmelden

Beobachtungen / Log / Umgebung

Ausgabe: Zu den Ausgaben gehören die Bestätigungen, dass die Operationen erfolgreich ausgeführt wurden, sowie die erstellten Aufgaben samt Bewertung. Zudem werden die Hausaufgabenergebnisse angezeigt. Bei einem Fehlschlag werden Fehlermeldungen als Ausgaben gegeben.

Besonderheiten

Abhängigkeiten

Der Testfall ist von ⟨**T300**⟩ abhängig, da die Registrierung von Nutzern ordnungsgemäß funktionieren muss. Außerdem ist er von ⟨**T100**⟩ und ⟨**T600**⟩ abhängig, da die Erstellung und Bearbeitung von Hausaufgaben im Vorfeld einmal durchgeführt werden müssen.

3.3.3 Testfall $\langle T300 \rangle$ - Registrierung und Zugang zum Spiel

Ziel

Der Zweck dieses Tests ist es alle Funktionen, welche die Registrierung und den Zugang zum Spiel betreffen, ausführlich zu testen.

Objekte/Methoden/Funktionen

$\langle \mathbf{F10} \rangle, \langle \mathbf{F20} \rangle, \langle \mathbf{F30} \rangle$

Pass/Fail Kriterien

Ein Nutzer führt die Registrierung zum Spiel durch. Wenn er sich mit seinen gewählten Nutzerdaten zum Spiel anmelden kann (und auch nur mit diesen Daten) und danach auch ordnungsgemäß abgemeldet wird, gilt der Test als bestanden. Falls er allerdings auch mit falschen Daten Zugang erhält, sich mit seinen gewählten Daten nicht anmelden kann oder die Registrierung beziehungsweise die Abmeldung nicht korrekt ausgeführt werden, gilt der Test als fehlgeschlagen.

Vorbedingung

Es gibt keine Vorbedingungen.

Einzelschritte

Eingabe:

1. Die URL des Spiels aufrufen
2. Gewünschte Nutzerdaten eingeben
3. Auf den Button „Sign Up “
4. Mit den registrierten Daten über „Login “anmelden
5. Über den Button „Sign Out “abmelden

Beobachtungen / Log / Umgebung

Ausgabe: Es werden bei einem Fehlschlag Fehlermeldungen ausgegeben.

Besonderheiten

Wenn sich ein Student über seine y-Nummer registriert, muss er als Student erkannt werden.

Abhängigkeiten

Es gibt keine weiteren Abhängigkeiten.

3.3.4 Testfall $\langle T400 \rangle$ - The SQL-Alchemist - Oberfläche und Einstellungen

Ziel

Der Zweck dieses Tests ist es alle Funktionen, die in den verschiedenen Menüs angeboten werden zu testen. Dabei werden die direkt auf die Spielmodi bezogenen Funktionen nicht berücksichtigt.

Objekte/Methoden/Funktionen

$\langle$**F40**$\rangle$, $\langle$**F50**$\rangle$, $\langle$**F60**$\rangle$, $\langle$**F70**$\rangle$, $\langle$**F80**$\rangle$, $\langle$**F90**$\rangle$, $\langle$**F160**$\rangle$, $\langle$**F170**$\rangle$

Pass/Fail Kriterien

Ein Nutzer führt die verschiedenen Funktionen aus. Da es sich bei diesen ausschließlich um Optionen zur Änderung bestimmter Elemente des Spiels, beziehungsweise einfache Anzeigen handelt gilt der Test als Erfolg, falls Änderungen in den Optionen korrekt übernommen werden und alle Anzeigen wie gefordert angezeigt werden. Ein Fehlschlag tritt auf, wenn eine der gerade genannten Bedingungen nicht zutrifft.

Vorbedingung

Der Nutzer muss sich bereits einen Account erstellt haben und im Spiel angemeldet sein.

Einzelschritte

Eingabe:

1. Auf „Profile “ klicken und Anzeige überprüfen
2. Ins Hauptmenü wechseln und auf „Settings “ klicken
3. Auf „Change Username“ klicken
4. Neuen Benutzernamen eingeben und bestätigen
5. Kontrollieren ob Änderungen übernommen wurden
6. Auf „Change Password “ klicken
7. Neues Passwort eingeben und bestätigen
8. Kontrollieren ob Änderungen übernommen wurden
9. Auf den Avatar klicken und diesen ändern
10. Kontrollieren ob Änderungen übernommen wurden
11. Auf „Sound On/Off “ und „Music On/Off “ klicken
12. Ins Hauptmenü wechseln und auf „Leaderboards “ klicken
13. Im Textfeld den Benutzernamen des aktuellen Nutzers eingeben und bestätigen
14. Zu „Settings “ wechseln und über „Delete User “ den eigenen Account löschen

Beobachtungen / Log / Umgebung

Ausgabe: Die geänderten Einstellungen werden vom Spiel übernommen und dargestellt. Außerdem werden die Ranglisten und das Profil angezeigt. Bei Fehlern werden entsprechende Meldungen ausgegeben.

Besonderheiten

Abhängigkeiten

Der Testfall ist von ⟨**T300**⟩ abhängig, da die Registrierung von Nutzern ordnungsgemäß funktionieren muss.

3.3.5 Testfall $\langle T500 \rangle$ - The SQL-Alchemist - Story-Modus

Ziel

Der Zweck dieses Tests ist es diejenigen Funktionen, welche zum Spielen der Story verwendet werden zu überprüfen. Dabei werden die bereits in $\langle$**T300**$\rangle$ getesteten Funktionen nicht betrachtet.

Objekte/Methoden/Funktionen

$\langle$**F100**$\rangle$, $\langle$**F110**$\rangle$, $\langle$**F120**$\rangle$, $\langle$**F130**$\rangle$, $\langle$**F140**$\rangle$

Pass/Fail Kriterien

Ein Nutzer spielt den Story-Modus. Dabei gilt der Testlauf als Erfolg, falls der Nutzer einen Durchlauf durch den SQL-Trainer und das Minispiel, ohne Zwischenfälle, absolvieren kann. Als Fehlschlag zählt er, wenn es zu Fehlern in der Ausführung kommt und eine der Funktionen nicht durchgeführt werden kann.

Vorbedingung

Der Nutzer muss sich bereits einen Account erstellt haben und im Spiel angemeldet sein.

Einzelschritte

Eingabe:

1. Auf „Story-Mode “klicken
2. Das automatisch gestartete Tutorial absolvieren
3. Die Anzeige des „Laboratory “überprüfen
4. Den SQL-Trainer über die „Scrollcollection “starten
5. Ein SQL-Statement bearbeiten
6. Das Minispiel über das „Dungeon “starten
7. Die ersten vier Level abschließen
8. Überprüfen ob das fünfte Level ein „Endlevel “ist
9. Zurück zum „Laboratory “wechseln
10. Den „Story-Mode “verlassen
11. Die begonnene Story im „Settings “-Menü über „Story reset “zurücksetzen
12. Überprüfen ob die Änderung übernommen wurden

Beobachtungen / Log / Umgebung

Ausgabe: Im SQL-Trainer und im Minispiel werden die erreichten Ergebnisse angezeigt. Außerdem werden die einzelnen Menüs und die Spielbereiche grafisch dargestellt. Bei einem Fehlschlag werden entsprechende Meldungen ausgegeben.

Besonderheiten

Abhängigkeiten

Der Testfall ist von ⟨**T300**⟩ abhängig, da die Registrierung von Nutzern ordnungsgemäß funktionieren muss.

3.3.6 Testfall ⟨$T600$⟩ - The SQL-Alchemist - Hausaufgaben- und Trivia-Modus

Ziel

Der Zweck dieses Tests ist es, die auf den Hausaufgaben- und den Trivia-Modus bezogenen Funktionen zu prüfen. Dabei werden die bereits in ⟨**T300**⟩ getesteten Funktionen nicht betrachtet.

Objekte/Methoden/Funktionen

⟨**F130**⟩, ⟨**F150**⟩

Pass/Fail Kriterien

Ein Nutzer spielt den Hausaufgaben-Modus. Dabei gilt der Testlauf als Erfolg, falls der Nutzer einen Durchlauf durch den SQL-Trainer ohne Zwischenfälle, absolvieren und auch die Hausaufgaben wie geplant bearbeiten kann. Als Fehlschlag zählt er, wenn es zu Fehlern in der Ausführung kommt und eine der Funktionen nicht durchgeführt werden kann.

Vorbedingung

Der Nutzer muss sich einen Account erstellt haben und im Spiel als Student angemeldet sein. Außerdem muss bereits ein Hausaufgabenpaket erstellt worden sein.

Einzelschritte

Eingabe:

1. Auf „Homework „ klicken
2. Die gestellten SQL-Statements bearbeiten bis das Aufgabenpaket vollständig gelöst wurde
3. Überprüfen ob die bearbeiteten Aufgaben richtig analysiert werden

Beobachtungen / Log / Umgebung

Ausgabe: Im SQL-Trainer werden die erreichten Ergebnisse angezeigt. Außerdem werden die einzelnen Menüs grafisch dargestellt. Bei einem Fehlschlag werden entsprechende Meldungen ausgegeben.

Besonderheiten

Abhängigkeiten

Der Testfall ist von ⟨**T300**⟩ abhängig, da die Registrierung von Nutzern ordnungsgemäß funktionieren muss. Zudem ist der Testfall von ⟨**T100**⟩, genauer vom Erstellen der Hausaufgabenpakete abhängig.

4 Integrationstest

Integrationstests dienen dazu, das Zusammenspiel einzelner Komponenten zu testen. Dabei wird speziell in dieser Software darauf Wert gelegt, dass die einzelnen Schnittstellen die richtigen Daten sichtbar machen und diese auch, solange keine Änderungen vorgenommen werden, auch konsistent bleiben. Die soll zum einen mit Einblick auf die Datenbank als auch mit mehrmaligen Ausführen der zur zu testenden Schnittstelle passenden Funktionen gewährleistet werden.

4.1 Zu testende Komponenten

Da im Projekt nur zwei Komponenten vorhanden sind und diese wenig aussagekräftig erscheinen werden hier, anstatt der Komponenten, die zu testenden Schnittstellen aufeführt:

Nr	Schnittstelle	Testfälle	Kommentar
1	<I10> Session	T700	
2	<I20> Userdaten	T700	
3	<I40> SQL-Aufgaben	T800	
4	<I50> Ranglisten	T900	
5	<I60> Profildaten	T1000	

4.2 Testverfahren

Die Tests wurden manuell und mit Einsicht auf die Datenbank durchgeführt.

4.2.1 Testskripte

Es wurden keine Testskripte verwendet.

4.3 Testfälle

4.3.1 Testfall $\langle T700 \rangle$ - I10, Session und I20, Userdaten

Ziel

Ziel des Tests ist es, die dauerhaft korrekte Erstellung und Schließung von Sessions zu überprüfen. Da dafür unter anderem das Einloggen nötig ist, wird dabei gleichzeitig die korrekte Speicherung der Userdaten getestet.

Objekte/Methoden/Funktionen

$\langle \mathbf{F20} \rangle, \langle \mathbf{F30} \rangle$

Pass/Fail Kriterien

Mehrfaches an- und abmelden induziert korrektes erstellen und schließen von Sessions sowie korrekte Speicherung und Kontrolle der Userdaten. Der Tester sollte also nach dem Einloggen in das Hauptmenü und nach dem Ausloggen in den Startscreen weitergeleitet werden. Geschieht dies fehlerfrei gelten beide Tests als bestanden.

Vorbedingung

Der Tester muss sich in der Software registriert haben.

Einzelschritte

1. Auf die Seite des SQL-Alchemist gehen
2. Den Start-Screen bestätigen
3. Accountdaten eingeben
4. Zum Hauptmenü weitergeleitet werden
5. Logout-Button betätigen
6. Zum Start-Screen weitergeleitet werden
7. Wiederhole ab 2.

Beobachtungen / Log / Umgebung

Dem Tester sollte nach dem Einloggen das Hauptmenü und nach dem Ausloggen der Startscreen angezeigt werden.

Besonderheiten

–

Abhängigkeiten

–

4.3.2 Testfall $\langle T800 \rangle$ - I40, SQL-Aufgaben

Ziel

Ziel des Tests ist es, die richtige Verteilung von SQL-Aufgaben auf die verschiedenen Schwierigkeitsgerade und den Informationsaustausch zwischen Front-End und Teamprojekt zu testen. Die Kontrolle der SQL-Lösungen gehört in die Zuständigkeit des Teamprojekts und wird hier nicht getestet.

Objekte/Methoden/Funktionen

$\langle$**F130**$\rangle$

Pass/Fail Kriterien

Es wird getestet, ob nach der Auswahl eines Scwierigkeitsgrads eine passende Aufgabe angezeigt wird und ob auf eine eingegebene Lösung eine Antwort des Programms erfolgt. Geschieht dies ohne Probleme gilt der Test als bestanden

Vorbedingung

Der Tester muss sich in die Software eingeloggt haben.

Einzelschritte

1. Den Trivia-Mode starten
2. Einen Schwierigkeitsgrad auswählen
3. Anwort auf Aufgabenstellung eingeben
4. Wiederhole ab 2. mit neuem Schwierigkeitsgrad

Beobachtungen / Log / Umgebung

Dem Tester sollte nach abschicken der potentiellen Antwort ausgegeben werden, ob die Antwort falsch oder richtig ist.

Besonderheiten

Die korrektheit der Ausgabe wird hierbei nicht beachtet.

Abhängigkeiten

Mit diesem Test werden gleichzeitig die Funktionen für die SQL-Aufgaben in der Story und die Kommunikation für den Hausaufgabenmodus getestet, da sie nach dem gleichen Schema funtkionieren.

4.3.3 Testfall $\langle T900 \rangle$ - I50, Ranglisten

Ziel

Ziel des Tests ist es, die korrekte Sortierung der Spielstände zu testen.

Objekte/Methoden/Funktionen

$\langle \mathbf{F160} \rangle$

Pass/Fail Kriterien

Wenn die Spieler in der richtigen Reihenfolge, bezogen auf das ausgewählte Kriterium, aufgelistet werden, gilt der Test als bestanden.

Vorbedingung

Der Tester muss sich in die Software eingeloggt haben.

Einzelschritte

1. Die Rankings aufrufen
2. Die Rankings nach „Lofi-Coins“sortieren
3. Die Rankings nach „Spend Time“sortieren
4. Die Rankings nach „Runs“sortieren
5. Die Rankings nach „SQL-Statements“sortieren
6. Die Rankings nach „Success Rate“sortieren
7. Die Rankings nach „Score“sortieren

Beobachtungen / Log / Umgebung

Die Ranglisten sollten sich anhand des ausgewählten Kriteriums, absteigend sortieren.

Besonderheiten

–

Abhängigkeiten

–

4.3.4 Testfall $\langle T1000 \rangle$ - l60, Profil

Ziel

In diesem Test so überprüft werden, dass die Profildaten des eingeloggten Users korrekt angezeigt werden.

Objekte/Methoden/Funktionen

$\langle$**F40**$\rangle$

Pass/Fail Kriterien

Wenn dem Tester sein aktueller Avatar, Username, aktuellen und insgesamt verdienten Coins, seine Gesamtpunktzahl, Run-Anzahl, mit SQL-Aufgaben verbrachte Zeit, die Anzahl an gelösten Statements und die Erfolgsrate für SQL-Aufgaben angezeigt werden, gilt der Test als bestanden.

Vorbedingung

Der Tester muss sich in die Software eingeloggt haben.

Einzelschritte

1. Die Rankings aufrufen

Beobachtungen / Log / Umgebung

Dem Tester sollten seine Profildaten angezeigt werden.

Besonderheiten

–

Abhängigkeiten

–

5 Unit-Tests

In den Unittests wird die Funktion der einzelnen Komponenten in sich getestet. Da dies für das Front-End, da in Javascript entwickelt, recht umständlich und zeitaufwändig ist, wurde dieser Testabschnitt durchgeführt. Ziel ist es sicher zu stellen, dass die programmierten Klassen und Methoden ihren jeweiligen Zweck zuverlässig erfüllen.

Testziel...

5.1 Zu testende Komponenten

Nr	Komponenten	Testfälle	Kommentar
1	C20, Back-End	T1100	

5.2 Testverfahren

Die Tests wurden mit JUnit durchgeführt.

5.2.1 Testskripte

Es wurden keine speziellen Testskripte verwendet.

5.3 Testfälle

5.3.1 Testfall $\langle T1100 \rangle$ - Klasse x

Ziel Ziel ist es, die ordnungsgemäße Funktionstüchtigkeit der verschiedenen Klassen und Methoden des Back-Ends sicher zu stellen.

Objekte/Methoden/Funktionen Die jeweiligen Methoden oder Klassen sind im jeweiligen Abschnitt in den Testprotokollen zu finden.

Pass/Fail Kriterien Der Gesamttest gild als bestanden, wenn jede Klasse und MEthode ihren gewünschten Zweck erfüllt.

Vorbedingung

Einzelschritte Zu jeder Klasse und Methode werden JUnit-Tests programmiert und die wichtigsten wurden in den Testdokumenten Protokolliert.

Beobachtungen / Log / Umgebung

Besonderheiten

Abhängigkeiten