

Entwicklung eines Lagerabrufs- und Verwaltungssystems mit Datenbank und Android-Anwendung zum Einsatz in der Gastronomie

Studienarbeit

3. Studienjahr

Modul T3100

des Studienganges Mechatronik

an der Dualen Hochschule Baden-Württemberg

am Standort Stuttgart

von

Marvin Mai und Daniel Schifano

Bearbeitungszeitraum

12 Wochen

Betreuer der dualen Hochschule

Prof. Dr.-Ing. Johannes Moosheimer

Ehrenverantwortliche Erklärung

Gemäß § 5 (2) der „Studien- und Prüfungsordnung DHBW Technik“ vom 06. November 2013.

Ich habe die vorliegende Arbeit selbstständig verfasst und keine anderen, als die angegebenen Quellen und Hilfsmittel verwendet.

Stuttgart, 09.01.2018

Ort, Datum

Unterschrift

Stuttgart, 09.01.2018

Ort, Datum

Unterschrift

Inhaltsverzeichnis

Ehrenverantwortliche Erklärung.....	II
Abbildungsverzeichnis	V
1 Anforderungen.....	1
1.1 Aufgabenstellung.....	1
2 Pflichtenheft.....	2
2.1 Zielbestimmung	2
2.1.1 Musskriterien.....	2
2.1.2 Wunschkriterien	2
2.2 Produkteinsatz.....	3
2.2.1 Anwendungsbereiche	3
2.2.2 Zielgruppen	3
2.2.3 Betriebsbedingungen	3
2.3 Produktübersicht.....	4
2.4 Produktfunktionen.....	5
2.4.1 Anwendungsfälle des Kassensystem-Managers.....	5
2.4.2 Anwendungsfälle der Android-Anwendung	14
2.5 Produktdaten	19
2.6 Produktleistungen.....	20
2.7 Benutzungsoberfläche.....	21
2.7.1 Android-Application.....	21
2.7.2 Lokale Benutzeranwendung.....	22
2.8 Nichtfunktionale Anforderungen	23
2.9 Technische Produktumgebung.....	23
2.9.1 Software.....	23
2.9.2 Hardware	23
3 Entwurf	24
3.1 Klassendesign des Datenbank-Systems	24

3.2	Klassendesign der Android-Anwendung.....	26
4	Implementierung.....	28
4.1	Implementierung des Datenbank-Systems.....	28
4.1.1	Klasse DatabaseService_Interface	28
4.1.2	Klasse RestApiController	34
4.2	Implementierung der Android-Anwendung	37
4.2.1	Klasse TableSelect	37
4.2.2	Klasse MainActivity	38
5	Test	42
5.1	Test des Datenbank-Systems	42
5.1.1	Benutzeranwendung Kassensystem-Manager.....	42
5.2	Test der Android-Anwendung.....	60
6	Fazit.....	75
7	Anhang.....	76
7.1	Installationsanweisung	76
7.1.1	Datenbank-System	76
7.1.2	Android Applikation	78
8	Testdokumentation	79
8.1	Datenbank-Modul	80
8.1.1	DatabaseService-Klasse.....	80
8.1.2	Server-Modul	112
8.1.3	Benutzeranwendung	116

Abbildungsverzeichnis

Abbildung 1: Systemübersicht des Projekts.....	4
Abbildung 2: Anwendungsfalldiagramm des Kassensystem-Managers	6
Abbildung 3: Anwendungsfalldiagramm der Android-Anwendung.....	14
Abbildung 4: Strukturierung der Datenbank und Abhängigkeiten der Tabellen	19
Abbildung 5: Entwurf der Benutzeroberfläche der Android-Applikation	21
Abbildung 6: Entwurf der Benutzeroberfläche des Kassensystem-Managers	22
Abbildung 7: Software-Package "database"	24
Abbildung 8: Das Software-Package "RestApiController"	25
Abbildung 9: Software-Package "Kassensystemapplikation"	26

1 Anforderungen

1.1 Aufgabenstellung

Laut Aufgabenstellung soll ein Lagerabrufs- und Verwaltungssystem mit Datenbank und Android-Anwendung entwickelt werden. Das System besteht aus einem zentralen Computer, der über ein Netzwerk mit beliebig vielen Handheld-Geräten verbunden werden kann. Das Netzwerk wird von einem Router via Wireless Local Area Network (WLAN) zur Verfügung gestellt.

Mit der Hilfe einer Manager-Applikation, die eine Benutzeroberfläche zur Datenbankverwaltung darstellt, soll auf dem Computer der Lagerbestand verwaltet und die Wareneingänge, beziehungsweise Warenausgänge gesteuert werden. Ebenfalls sollen in der Manager-Applikation Lagerartikel bearbeitet werden können.

Auf den Handheld-Geräten soll es möglich sein, die derzeit im Lager verfügbaren Artikel, inklusive deren Bestand, einzusehen. Ebenfalls soll es an den Handheld-Geräten möglich sein, kundenbezogen eine Bestellung zusammenzufassen. Nachdem der Bestellvorgang beendet wurde, soll mithilfe eines Bon-Druckers ein Bestellbon/Lieferschein ausgedruckt werden können.

Das gesamte System soll für den Gastronomiebereich ausgelegt werden.

2 Pflichtenheft

2.1 Zielbestimmung

2.1.1 Musskriterien

Datenbank

- Speicherung von Daten in vorgegebenen Strukturen
 - Artikel
 - Lagerbestand
 - Bestellungen
 - Tische
- Daten auf Anfrage ausgeben und bearbeiten

Datenbank-Service

- Bereitstellen einer Schnittstelle für Zugriffe auf die Datenbank
 - Verändern und Ausgeben der Tabelleninhalte
- Bestellung über einen Bon-Drucker ausdrucken

Lokale Benutzeranwendung

- Einsehen der Bestellungen
- Einsehen und Bearbeiten der Artikel, des Lagerbestandes und der Tische

Server

- Bedienen von Anfragen von beliebig vielen Clients
- Senden von Statusmeldungen und Datenbankinhalten

Client

- Tische, Lagerbestand und Artikel abrufen
- Bestellungen erstellen und an den Server senden
 - Empfangen von Statusmeldungen des Servers nach Absenden einer Bestellung
 - Nicht bezahlte Bestellung eines Tisches abrufen

2.1.2 Wunschkriterien

- Ergänzung der Datenbank
 - Mitarbeiter

- Anmeldedaten
- Ergänzung des Clients
 - Login-Dialog für Mitarbeiter
- Ergänzungen des Servers
 - Zugriffsberechtigungen beim Login überprüfen

2.2 Produkteinsatz

2.2.1 Anwendungsbereiche

- Bestellaufnahme und Verarbeitung in der Gastronomie
 - Lagerverwaltung

2.2.2 Zielgruppen

- Kellner: Bestellungen am Smartphone eingeben
- Köche: Empfangen der Bestellungen am Bon-Drucker
- Gastronom(Chef): Verwaltung des Lagerbestandes und Artikel

2.2.3 Betriebsbedingungen

- WLAN-Netzwerk, keine Internetanbindung notwendig
- Tägliche Betriebszeit: Dauerbetrieb

2.3 Produktübersicht

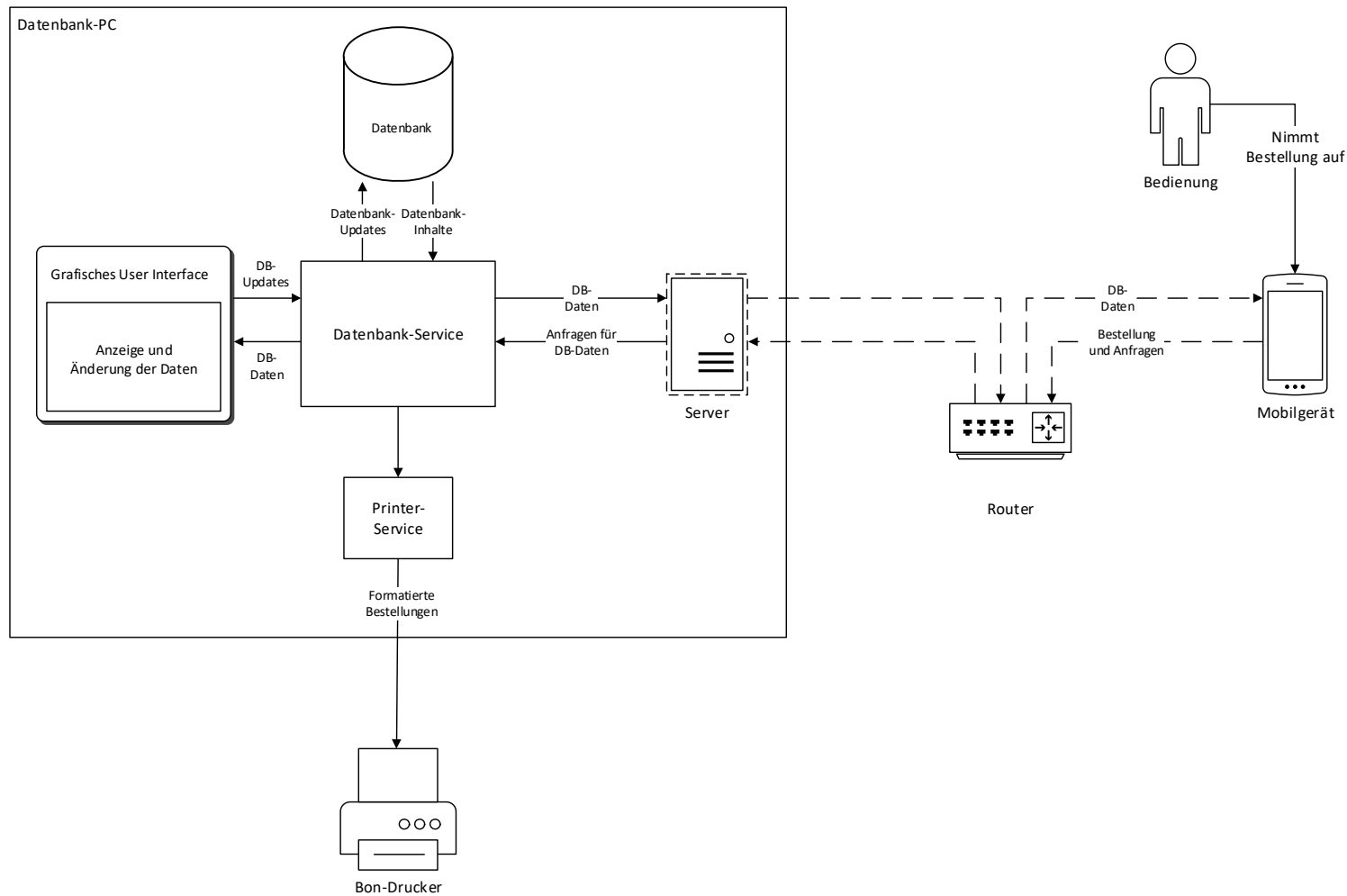


Abbildung 1: Systemübersicht des Projekts

2.4 Produktfunktionen

2.4.1 Anwendungsfälle des Kassensystem-Managers

Im Folgenden werden die möglichen Anwendungsfälle der Benutzeranwendung Kassensystem-Manager dargestellt und spezifiziert.

Zunächst werden im Anwendungsfalldiagramm alle Funktionen die der Benutzer aktiv anwendet, grafisch dargestellt.

Anschließend werden diese Funktionen in einer tabellarischen Auflistung genauer beschrieben.

2.4.1.1 Anwendungsfalldiagramm

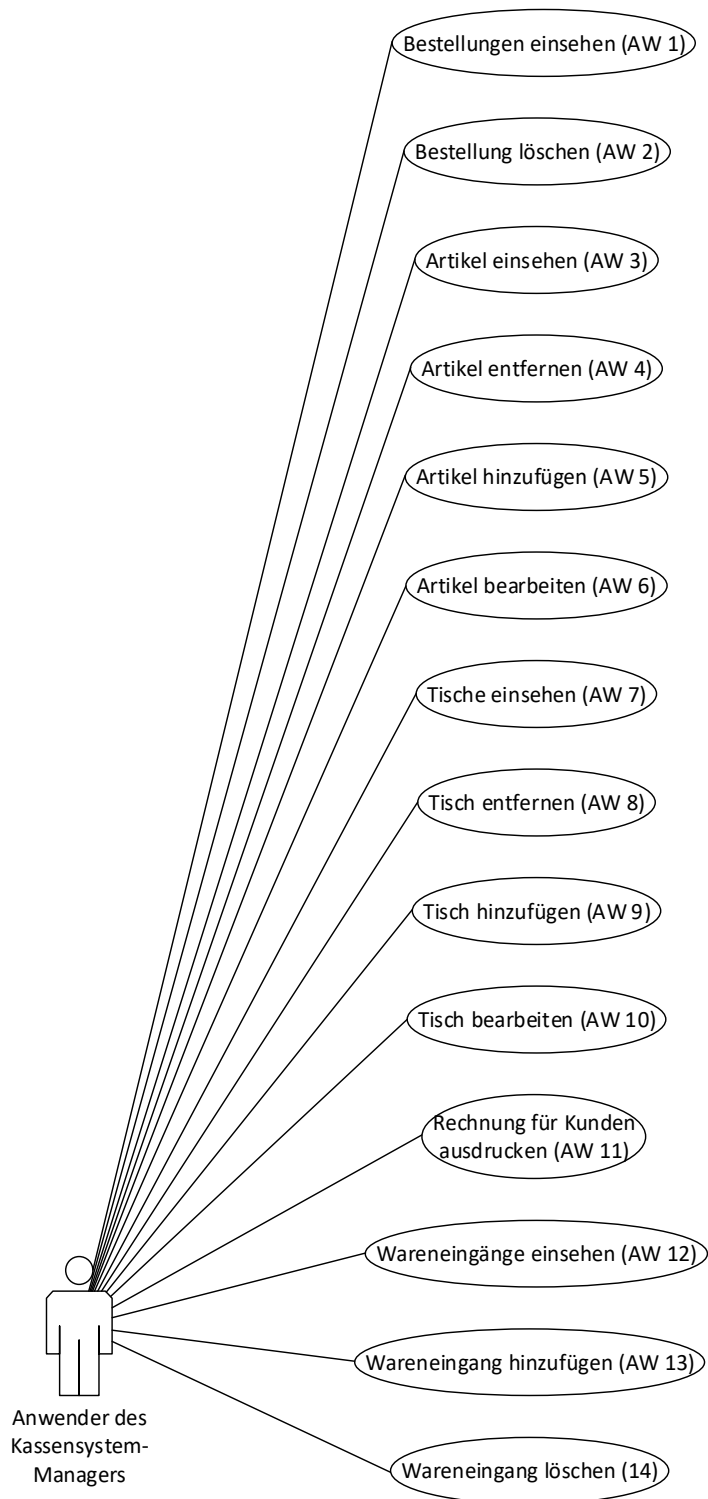


Abbildung 2: Anwendungsfalldiagramm des Kassensystem-Managers

2.4.1.2 Spezifizierung der Anwendungsfälle

Bezeichnung	1 – Bestellungen einsehen
Ziel	Der Anwender kann in einer tabellarischen Übersicht alle Bestellungen mit ihren Informationen wie bestellte Artikel, Preis, Datum und Tisch einsehen, die noch nicht bezahlt worden sind.
Vorbedingung	Es besteht der Bedarf, alle offenen Bestellungen einzusehen. Das ist beispielsweise der Fall, wenn ein Beleg für die Küche abhandengekommen ist und die Informationen über diese Bestellung benötigt werden.
Nachbedingung	Es wurden alle Informationen über die Bestellungen richtig in der Übersicht dargestellt. Es wurden nur offene Bestellungen angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt und keine Informationen.
Akteure	Gastronom, Koch

Bezeichnung	2 – Bestellung löschen
Ziel	Der Anwender kann über einen Rechtsklick auf eine Bestellung einen Menüeintrag zum Löschen einer Bestellung auswählen. Anschließend wird diese Bestellung endgültig aus der Datenbank gelöscht.
Vorbedingung	Es besteht der Bedarf eine Bestellung zu löschen. Das ist bspw. der Fall, wenn eine Bestellung fälschlicherweise aufgegeben wurde und storniert werden muss.
Nachbedingung	Die Bestellung wird nicht mehr in der Benutzeroberfläche angezeigt und wurde aus der Datenbank gelöscht.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom, Koch, Bedienung

Bezeichnung	3 – Artikel einsehen
Ziel	Der Anwender kann in einer tabellarischen Übersicht alle verfügbaren Artikel mit ihren Informationen wie Bezeichnung, Lagerbestand und Preis einsehen.
Vorbedingung	Es besteht der Bedarf, alle Artikel einzusehen, bspw. wenn das Sortiment neu strukturiert werden soll oder wenn neue Wareneingänge hinzugefügt werden sollen.
Nachbedingung	Es werden alle Artikel mit ihren Informationen korrekt angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt und keine Informationen.
Akteure	Gastronom

Bezeichnung	4 – Artikel entfernen
Ziel	Ein Artikel soll in der Datenbank als nicht verfügbar markiert und anschließend nicht mehr in der tabellarischen Übersicht angezeigt werden.
Vorbedingung	Es besteht der Bedarf einen Artikel zu entfernen, bspw. wenn dieser aus dem Sortiment genommen werden soll oder dieser fälschlicherweise angelegt wurde.
Nachbedingung	Der Artikel wurde in der Datenbank als nicht verfügbar markiert und wird nicht mehr in der tabellarischen Übersicht dargestellt. In bisherigen Bestellungen wird der Artikel aber trotzdem weiterhin angezeigt. Auch die Wareneingänge dieses Artikels werden nicht mehr angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	5 – Artikel hinzufügen
Ziel	Es wird ein neuer Artikel mit all seinen Informationen wie Bezeichnung und Verkaufspreis neu erstellt. Außerdem wird gleichzeitig ein neuer Wareneingang für diesen Artikel hinzugefügt.
Vorbedingung	Es besteht der Bedarf einen neuen Artikel anzulegen. Dies ist bspw. der Fall, wenn ein neuer Artikel das erste Mal geliefert wird. Dann wird gleichzeitig ein Wareneingang mit der entsprechend gelieferten Anzahl angelegt.
Nachbedingung	Der Artikel wurde neu in der Datenbank angelegt und wird in der tabellarischen Übersicht angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	6 – Artikel bearbeiten
Ziel	Die Informationen eines Artikels sollen bearbeitet werden. Der bisherige Artikel wird als nicht verfügbar markiert und der bearbeitete Artikel wird der Datenbank hinzugefügt.
Vorbedingung	Es besteht der Bedarf, die Informationen eines Artikels zu bearbeiten. Das ist bspw. der Fall, wenn sich der Preis oder die Bezeichnung eines Artikels ändert. Es muss der entsprechende Artikel angeklickt werden und anschließend im entsprechenden Formularfeld die neuen Informationen hinzugefügt werden.
Nachbedingung	Der Artikel wurde mit den bearbeiteten Informationen in der Datenbank erstellt und der bisherige Artikel als nicht verfügbar markiert. In der Übersicht wird der Artikel mit einer neuen ID dargestellt, der bisherige Artikel mit den alten Informationen allerdings nicht mehr.

Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	7 – Tische einsehen
Ziel	Es sollen alle verfügbaren Tische in einer tabellarischen Übersicht dargestellt werden.
Vorbedingung	Es besteht der Bedarf, die Tische einzusehen. Das ist bspw. der Fall, wenn der Verkaufsraum neu strukturiert werden soll.
Nachbedingung	Die Tische wurden korrekt in einer tabellarischen Übersicht angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	8 – Tisch entfernen
Ziel	Einer der Tische soll entfernt werden.
Vorbedingung	Es besteht der Bedarf, einen Tisch zu entfernen. Dies ist bspw. der Fall, wenn im Verkaufsraum ein Tisch abgebaut oder umgebaut werden soll.
Nachbedingung	Der Tisch wurde in der Datenbank als nicht verfügbar markiert und wird nicht mehr in der tabellarischen Übersicht angezeigt. In vergangenen Bestellungen soll der Tisch weiterhin abrufbar sein. Es muss über einen Rechtsklick auf den entsprechenden Tisch der Menüeintrag zum Löschen ausgewählt werden.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	9 – Tisch hinzufügen
Ziel	Es soll ein neuer Tisch zu der Datenbank hinzugefügt werden.
Vorbedingung	Es besteht der Bedarf, einen neuen Tisch anzulegen. Das ist bspw. der Fall, wenn ein neuer Tisch im Verkaufsraum aufgestellt werden soll.
Nachbedingung	Der Tisch wurde erfolgreich der Datenbank hinzugefügt und wird in der tabellarischen Übersicht angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	10 – Tisch bearbeiten
Ziel	Die Informationen eines Tisches sollen bearbeitet werden. Der Tisch mit den alten Informationen wird als nicht verfügbar markiert und ist weiterhin in vergangenen Bestellungen abrufbar.
Vorbedingung	Es besteht der Bedarf, einen Tisch zu bearbeiten. Dies ist bspw. der Fall, wenn ein Tippfehler passiert ist oder die Namensvergebung der Tische in der Gastronomie geändert werden soll. Es muss der entsprechende Tisch angeklickt werden und anschließend im entsprechenden Formularfeld die neuen Informationen hinzugefügt werden.
Nachbedingung	Der Tisch wird mit seinen neuen Informationen in der tabellarischen Übersicht dargestellt und in der Datenbank neu hinzugefügt. Der alte Artikel wurde als nicht verfügbar markiert.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	11 – Rechnung für Kunden ausdrucken
Ziel	Der Anwender möchte eine schon existierende Bestellung ausdrucken, bspw. weil der Beleg verloren gegangen oder verschmutzt worden ist.
Vorbedingung	Es besteht der Bedarf eine Bestellung ein weiteres Mal auszudrucken. Über einen Rechtsklick auf die Bestellung kann ein Menüeintrag ausgewählt werden, über den dann die Bestellung ausgedruckt wird.
Nachbedingung	Es wurde über den Bon-Drucker erfolgreich und korrekt eine Bestellung in Form eines Kundenbeleges ausgedruckt.
Nachbedingung im Sonderfall	Der Bon wurde nicht ausgedruckt, weil der Drucker nicht installiert oder ausgeschaltet ist.
Akteure	Bedienung, Gastronom

Bezeichnung	12 – Wareneingänge einsehen
Ziel	Es sollen alle Wareneingänge von verfügbaren Artikeln in einer tabellarischen Übersicht dargestellt werden.
Vorbedingung	Es besteht der Bedarf, die Wareneingänge einzusehen. Dies ist bspw. der Fall, wenn eine bisherige Eingabe kontrolliert werden soll oder eine falsche Eingabe gelöscht werden soll.
Nachbedingung	Es werden alle Wareneingänge in einer tabellarischen Übersicht dargestellt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	13 – Wareneingang hinzufügen
Ziel	Es soll ein neuer Wareneingang für einen Artikel hinzugefügt werden.
Vorbedingung	Es besteht der Bedarf, einen neuen Wareneingang hinzuzufügen. Das ist bspw. der Fall, wenn vom Lieferanten eine neue Lieferung erhalten wurde und diese in das Warensystem eingepflegt werden soll. Es muss in der Artikelübersicht ein Artikel ausgewählt und dann im entsprechenden Formularfeld die Anzahl eingegeben werden.
Nachbedingung	Der neue Wareneingang wurde in der Wareneingangsübersicht dargestellt und der Datenbank hinzugefügt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

Bezeichnung	14 – Wareneingang löschen
Ziel	Ein Wareneingang soll aus der Datenbank und der tabellari-schen Übersicht gelöscht werden.
Vorbedingung	Es besteht der Bedarf, einen Wareneingang zu löschen. Das ist bspw. der Fall, wenn ein Wareneingang fälschlicherweise angelegt wurde oder eine Lieferung nach dem Hinzufügen in das Warensystem beschädigt wurde.
Nachbedingung	Der Wareneingang wurde endgültig aus der Datenbank und der tabellarischen Übersicht entfernt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Gastronom

2.4.2 Anwendungsfälle der Android-Anwendung

Im Folgenden werden die möglichen Anwendungsfälle der Android-Anwendung dargestellt und spezifiziert.

2.4.2.1 Anwendungsfalldiagramm

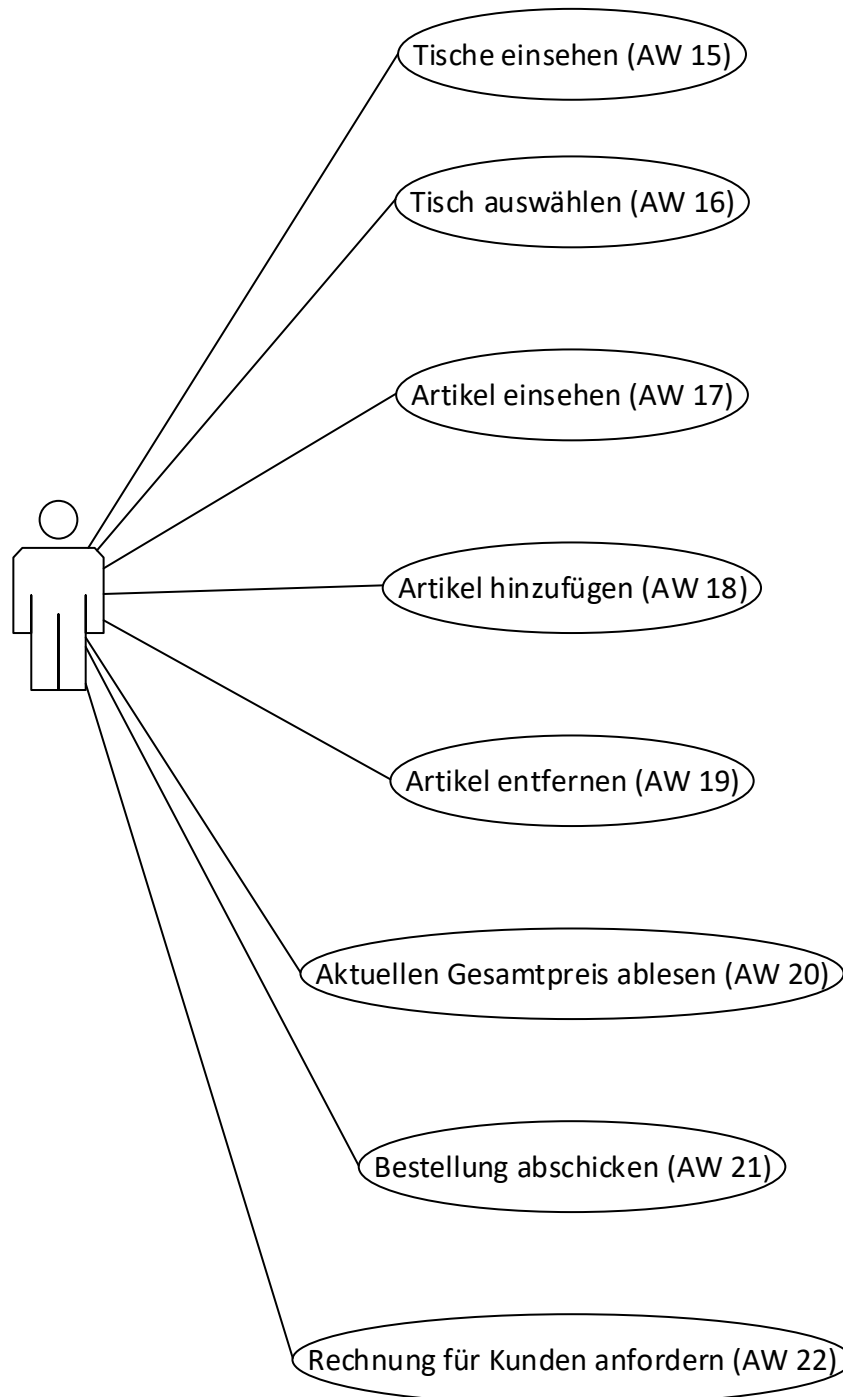


Abbildung 3: Anwendungsfalldiagramm der Android-Anwendung

2.4.2.2 Spezifizierung der Anwendungsfälle

Bezeichnung	15 - Tische einsehen
Ziel	Der Anwender kann in einem Dropdown-Menü alle verfügbaren Tische einsehen.
Vorbedingung	Es besteht der Bedarf, eine Bestellung aufzunehmen. Dieser entsteht immer, wenn ein Kunde eine Bestellung aufgeben möchte.
Nachbedingung	Es wurden alle Informationen richtig in dem Dropdown-Menü dargestellt. Es wurden nur verfügbare Tische angezeigt.
Nachbedingung im Sonderfall	Es wird eine Fehlermeldung angezeigt.
Akteure	Bedienung

Bezeichnung	16 - Tisch auswählen
Ziel	Der Anwender kann aus dem Dropdown Menü den Tisch auswählen, an dem der Kunde bestellen möchte.
Vorbedingung	Es besteht der Bedarf, eine Bestellung von einem Kunden aufzunehmen. Ebenfalls befindet sich der Kunde an einem Tisch.
Nachbedingung	Der Tisch an dem sich der Kunde befindet wird ausgewählt und der Bestellung hinzugefügt.
Nachbedingung im Sonderfall	Der Tisch wird nicht ausgewählt und eine Fehlermeldung wird angezeigt.
Akteure	Bedienung

Bezeichnung	17 - Artikel einsehen
Ziel	Der Anwender kann in einer tabellarischen Übersicht alle verfügbaren Artikel mit ihren Informationen Preis und Lagerbestand einsehen.
Vorbedingung	Es besteht der Bedarf, eine Bestellung von einem Kunden aufzunehmen. Der Tisch an dem sich der jeweilige Kunde befindet wurde bereits ausgewählt.

Nachbedingung	Es wurden alle Informationen richtig in einer tabellarischen Übersicht dargestellt. Es wurden nur verfügbare Artikel angezeigt.
Nachbedingung im Sonderfall	Die Artikel werden nicht angezeigt und eine Fehlermeldung wird dargestellt.
Akteure	Bedienung

Bezeichnung	18 - Artikel hinzufügen
Ziel	Der Anwender kann einer Bestellung die ein Kunde aufgibt, Artikel hinzufügen.
Vorbedingung	Es besteht der Bedarf, eine Bestellung von einem Kunden aufzunehmen. Der Tisch an dem sich der Kunde befindet wurde bereits ausgewählt. Der Kunde möchte einen oder mehrere Artikel, die in der Speisekarte vorhanden sind bestellen.
Nachbedingung	Es wurden alle ausgewählten Artikel der Bestellung hinzugefügt. Inklusive Preis und Anzahl der Artikel die bestellt wurden.
Nachbedingung im Sonderfall	Die Artikel werden der Bestellung nicht hinzugefügt und eine Fehlermeldung erscheint.
Akteure	Bedienung

Bezeichnung	19 - Artikel entfernen
Ziel	Der Anwender kann bei einer Bestellung die ein Kunde aufgibt, Artikel wieder entfernen.
Vorbedingung	Der Kunde entscheidet sich während der Bestellung um und möchte einen anderen Artikel bestellen. Ebenfalls ist es möglich, dass die Bedienung zuerst den falschen Artikel eingibt, ihren Fehler jedoch bemerkt.
Nachbedingung	Der zu löschende Artikel wurde erfolgreich aus der vom Kunden aufgegebenen Bestellung entfernt.
Nachbedingung im Sonderfall	Die Artikel werden nicht aus der vom Kunden aufgegebenen Bestellung entfernt und eine Fehlermeldung wird angezeigt.
Akteure	Bedienung

Bezeichnung	20 - Aktuellen Gesamtpreis ablesen
Ziel	Der Anwender kann bei einer Bestellung die ein Kunde aufgibt den Gesamtpreis ablesen. Dieser setzt sich aus den einzelnen Artikeln, die der Bestellung hinzugefügt wurden, zusammen.
Vorbedingung	Es besteht der Bedarf, dass ein Kunde den Gesamtpreis wissen möchte. Dies tritt zum Beispiel ein, wenn der Kunde bezahlen möchte. Ebenfalls kann die Bedienung dem Kunden während einer Bestellung den bisherigen Preis der Bestellung übermitteln.
Nachbedingung	Der Gesamtpreis wurde richtig ermittelt und für den Anwender dargestellt.
Nachbedingung im Sonderfall	Eine Fehlermeldung wird dargestellt.
Akteure	Bedienung

Bezeichnung	21 - Bestellung abschicken
Ziel	Der Anwender kann die Bestellung die ein Kunde aufgibt an die Küche/Koch weiterleiten.
Vorbedingung	Es besteht der Bedarf, dass ein Kunde eine Bestellung aufgeben möchte. Hat der Kunde seine Bestellung bei einer Bedienung aufgegeben, soll diese Bestellung in der Küche bearbeitet werden, damit sie dem Kunden serviert werden kann.
Nachbedingung	Die Bestellung des Kunden wurde erfolgreich an die Küche übermittelt.
Nachbedingung im Sonderfall	Die Bestellung wird nicht übermittelt und eine Fehlermeldung wird dargestellt.
Akteure	Bedienung

Bezeichnung	22 - Rechnung für den Kunden anfordern
Ziel	Der Kunde bekommt für seine Bestellung einen Beleg ausgedruckt.
Vorbedingung	Es besteht der Bedarf, dass ein Kunde seine Bestellung bezahlen möchte/muss.
Nachbedingung	Die Artikel die der Kunde bestellt hat, sowie der Gesamtpreis werden auf einen Beleg ausgedruckt.
Nachbedingung im Sonderfall	Der Beleg wird nicht ausgedruckt und eine Fehlermeldung wird dargestellt
Akteure	Bedienung

2.5 Produktdaten

Die folgenden Daten müssen von der Datenbank gespeichert und verwaltet werden:

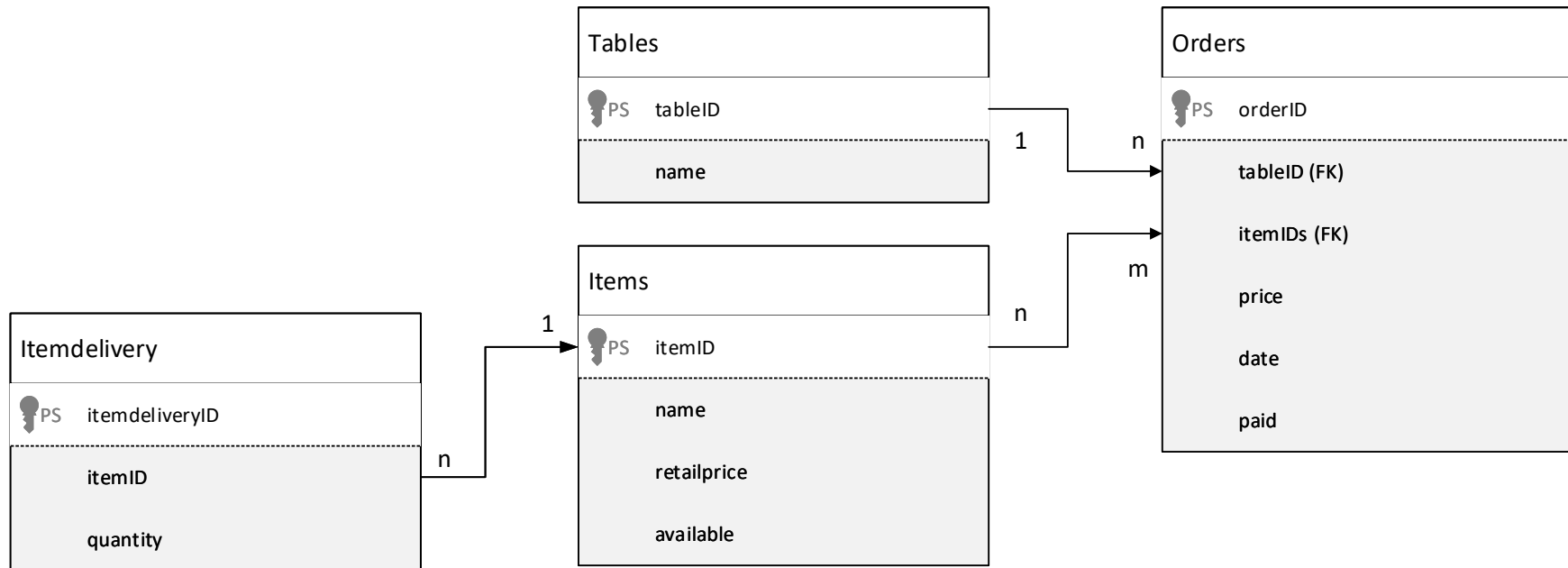


Abbildung 4: Strukturierung der Datenbank und Abhängigkeiten der Tabellen

Je nach Implementierung weiterer Funktionen wird dieses Datenmodell angepasst und erweitert. Diese werden gegebenenfalls separat dokumentiert.

2.6 Produktleistungen

Das Abrufen von Datenbankinhalten am Mobiltelefon sollte mit einer geringen Verzögerung ablaufen. Als maximale Dauer einer Aktualisierung mit großem Umfang werden 5 Sekunden angesetzt.

Das Abschicken einer Bestellung vom Mobiltelefon an die Datenbank bis zur nachfolgenden Bestätigung des Empfangs sollte ebenfalls 5 Sekunden nicht überschreiten.

An der lokalen Benutzeranwendung, die auf dem Datenbank-PC läuft, soll die Dauer der Datenaktualisierung nicht über 3 Sekunden liegen. Die Dauer für Änderungen an Daten bei üblichem Umfang soll nicht länger als 2 Sekunden betragen. Bei Änderung großer Datenbereiche sind 4 Sekunden das Maximum.

2.7 Benutzungsoberfläche

2.7.1 Android-Application

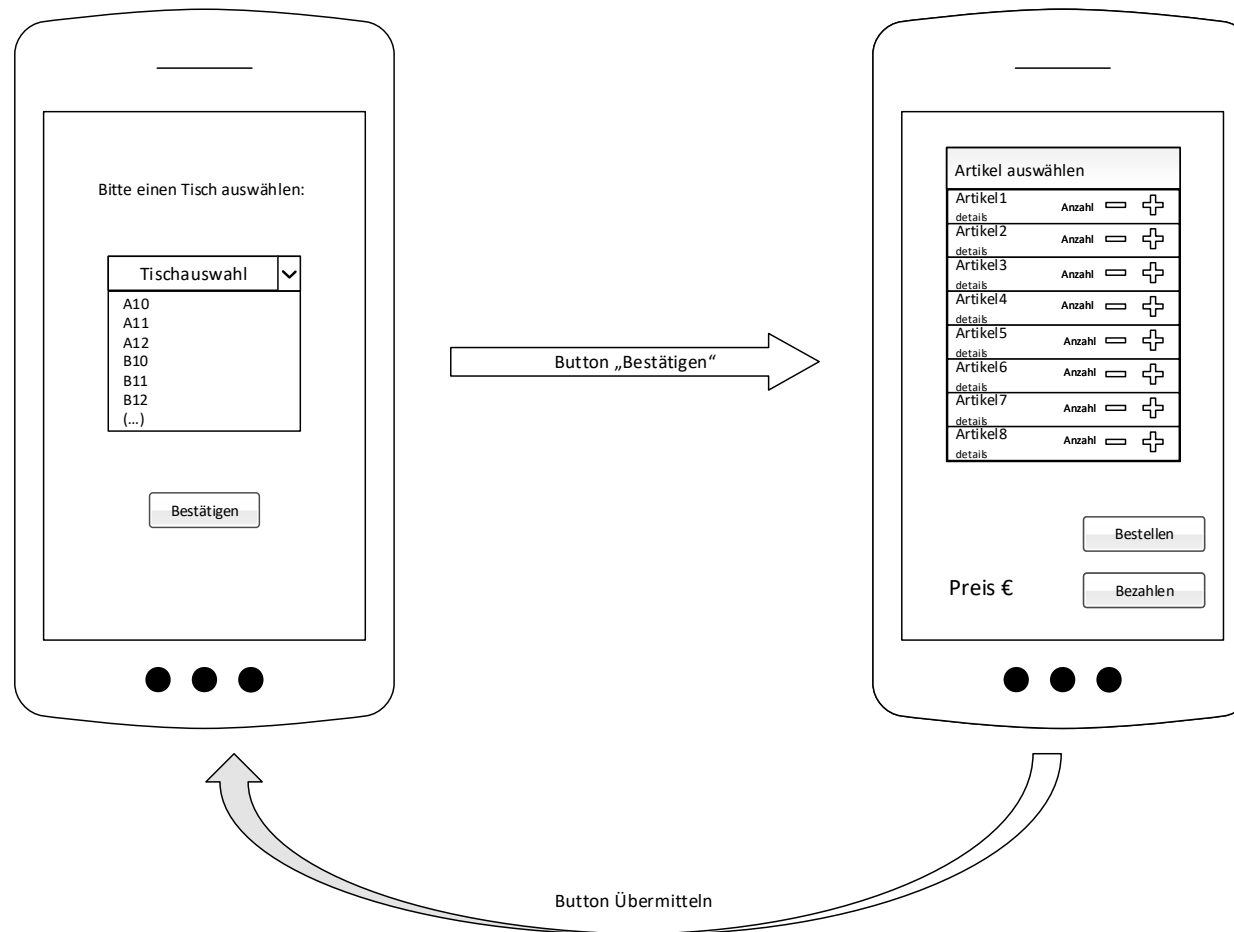


Abbildung 5: Entwurf der Benutzeroberfläche der Android-Applikation

2.7.2 Lokale Benutzeranwendung

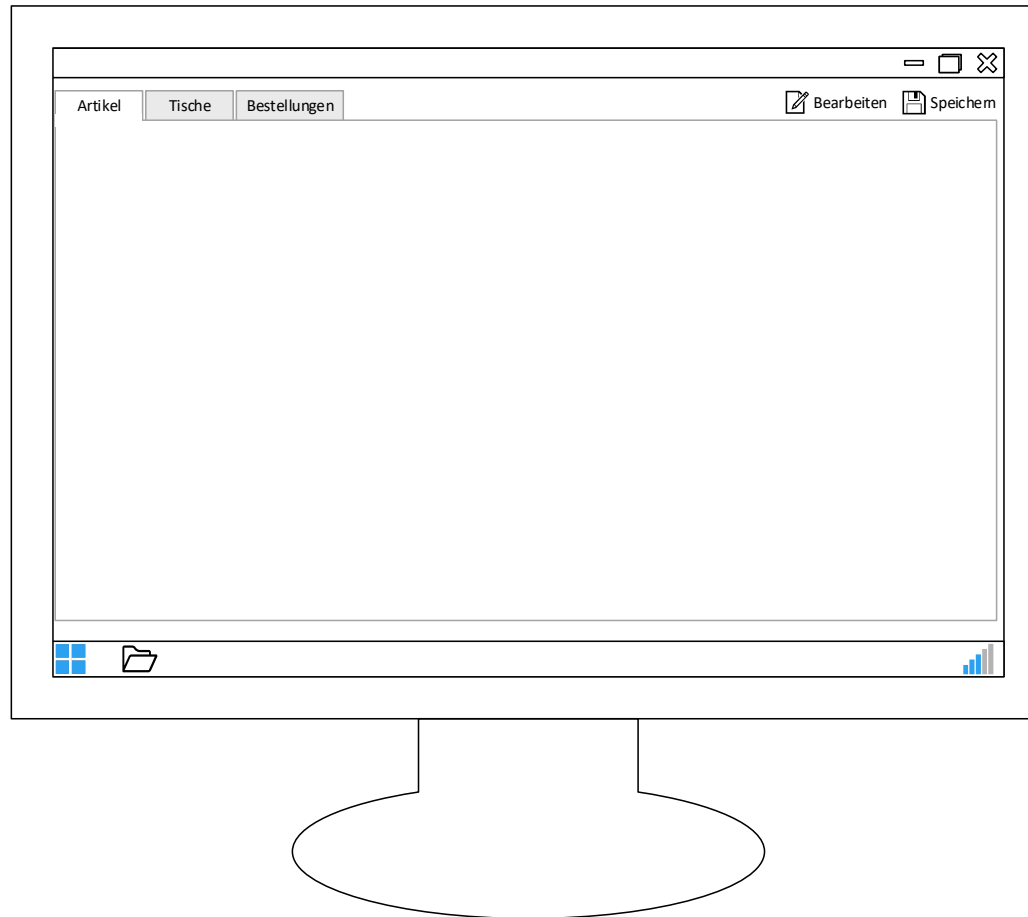


Abbildung 6: Entwurf der Benutzeroberfläche des Kassensystem-Managers

2.8 Nichtfunktionale Anforderungen

Ordnungsmäßigkeit der Buchführung

- Evtl. Ergänzung einer Erstellung einer PDF des Belegs
- Andere Möglichkeit: doppelt ausdrucken zur Archivierung

Sicherheitsanforderungen, z. B. Passwortschutz, Mitlaufen von Protokollen, sichere Übertragung

- Erweiterung mit Login-Daten: Mitarbeiter können sich am Mobiltelefon einloggen
 - Verwendung von Hashcodes für Passwörter
- Evtl. Verwendung von Https statt Http

2.9 Technische Produktumgebung

2.9.1 Software

Datenbank-Rechner

- Betriebssystem: Windows 10
- MySQL-Server, MySQL-Workbench zur Verwaltung
- Java Runtime Environment (JRE)

Smartphone

- Betriebssystem: Android, Version > 5

2.9.2 Hardware

Datenbank-Rechner

- Ausreichend Leistung (CPU, RAM)
- Bon-Drucker
- Schnittstelle zum Bon-Drucker (USB, ...)
- Bildschirm, Tastatur, Maus, optional Touchscreen
- Netzwerkkarte

Smartphone

- Ausreichend Speicherplatz

3 Entwurf

3.1 Klassendesign des Datenbank-Systems

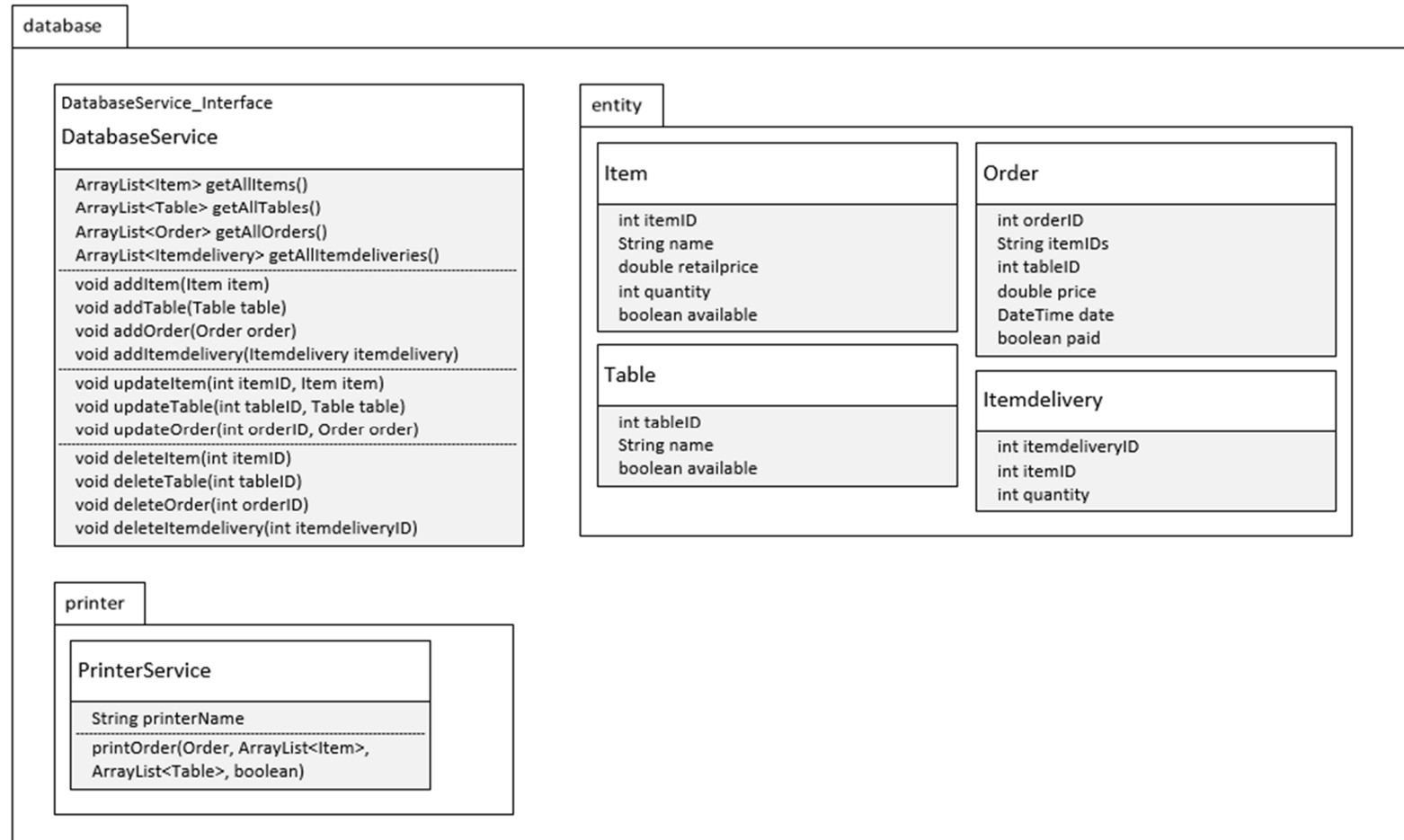


Abbildung 7: Software-Package "database"

Das Datenbank-System ist in zwei Hauptkomponenten aufgeteilt: Der verwaltende Hintergrundprozess und die Benutzeranwendung mit grafischer Oberfläche. Zum Hintergrundprozess gehören mehrere Komponenten: Das Package „database“ (siehe Abbildung 7) und das Package „RestApi“ (siehe Abbildung 8).

Der Database-Service im „database“-Package wird von allen anderen Komponenten verwendet um auf Daten der Datenbank zuzugreifen oder diese zu ändern. Das darin befindliche Package „entity“ beschreibt die Tabellen der Datenbank und dient dazu, neue Datensätze zu erzeugen oder Daten der Datenbank in diesen zu speichern. Das letzte Package „printer“ stellt eine Schnittstelle zur Verfügung, worüber eine Bestellung ausgedruckt werden kann.

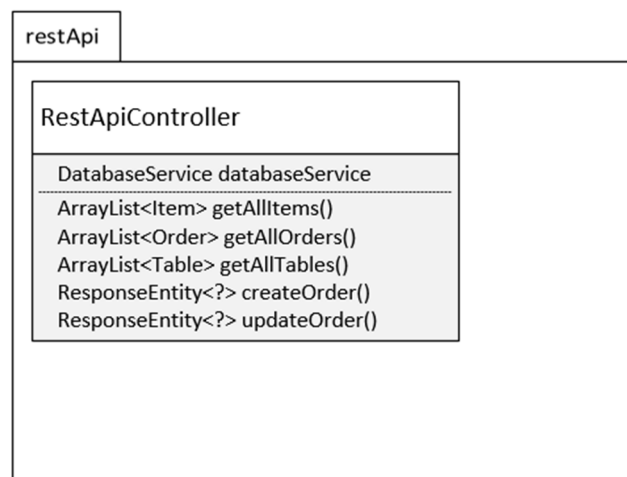


Abbildung 8: Das Software-Package "RestApiController"

Das Package „restApi“ stellt eine Netzwerkschnittstelle zur Verfügung, über die im Netzwerk Anfragen gestellt werden können. Über diese sind Funktionen des Database-Services ansprechbar und somit Daten der Datenbank abruf- und änderbar.

3.2 Klassendesign der Android-Anwendung

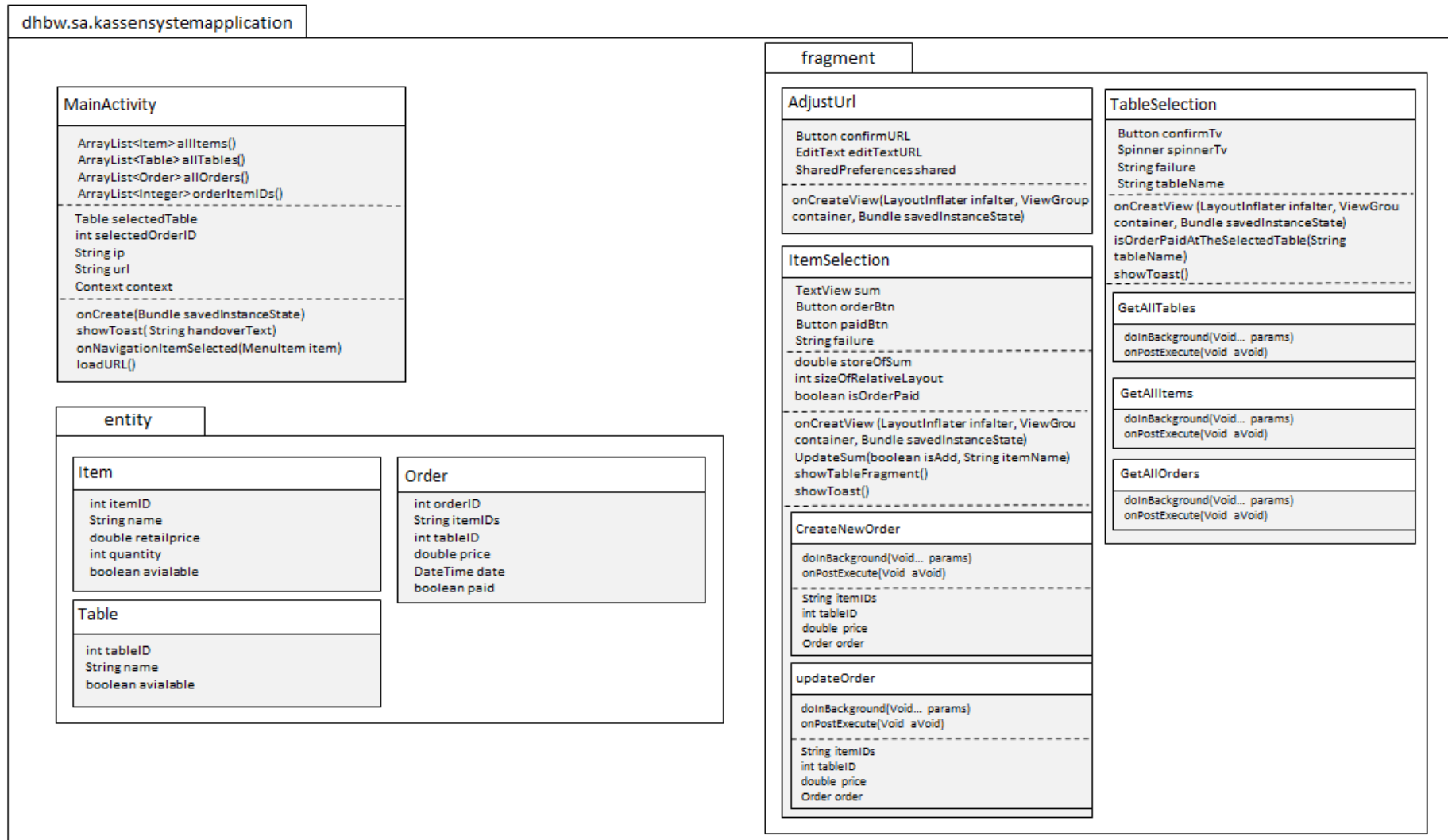


Abbildung 9: Software-Package "Kassensystemapplikation"

Die Android-Anwendung besteht aus drei verschiedenen Komponenten. Alle drei sind zusammengefasst in dem Package „dhbw.sa.kassensystemapplication“. In der Klasse „MainActivity“ werden die Informationen gespeichert, welche die Applikation von der Datenbank erhält.

Die zweite Komponente umfasst die Benutzeranwendung. Diese Komponente spiegelt sich in dem Package „fragment“ wieder. Dort werden alle Eingaben des Benutzers behandelt. In diesem Package befindet sich ebenfalls die Schnittstelle zu dem Datenbank-System und kann so die benötigten Informationen kommunizieren.

Die letzte Komponente besteht aus dem Package „entity“ und umfasst ebenfalls die Tabellen der Datenbank (siehe Kapitel 3.1 Entwurf des Datenbank-Systems). Diese werden in der Applikation für eine korrekte Kommunikation benötigt.

4 Implementierung

4.1 Implementierung des Datenbank-Systems

Im Folgenden sind ausgewählte Klassen des Datenbank-System exemplarisch dokumentiert. Unter dem folgenden Link ist die gesamte Dokumentation abrufbar:

<https://kassensystem.github.io/DatabaseSystem/>

4.1.1 Klasse DatabaseService_Interface

dhbw.sa.kassensystem_rest.database

Interface DatabaseService_Interface

All Known Implementing Classes:

[DatabaseService](#)

public interface DatabaseService_Interface

Der DatabaseService stellt die Schnittstelle zu einer MySQL-Datenbank dar.

Author:
Marvin Mai

Method Summary

Modifier and Type	Method and Description
void	addItem (Item item) Fuegt der Datenbank einen neuen Artikel hinzu.
void	addItemdelivery (Itemdelivery itemdelivery) Fuegt der Datenbank einen neuen Wareneingang hinzu.
void	addOrder (Order order) Fuegt der Datenbank eine neue Bestellung hinzu.
void	addTable (Table table) Fuegt der Datenbank einen neuen Tisch hinzu.
void	connect () Stellt eine Verbindung zur MySQL-Datenbank her.
void	deleteItem (int itemID) Markiert einen Artikel als nicht verfügbar.

void	deleteItemdelivery (int itemdeliveryID) Loescht einen Wareneingang aus der Datenbank.
void	deleteOrder (int orderID) Loescht eine Bestellung aus der Datenbank.
void	deleteTable (int tableID) Markiert einen Tisch als nicht verfuegbar.
void	disconnect () Beendet eine bestehende Verbindung mit einem MySQL-Server.
java.util.ArrayList< Itemdelivery >	getAllItemdeliveries () Fragt die Wareneingaenge der Datenbank ab.
java.util.ArrayList< Item >	getAllItems () Fragt die Artikel der Datenbank ab.
java.util.ArrayList< Order >	getAllOrders () Fragt die Bestellungen der Datenbank ab.
java.util.ArrayList< Table >	getAllTables () Fragt die Tische der Datenbank ab.
Item	getItemById (int itemID) Liefert ein Item in Abhaengigkeit von einer ID.
Order	getOrderByID (int orderID) Liefert eine Bestellung in Abhaengigkeit von einer ID.
Table	getTableById (int tableID) Liefert einen Table in Abhaengigkeit von einer ID.
void	printOrderByID (int orderID) Ausdrucken einer Bestellung in Abhaengigkeit von einer ID;
void	updateItem (int itemID, Item item) Aktualisiert die Daten eines Artikels.
void	updateOrder (int orderID, Order order) Aktualisiert die Daten einer Bestellung.
void	updateTable (int tableID, Table table) Aktualisiert die Daten eines Tisches.

Method Detail

connect

void connect()

Stellt eine Verbindung zur MySQL-Datenbank her.

Throws:

java.lang.IllegalStateException - wenn die Datenbank nicht erreichbar ist.

disconnect

void disconnect()

Beendet eine bestehende Verbindung mit einem MySQL-Server.

getAllItems

java.util.ArrayList<[Item](#)> getAllItems()

Fragt die Artikel der Datenbank ab.

Returns:

Artikel der Datenbank.

getAllTables

java.util.ArrayList<[Table](#)> getAllTables()

Fragt die Tische der Datenbank ab.

Returns:

Tische der Datenbank.

getAllOrders

java.util.ArrayList<[Order](#)> getAllOrders()

Fragt die Bestellungen der Datenbank ab.

Returns:

Bestellungen der Datenbank.

getAllItemdeliveries

java.util.ArrayList<[Itemdelivery](#)> getAllItemdeliveries()

Fragt die Wareneingaenge der Datenbank ab.

Returns:

Wareneingaenge der Datenbank.

getOrderById

[Order](#) getOrderById(int orderID)

Liefert eine Bestellung in Abhängigkeit von einer ID.

Parameters:

orderId - ID der Bestellung.

Returns:

Order mit angegebener ID.

getItemById

[Item](#) getItemById(int itemID)

Liefert ein [Item](#) in Abhängigkeit von einer ID.

Parameters:

itemID - ID des Artikels.

Returns:

Item mit angegebener ID.

getTableById

[Table](#) getTableById(int tableID)

Liefert einen [Table](#) in Abhängigkeit von einer ID.

Parameters:

tableID - ID des Tisches.

Returns:

Table mit angegebener ID.

addItem

void addItem([Item](#) item)

Fügt der Datenbank einen neuen Artikel hinzu.

Parameters:

item - neuer Artikel.

addTable

void addTable([Table](#) table)

Fügt der Datenbank einen neuen Tisch hinzu.

Parameters:

table - neuer Tisch.

addOrder

void addOrder([Order](#) order)

Fügt der Datenbank eine neue Bestellung hinzu.

Parameters:
order - neue Bestellung.

addItemdelivery

void addItemdelivery([Itemdelivery](#) itemdelivery)

Fuegt der Datenbank einen neuen Wareneingang hinzu.

Parameters:
itemdelivery - neuer Wareneingang.

updateItem

void updateItem(int itemID,
[Item](#) item)

Aktualisiert die Daten eines Artikels.

Parameters:
itemID - ID des zu aktualisierenden Artikels.
item - neue Artikeldaten.

updateTable

void updateTable(int tableID,
[Table](#) table)

Aktualisiert die Daten eines Tisches.

Parameters:
tableID - ID des zu aktualisierenden Tisches.
table - neue Tischdaten.

updateOrder

void updateOrder(int orderID,
[Order](#) order)

Aktualisiert die Daten einer Bestellung.

Parameters:
orderID - ID der zu aktualisierenden Bestellung.
order - neue Bestelungsdaten.

deleteItem

void deleteItem(int itemID)

Markiert einen Artikel als nicht verfuegbar. Daten werden nicht geloescht.

Parameters:

itemID - ID des als nicht verfügbare zu markierenden Artikels.

deleteTable

void deleteTable(int tableID)

Markiert einen Tisch als nicht verfügbar. Daten werden nicht gelöscht.

Parameters:

tableID - ID des als nicht verfügbar zu markierenden Tisches.

deleteOrder

void deleteOrder(int orderID)

Löscht eine Bestellung aus der Datenbank.

Parameters:

orderID - ID der zu löschenden Bestellungen.

deleteItemdelivery

void deleteItemdelivery(int itemdeliveryID)

Löscht einen Wareneingang aus der Datenbank.

Parameters:

itemdeliveryID - ID des zu löschenden Wareneingangs.

printOrderByID

void printOrderByID(int orderID)

Ausdrucken einer Bestellung in Abhängigkeit von einer ID.

Parameters:

orderID - ID der auszudruckenden Order.

4.1.2 Klasse RestApiController

Class RestApiController

java.lang.Object

dhbw.sa.kassensystem_rest.restApi.controller.RestApiController

```
@RestController
@ComponentScan(value="dhbw.sa.kassensystem_database.database")
@RequestMapping(value="/api")
public class RestApiController
extends java.lang.Object
```

Der RestApiController stellt einen Server dar, über den Funktionen des Database-Services angesprochen werden können. Diese sind über das Netzwerk verfügbar. Dabei müssen die entsprechenden Pfade beachtet werden. Der Root-Pfad ist ".../api".

Author:
 Marvin Mai

Constructor Summary

Constructors

Constructor and Description

[RestApiController\(\)](#)

Method Summary

Modifier and Type	Method and Description
org.springframework- work.http.ResponseEntity<?>	createOrder(Order order) Erstellt eine neue Bestellung in der MySQL-Datenbank.
java.util.ArrayList< Item >	getAllItems() Durch das Ansprechen des Pfades "..."
java.util.ArrayList< Order >	getAllOrders() Durch das Ansprechen des Pfades "..."
java.util.ArrayList< Table >	getAllTables() Durch das Ansprechen des Pfades "..."
org.springframework- work.http.ResponseEntity<?>	updateOrder(int orderID, Order order) Updatet eine bereits existierende Bestellung in der Datenbank.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

RestController

```
public RestApiController()
```

Method Detail

getAllItems

```
@RequestMapping(value="/items")  
public java.util.ArrayList<Item> getAllItems()
```

Durch das Ansprechen des Pfades ".../api/items" können die Artikel der Datenbank abgefragt werden.

Returns:

Liste aller Artikel der Datenbank.

getAllOrders

```
@RequestMapping(value="/orders")  
public java.util.ArrayList<Order> getAllOrders()
```

Durch das Ansprechen des Pfades ".../api/orders" können die Bestellungen der Datenbank abgefragt werden.

Returns:

Liste aller Bestellungen der Datenbank.

getAllTables

```
@RequestMapping(value="/tables")  
public java.util.ArrayList<Table> getAllTables()
```

Durch das Ansprechen des Pfades ".../api/tables" können die Tische der Datenbank abgefragt werden.

Returns:

Liste aller Tische der Datenbank.

createOrder

```
@RequestMapping(value="/order/",  
                method=POST)  
public org.springframework.http.ResponseEntity<?> createOrder(@RequestBody  
                                                             Order order)
```

Erstellt eine neue Bestellung in der MySQL-Datenbank.

Parameters:

order - neu zu erstellende Bestellung.

Returns:

ResponseEntity, das Erstellen entweder bestätigt oder eine Fehlermeldung liefert.

updateOrder

```
@RequestMapping(value="/order/{orderId}",  
                method=PUT)  
public org.springframework.http.ResponseEntity<?> updateOrder(@PathVariable(value="orderId")  
                                                             int orderId,  
                                                             @Request-  
Body  
                                                             Order order)
```

Updatet eine bereits existierende Bestellung in der Datenbank.

Parameters:

orderId - Zu aktualisierende Bestellung.

order - Bestellung, deren Daten anstelle der existierenden Bestellung gespeichert werden sollen.

Returns:

ResponseEntity, das Updaten entweder bestätigt oder eine Fehlermeldung liefert.

4.2 Implementierung der Android-Anwendung

Im Folgenden sind zwei ausgewählte Klassen der Android-Anwendung exemplarisch dokumentiert. Diese exemplarische Dokumentation ist stark gekürzt. Die gesamte Dokumentation ist unter folgendem Link abrufbar: <https://nunay.github.io/Kassensytem-AndroidApplikation/JavaDoc/>

4.2.1 Klasse TableSelect

```
public class TableSelection  
extends android.support.v4.app.Fragment
```

In dieser Klasse wird der Startbildschirm der Applikation erstellt. Dieser wird ebenfalls aufgerufen, wenn angefangen wird eine Bestellung aufzugeben.

Author:
Daniel Schifano

Method Detail

- onCreateView
- public android.view.View onCreateView(android.view.LayoutInflater inflater,
android.view.ViewGroup container, android.os.Bundle savedInstanceState)

Diese Methode wird aufgerufen wenn das Fragment erstellt wird. Dabei wird der Befehl gegeben, dass alle Artikel, Tische und Bestellungen vom Server angefordert werden. Für "Bestellung aufgeben" wird hier der Startbildschirm initialisiert.

Overrides:

onCreateView in class android.support.v4.app.Fragment

Parameters:

inflater - Instantiiert ein XML-Layout in ein passendes View Objekt

container - Erlaubt den Zugriff auf container Eigenschaften

savedInstanceState - Gibt an in welchem Abschnitt des Lebenszyklus die App sich befindet. Ob sie z.B. geschlossen wurde oder gestartet wurde.

Returns:

View die dargestellt werden soll.

4.2.2 Klasse MainActivity

public class MainActivity

extends android.support.v7.app.AppCompatActivity

implements android.support.design.widget.NavigationView.OnNavigationItemSelectedListener

Diese Klasse dient als container (Hintergrund) für alle anderen Klassen. Zusätzlich werden in dieser Klasse alle Informationen die von der Datenbank empfangen werden, gespeichert.

- Nested Class Summary
 - Nested classes/interfaces inherited from class android.support.v4.app.SupportActivity

android.support.v4.app.SupportActivity.ExtraData

- Field Summary

Fields

Modifier and Type	Field and Description
static java.util.ArrayList< Item >	allItems Liste die alle Artikel der Datenbank beinhaltet.
static java.util.ArrayList< Order >	allOrders Liste die alle Bestellungen der Datenbank beinhaltet.
static java.util.ArrayList< Table >	allTables Liste die alle Tische der Datenbank beinhaltet.
static android.content.Context	context Der Hintergrund für alle weiteren Klassen wird hier gespeichert.
static java.lang.String	ip Speichert die IP-Adresse des Servers.
static java.util.ArrayList<java.lang.Integer>	orderItemIDs

Liste die alle ArtikelIDs einer Bestellung der Datenbank beinhaltet.

static int

[selectedOrderID](#)

Zwischenspeicher der ausgewählten BestellungsID.

static [Table](#)

[selectedTable](#)

Der Tisch an dem die Bestellung stattfindet.

static java.lang.String

[url](#)

Speichert die URL des Servers.

Modifier and Type

Method and Description

boolean

[loadURL\(\)](#)

In dieser Methode werden die IP-Adresse und die URL geladen.

protected void

[onCreate](#) (android.os.Bundle savedInstanceState)

Diese Methode wird aufgerufen wenn die App gestartet wird.

boolean

[onNavigationItemSelected](#)(android.view.MenuItem item)

Mit dieser Funktion werden die verschiedenen Klassen (Fragments) die im Navigation-Drawer auswählbar sind aufgerufen.

void

[showToast](#)(java.lang.String handoverText)

Methode, die den übergebenen Text auf dem Smartphone darstellt.

- Method Detail

- onCreate

`protected void onCreate(android.os.Bundle savedInstanceState)`

Diese Methode wird aufgerufen wenn die App gestartet wird. Dabei wird das Layout(Hintergrund) für alle weiteren Klassen initialisiert.

Overrides:

onCreate in `class android.support.v7.app.AppCompatActivity`

Parameters:

`savedInstanceState` - Gibt an in welchem Abschnitt des Lebenszyklus die App sich befindet. Ob sie z.B. geschlossen wurde oder gestartet wurde.

- **showToast**

`public void showToast(java.lang.String handoverText)`

Methode, die den übergebenen Text auf dem Smartphone darstellt.

Parameters:

`handoverText` - Der Text welcher dargestellt werden soll.

- **onNavigationItemSelectedListener**

`public boolean onNavigationItemSelectedListener(android.view.MenuItem item)`

Mit dieser Funktion werden die verschiedenen Klassen (Fragments) die im Navigation-Drawer auswählbar sind aufgerufen.

Specified by:

onNavigationItemSelectedListener in interface `android.support.design.widget.NavigationView.OnNavigationItemSelectedListenerListener`

Parameters:

`item` - Das item das in dem Navigation-Drawer ausgewählt/angeklickt wurde.

Returns:

`true`, wenn die Methode ohne Fehler abgearbeitet werden konnte.

- **loadURL**

`public boolean loadURL()`

In dieser Methode werden die IP-Adresse und die URL geladen.

Hierfür wird in der Klasse AdjustUrl die IP-Adresse und die URL über den Lebenszyklus der Applikation gespeichert.

Returns:

true, wenn bereits ein URL gespeichert wurde. False wenn noch kein URL gespeichert wurde.

5 Test

5.1 Test des Datenbank-Systems

Im Folgenden sind die Tests der öffentlichen Methoden des Kassensystem-Managers dokumentiert. Die gesamte Testdokumentation ist im Anhang abrufbar.

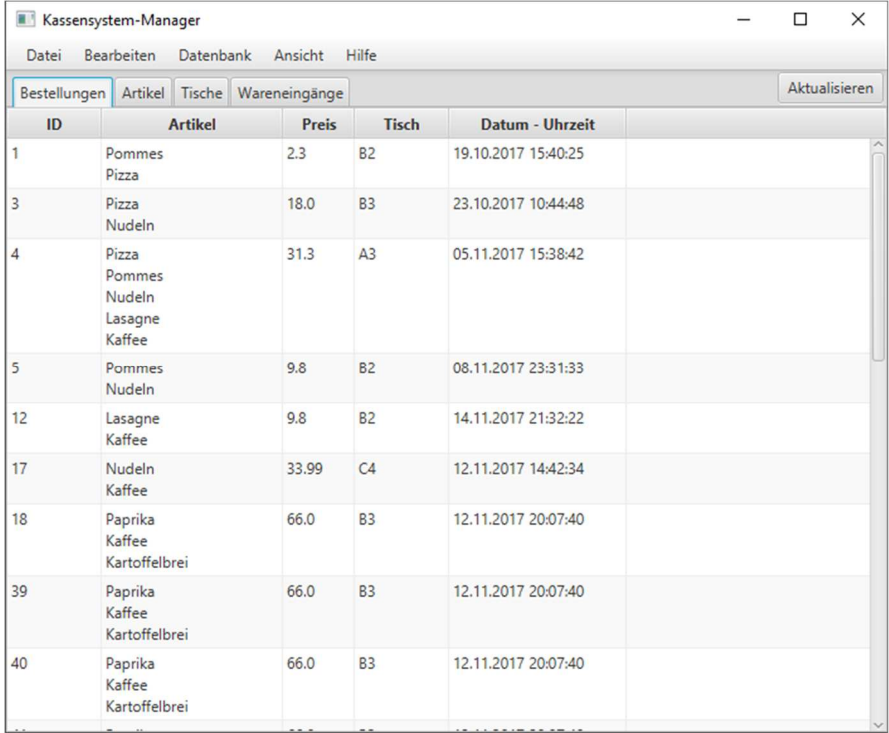
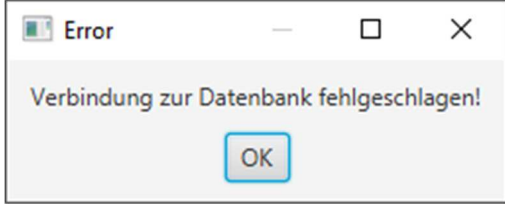
Es wird jeweils der Anwendungsfall und die getestete Methode beschrieben, außerdem Normalablauf und Sonderfälle, bspw. einem Laufzeitfehler wie Verbindungsprobleme oder falsche Eingaben. Anschließend wird das zu erwartende und das tatsächliche Testergebnis für den Normalablauf und den Sonderfall dokumentiert. In einigen Fällen wurden die Testergebnisse gekürzt.

5.1.1 Benutzeranwendung Kassensystem-Manager

5.1.1.1 Abrufen von Datenbankinhalten

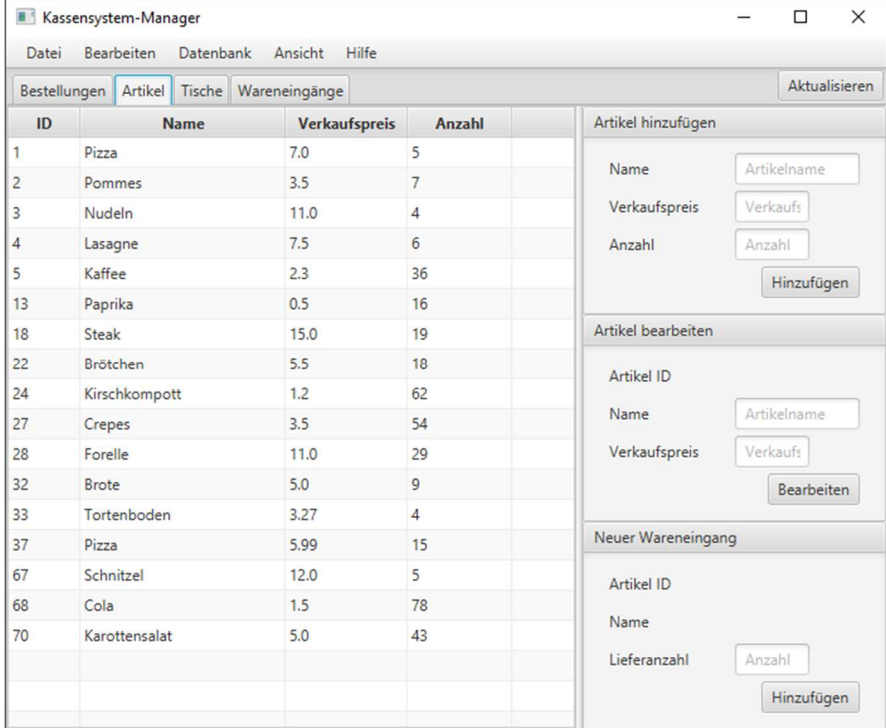
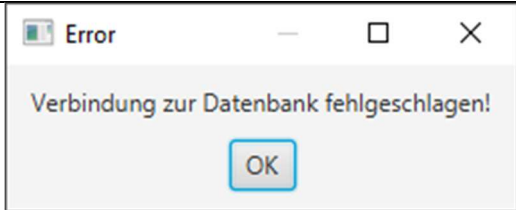
Abrufen der Bestellungen

Anwendungsfall	Einsehen aller Bestellungen in dem Kassensystem-Manager (AW 1)
Verwendete Methode	<code>void refreshOrderData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle nicht bezahlten Bestellungen in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller nicht bezahlten Bestellungen.

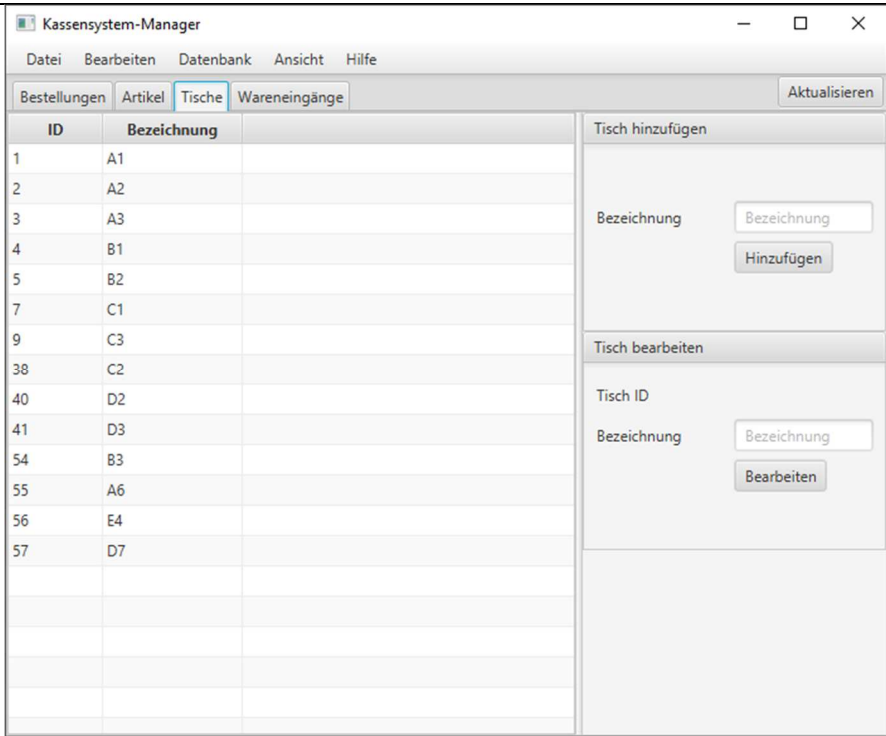
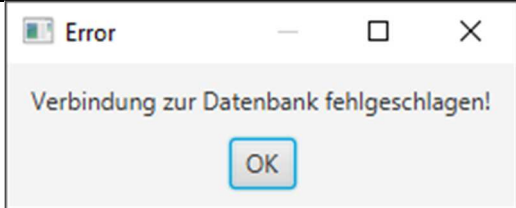
<p>Tatsächliches Testergebnis</p>	
<p>Sonderfall</p>	<p>Es besteht keine Verbindung zum MySQL-Server.</p>
<p>Erwartetes Testergebnis</p>	<p>Es wird eine Fehlermeldung angezeigt.</p>
<p>Tatsächliches Testergebnis</p>	
<p>Test bestanden</p>	<p>Normalablauf: Ja Sonderfall: Ja</p>

Abrufen der Artikel

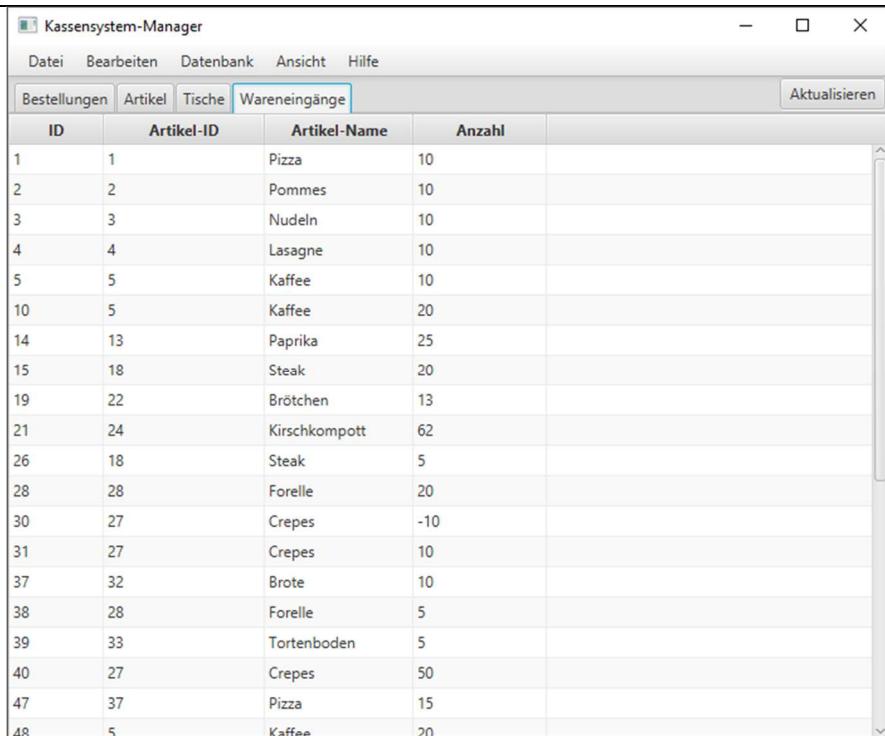
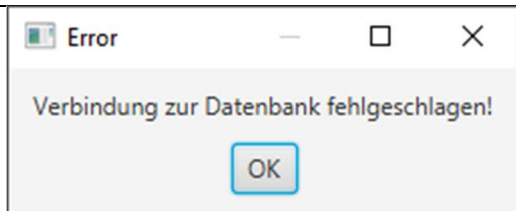
<p>Anwendungsfall</p>	<p>Einsehen aller Artikel im Kassensystem-Manager (AW 3)</p>
<p>Verwendete Methode</p>	<p><code>void refreshItemData()</code></p>
<p>Normalablauf</p>	<p>Nach dem Öffnen der Anwendung werden alle verfügbaren Artikel in einem Tab dargestellt.</p>

Erwartetes Testergebnis	Eine tabellarische Darstellung aller Artikel.
Tatsächliches Testergebnis	 <p>The screenshot shows the 'Kassensystem-Manager' application. It has a menu bar with 'Datei', 'Bearbeiten', 'Datenbank', 'Ansicht', and 'Hilfe'. Below the menu is a tabbed interface with 'Bestellungen', 'Artikel' (selected), 'Tische', and 'Wareneingänge'. The 'Artikel' tab displays a table with columns 'ID', 'Name', 'Verkaufspreis', and 'Anzahl'. The table contains 20 rows of data. To the right of the table are three panels: 'Artikel hinzufügen' (with fields for Name, Verkaufspreis, and Anzahl), 'Artikel bearbeiten' (with fields for Artikel ID, Name, and Verkaufspreis), and 'Neuer Wareneingang' (with fields for Artikel ID, Name, and Lieferanzahl). Each panel has a 'Hinzufügen' or 'Bearbeiten' button.</p>
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	 <p>The screenshot shows an error dialog box titled 'Error'. The message inside reads 'Verbindung zur Datenbank fehlgeschlagen!' (Database connection failed!). There is an 'OK' button at the bottom center of the dialog.</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Abrufen der Tische

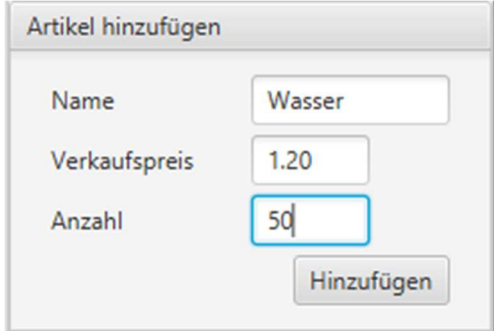
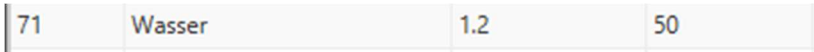
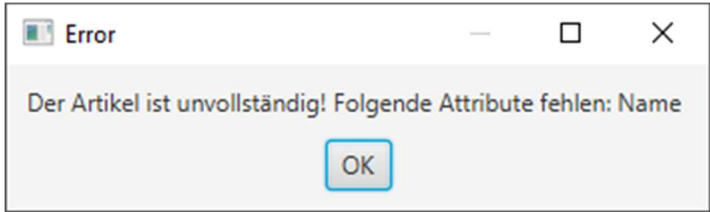
Anwendungsfall	Einsehen aller Tische im Kassensystem-Manager (AW 7)
Verwendete Methode	<code>void refreshTableData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle verfügbaren Tische in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller Tische.
Tatsächliches Testergebnis	
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

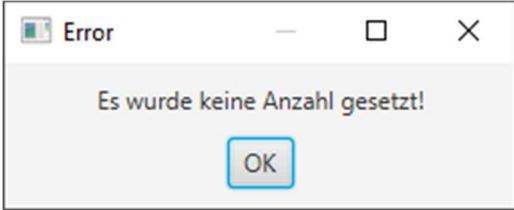
Abrufen der Wareneingänge

Anwendungsfall	Einsehen aller Wareneingänge in dem Kassensystem-Manager (AW 12)
Verwendete Methode	<code>void refreshItemdeliveryData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle Wareneingänge in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller Wareneingänge.
Tatsächliches Testergebnis	
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

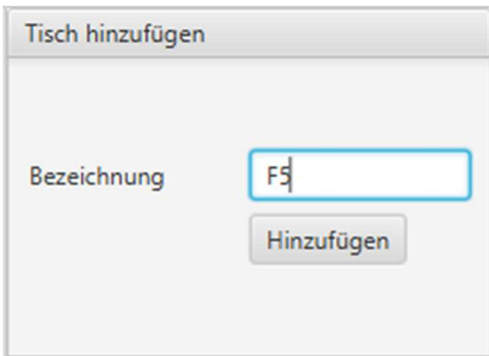

5.1.1.2 Hinzufügen von Datenbankinhalten

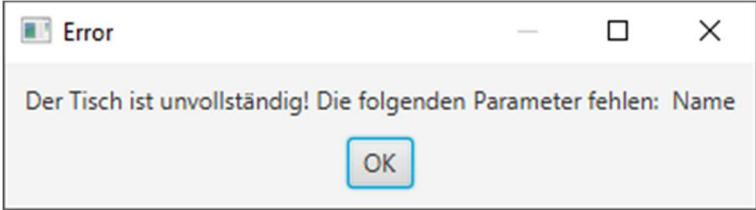
Hinzufügen von neuen Artikeln

Anwendungsfall	Hinzufügen eines neuen Artikels, der neu in das Sortiment/die Speisekarte aufgenommen wurde (AW 5)
Verwendete Methode	<code>void addItem(ActionEvent actionEvent)</code>
Normalablauf	In die Felder werden die Daten des neuen Artikels eingegeben. Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.
Erwartetes Testergebnis	Ein neuer Artikel mit den folgenden Daten: 
Tatsächliches Testergebnis	
Sonderfall	<ol style="list-style-type: none"> 1. Es wurde kein Name angegeben. 2. Es wurde keine Anzahl angegeben.
Erwartetes Testergebnis	<ol style="list-style-type: none"> 1. Anzeige einer Fehlermeldung mit einem fehlenden Namen. 2. Anzeige einer Fehlermeldung mit einer fehlenden Anzahl.
Tatsächliches Testergebnis	<ol style="list-style-type: none"> 1.  2.



	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

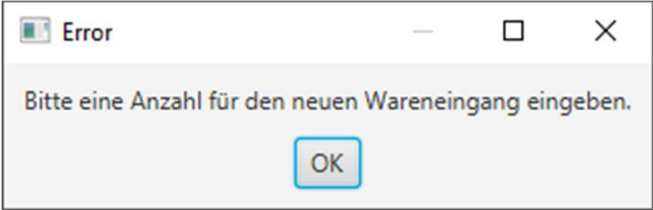
Hinzufügen von neuen Tischen

Anwendungs- fall	Hinzufügen eines neuen Tisches, der neu im Geschäftsbereich eingerichtet wird (AW 9)
Verwendete Methode	<code>void addTable(ActionEvent actionEvent)</code>
Normalablauf	Es werden in das Feld die Daten des neuen Tisches eingegeben. Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.
Erwartetes Testergebnis	Ein Tisch mit den folgenden Daten: 
Tatsächliches Testergebnis	
Sonderfall	Es wurde kein Name angegeben.
Erwartetes Testergebnis	Fehlermeldung mit der Meldung eines fehlenden Namens.

Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

Hinzufügen von neuen Wareneingängen

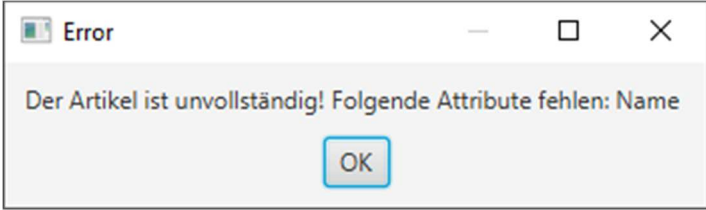
Anwendungsfall	Hinzufügen eines neuen Wareneingangs. Das wird während des Befüllens des Lagers gemacht (AW 13)
Verwendete Methode	<code>void addItemDelivery(ActionEvent actionEvent)</code>
Normalablauf	Ein Artikel wird angeklickt und anschließend die neue Anzahl eingegeben. Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.
Erwartetes Testergebnis	Ein neuer Wareneingang mit den folgenden Daten: 
Tatsächliches Testergebnis	
Sonderfall	Es wurde keine Anzahl eingegeben.
Erwartetes Testergebnis	Fehlermeldung mit der Meldung einer fehlenden Anzahl.

<p>Tatsächliches Testergebnis</p>	
<p>Test bestanden</p>	<p>Normalablauf: Ja Sonderfall: Ja</p>

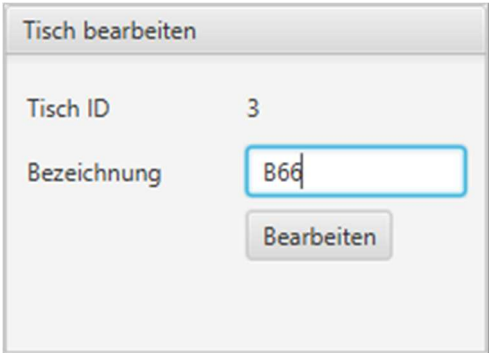
5.1.1.3 Bearbeiten von Datenbankinhalten

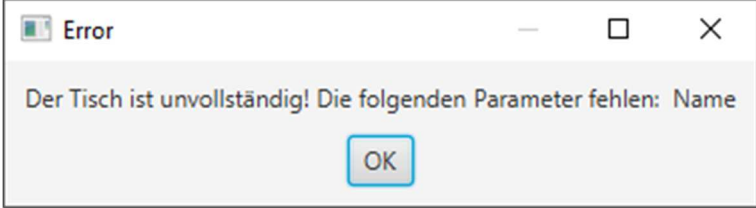
Bearbeiten von Artikeln

Anwendungsfall	Ändern der Daten eines Artikels, wie bspw. Preisänderung (AW 6)				
Verwendete Methode	<code>public void editItem(ActionEvent actionEvent)</code>				
Normalablauf	Ein Artikel wird angeklickt. Im entsprechenden Feld zum Bearbeiten des Artikels erscheinen die aktuellen Daten. Diese können bearbeitet werden. Wenn der „Bearbeiten“-Button gedrückt wird, wird der bisherige Artikel als nicht verfügbar markiert und ein neuer Datenbankeintrag mit den bearbeiteten Daten erzeugt. In der tabellarischen Übersicht wird der bearbeitete Artikel angezeigt.				
Erwartetes Testergebnis	<p>Der folgende Artikel soll aktualisiert werden:</p> <table><tr><td>3</td><td>Burger</td><td>4.5</td><td>57</td></tr></table> <p>Dieser Artikel soll mit den folgenden Daten aktualisiert werden:</p> <div><div>Artikel bearbeiten</div><div><div>Artikel ID3</div><div>NameVeggieBurger</div><div>Verkaufspreis6.50</div><div>Bearbeiten</div></div></div>	3	Burger	4.5	57
3	Burger	4.5	57		
Tatsächliches Testergebnis	<p>In den Daten ist nun der folgende Artikel zu finden:</p> <table><tr><td>6</td><td>VeggieBurger</td><td>6.5</td><td>57</td></tr></table>	6	VeggieBurger	6.5	57
6	VeggieBurger	6.5	57		
Sonderfall	Es wird kein Name angegeben.				
Erwartetes Testergebnis	Eine Fehlermeldung, die einen fehlenden Namen anmerkt.				

Tatsächliches Testergebnis	
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>


Bearbeiten von Tischen

Anwendungsfall	Ändern der Bezeichnung eines Tisches (AW 10)		
Verwendete Methode	<code>public void editTable(ActionEvent actionEvent)</code>		
Normalablauf	Ein Tisch wird angeklickt. Im entsprechenden Feld zum Bearbeiten des Tisches erscheinen die aktuellen Daten. Diese können bearbeitet werden. Wenn der „Bearbeiten“-Button gedrückt wird, wird der bisherige Tisch als nicht verfügbar markiert und ein neuer Datenbankeintrag mit den bearbeiteten Daten erzeugt. In der tabellarischen Übersicht wird der bearbeitete Tisch angezeigt.		
Erwartetes Testergebnis	<p>Der folgende Tisch soll aktualisiert werden:</p> <table border="1" data-bbox="450 1355 831 1411"> <tr> <td>3</td> <td>A3</td> </tr> </table> <p>Dieser Tisch soll mit den folgenden Daten bearbeitet werden:</p> 	3	A3
3	A3		
Tatsächliches Testergebnis	In der tabellarischen Übersicht ist nun der folgende Tisch zu finden:		

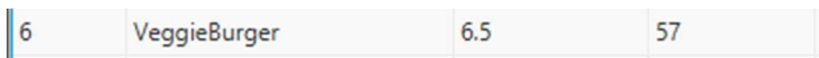
	7 B66
Sonderfall	Es wurde kein Name angegeben.
Erwartetes Testergebnis	Eine Fehlermeldung über den fehlenden Namen.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

5.1.1.4 Löschen von Datenbankinhalten


Löschen von Bestellungen

Anwendungsfall	Löschen einer fehlerhaften oder überschüssigen Bestellung (AW 2)
Verwendete Methode	<code>public void deleteOrder(ActionEvent actionEvent)</code>
Normalablauf	Eine Bestellung wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag der Bestellung verschwindet aus der tabellarischen Übersicht und ist nicht mehr in der Datenbank zu finden.
Erwartetes Testergebnis	Der Eintrag der Bestellung wird aus der tabellarischen Übersicht entfernt. Die folgende Bestellung soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Artikeln

Anwendungsfall	Löschen eines Artikels der aus dem Sortiment genommen wurde (AW 4)
Verwendete Methode	<code>public void deleteItem(ActionEvent actionEvent)</code>
Normalablauf	Ein Artikel wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag der Bestellung verschwindet aus der tabellarischen Übersicht und in der Datenbank wird der Artikel als nicht verfügbar markiert.
Erwartetes Testergebnis	Der folgende Artikel soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Tischen

Anwendungsfall	Löschen eines Tisches, der von der Verkaufsfläche entfernt wurde (AW 8)
Verwendete Methode	<code>public void deleteTable(ActionEvent actionEvent)</code>
Normalablauf	Ein Tisch wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag des Tisches verschwindet aus der tabellarischen Übersicht und in der Datenbank wird der Tisch als nicht verfügbar markiert.
Erwartetes Testergebnis	Der folgende Tisch soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Wareneingängen

Anwendungsfall	Ein Wareneingang soll gelöscht werden der bspw. fälschlicherweise angelegt wurde (AW 14)
Verwendete Methode	<code>public void deleteItemdelivery(ActionEvent actionEvent)</code>
Normalablauf	Ein Wareneingang wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag des Wareneingangs verschwindet aus der tabellarischen Übersicht und ist nicht mehr in der Datenbank zu finden.

Erwartetes Testergebnis	<p>Der folgende Wareneingang soll gelöscht werden:</p> <table><tr><td>9</td><td>5</td><td>Cola</td><td>50</td></tr></table> <p>In der tabellarischen Übersicht der Artikel wird die Anzahl des entsprechenden Artikels reduziert:</p> <table><tr><td>5</td><td>Cola</td><td>2.5</td><td>212</td></tr></table>	9	5	Cola	50	5	Cola	2.5	212
9	5	Cola	50						
5	Cola	2.5	212						
Tatsächliches Testergebnis	<p>Der Eintrag existiert nicht mehr in der Anwendung. Die Anzahl des Artikels in der Artikel-Übersicht wurde aktualisiert:</p> <table><tr><td>5</td><td>Cola</td><td>2.5</td><td>162</td></tr></table>	5	Cola	2.5	162				
5	Cola	2.5	162						
Sonderfall	keiner								
Erwartetes Testergebnis	--								
Tatsächliches Testergebnis	--								
Test bestanden	Normalablauf: Ja Sonderfall: --								

5.1.1.5 Ausdrucken einer Bestellung

Anwendungsfall	Nachträgliches Ausdrucken eines Belegs, nachdem der ursprüngliche Beleg verloren gegangen oder zerstört bzw. verschmutzt wurde (AW 12)					
Verwendete Methode	<code>public void printOrder(ActionEvent actionEvent)</code>					
Normalablauf	Über einen Rechtsklick wird eine Bestellung ausgewählt und der Menüeintrag zum Ausdrucken angeklickt. Über den Bon-Drucker wird der Kundenbeleg ausgedruckt.					
Erwartetes Testergebnis	<div>Der Bon-Drucker druckt einen Kundenbeleg mit den folgenden Bestelldaten:</div> <table><tr><td>7</td><td>Brot Cola Cola Burger</td><td>14.5</td><td>A1</td><td>03.12.2017 19:51:12</td></tr></table>	7	Brot Cola Cola Burger	14.5	A1	03.12.2017 19:51:12
7	Brot Cola Cola Burger	14.5	A1	03.12.2017 19:51:12		
Tatsächliches Testergebnis	<div>Ein Ausdruck wurde ausgegeben:</div> <div>-----Kundenbeleg----- Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655 Ihre Bestellung: Brot 5,00 EUR Cola 2,50 EUR Cola 2,50 EUR Burger 4,50 EUR Summe 14,50 EUR inkl. MWST 19% 2,89 EUR Sie saßen an Tisch A1. Vielen Dank für Ihren Besuch! 03.12.2017 19:51:12</div>					
Sonderfall	Der Drucker ist nicht angeschlossen oder abgeschaltet.					
Erwartetes Testergebnis	Der Beleg wird ausgedruckt, sobald der Drucker erreichbar ist.					
Tatsächliches Testergebnis	Nach dem Abschalten und wieder Anschalten des Drucker wird der Beleg wie erwartet ausgedruckt.					

Test bestanden	Normalablauf: Ja Sonderfall: Ja
-------------------	------------------------------------

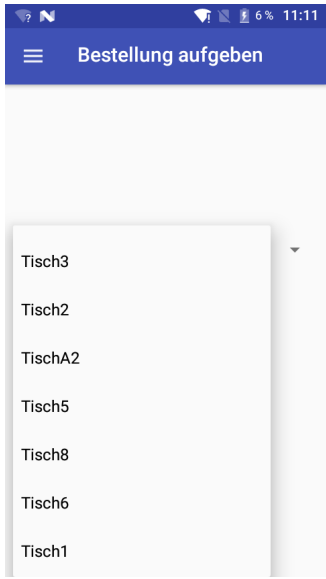
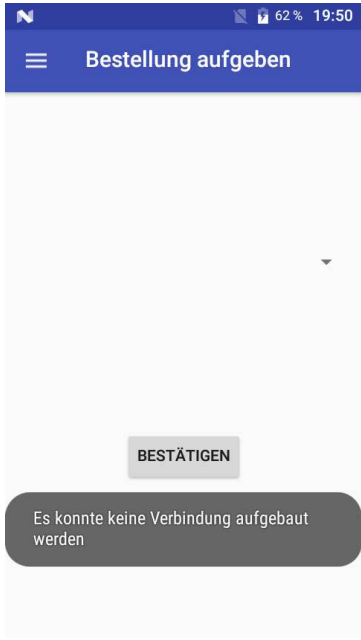
5.2 Test der Android-Anwendung

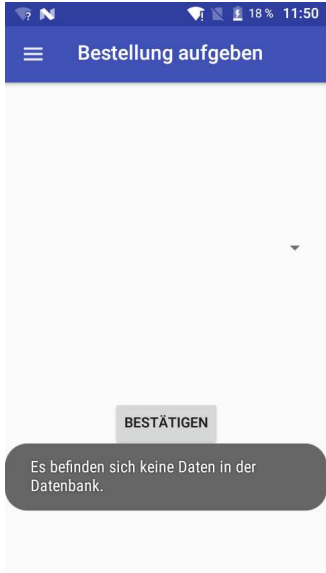
Im Folgenden sind die Tests der Android-Anwendung dokumentiert.

Es wird jeweils der Anwendungsfall und die getestete Methode beschrieben, außerdem Normalablauf und Sonderfälle, bspw. einem Laufzeitfehler wie Verbindungsprobleme oder falsche Eingaben. Anschließend wird das zu erwartende und das tatsächliche Testergebnis für den Normalablauf und den Sonderfall dokumentiert. In einigen Fällen wurden die Testergebnisse gekürzt.

Einsehen der Tische



Anwendungsfall	Einsehen aller verfügbaren Tische in der Android-Anwendung (AW 15)																																										
Verwendete Methode	doInBackground() onPostExecute() (Klasse: "GetAllTables")																																										
Normalablauf	Wird „Bestellung aufgeben“ im Navigation-Drawer ausgewählt, sollen in einem Dropdown-Menü alle verfügbaren Tische der Datenbank dargestellt werden.																																										
Erwartetes Testergebnis	<p>In dem Dropdown-Menü werden die verfügbaren Tische der Datenbank angezeigt. Folgende Tische, die unter dem Register „available“ mit einer 1 markiert sind, werden angezeigt:</p> <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>1</td><td>Tisch3</td><td>1</td></tr><tr><td>2</td><td>Tisch2</td><td>1</td></tr><tr><td>3</td><td>TischA2</td><td>1</td></tr><tr><td>4</td><td>asdf</td><td>0</td></tr><tr><td>5</td><td>Tisch5</td><td>1</td></tr><tr><td>6</td><td>hhahk</td><td>0</td></tr><tr><td>7</td><td>Tisch8</td><td>1</td></tr><tr><td>8</td><td>Tisch6</td><td>1</td></tr><tr><td>9</td><td>qwerz</td><td>0</td></tr><tr><td>10</td><td>asdfahik</td><td>0</td></tr><tr><td>11</td><td>asdsfas...</td><td>0</td></tr><tr><td>12</td><td>Tisch1</td><td>1</td></tr><tr><td>NULL</td><td>NULL</td><td>NULL</td></tr></table>	tableID	name	available	1	Tisch3	1	2	Tisch2	1	3	TischA2	1	4	asdf	0	5	Tisch5	1	6	hhahk	0	7	Tisch8	1	8	Tisch6	1	9	qwerz	0	10	asdfahik	0	11	asdsfas...	0	12	Tisch1	1	NULL	NULL	NULL
tableID	name	available																																									
1	Tisch3	1																																									
2	Tisch2	1																																									
3	TischA2	1																																									
4	asdf	0																																									
5	Tisch5	1																																									
6	hhahk	0																																									
7	Tisch8	1																																									
8	Tisch6	1																																									
9	qwerz	0																																									
10	asdfahik	0																																									
11	asdsfas...	0																																									
12	Tisch1	1																																									
NULL	NULL	NULL																																									

<p>Tatsächliches Testergebnis</p>	<p>Das Dropdown-Menü wurde wie folgt dargestellt:</p> 
<p>Sonderfall 1</p>	<p>Es kann keine Verbindung zur Datenbank aufgebaut werden, beziehungsweise der Server wurde nicht gestartet.</p>
<p>Erwartetes Testergebnis</p>	<p>Das Dropdown-Menü bleibt leer und es wird eine Fehlermeldung angezeigt, dass eine Verbindung zum Server nicht möglich ist.</p>
<p>Tatsächliches Testergebnis</p>	<p>Das Smartphone gibt folgende Fehlermeldung aus:</p> 
<p>Sonderfall 2</p>	<p>Es kann eine Verbindung aufgebaut werden. In der Datenbank befinden sich jedoch keine Tische.</p>
<p>Erwartetes Testergebnis</p>	<p>Das Dropdown-Menü bleibt leer und es wird eine „Fehlermeldung“ angezeigt, dass keine Tische verfügbar sind.</p>

Tatsächliches Testergebnis	<p>Das Smartphone gibt folgende Fehlermeldung aus:</p> 
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall 1: Ja</p> <p>Sonderfall 2: Ja</p>


Auswählen der Tische


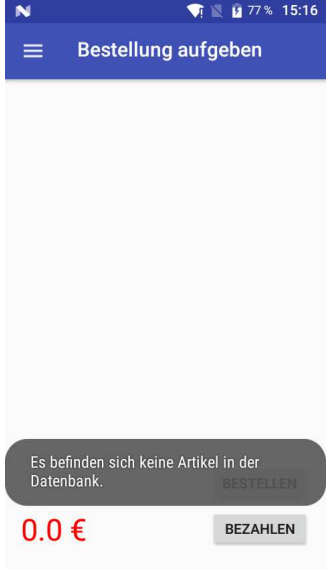
Anwendungsfall	Auswählen der in dem Dropdown-Menü angezeigten Tische (AW 16)
Verwendete Methode	<i>onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)</i> (Klasse: "GetAllTables")
Normalablauf	Der Anwender wählt aus dem Dropdown-Menü einen angezeigten Tisch aus und bestätigt diesen mit dem Button „BESTÄTIGEN“.

	
Erwartetes Testergebnis	Der Tisch wird gespeichert, damit er der Bestellung hinzugefügt werden kann und auf dem Kundenbeleg dargestellt werden kann. Das nächste Fragment wird dargestellt. In diesem Fragment werden alle verfügbaren Artikel angezeigt.
Tatsächliches Testergebnis	<p>Der Tisch wird gespeichert.</p> <p>Der Bildschirm des Smartphones wird folgendermaßen dargestellt:</p> 
Sonderfall	Es wurde kein Tisch ausgewählt und der „BESTÄTIGEN“-Button wird geklickt.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt, dass ein Tisch ausgewählt werden soll.
Tatsächliches	Das Smartphone gibt folgende Fehlermeldung aus:

Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

Artikel Einsehen

Anwendungsfall	Einsehen der Artikel, die in der Datenbank als verfügbar angelegt sind (AW 17)
Verwendete Methode	<i>onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)</i> (Klasse: "ItemSelect")
Normalablauf	Alle verfügbaren Artikel der Datenbank werden dargestellt.
Erwartetes Testergebnis	Alle Artikel werden dargestellt, inklusive Preis und Menge.
Tatsächliches Testergebnis	<p>Der Bildschirm des Smartphones wird folgendermaßen dargestellt:</p> 
Sonderfall 1	Die Verbindung zum Server wird getrennt.
Erwartetes Testergebnis	Die Artikel werden weiterhin angezeigt.
Tatsächliches Testergebnis	Der Bildschirm des Smartphones wird folgendermaßen dargestellt:

	
Sonderfall 2	Die Datenbank enthält keine verfügbaren Artikel die auf dem Smartphone angezeigt werden.
Erwartetes Testergebnis	Es werden keine Artikel angezeigt. Eine „Fehlermeldung“ gibt an, dass keine Artikel in der Datenbank verfügbar sind.
Tatsächliches Testergebnis	<p>Das Smartphone gibt folgende Fehlermeldung aus:</p> 
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall 1: Ja</p> <p>Sonderfall 2: Ja</p>

Artikel einer Bestellung hinzufügen

Anwendungsfall	Der Bestellung einen Artikel hinzufügen und löschen (AW 18 und AW 19)
Verwendete Methode	<i>onCreateView (LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)</i> (Klasse: "ItemSelect")
Normalablauf	Allen verfügbaren Artikeln werden ein „Plus“ und ein „Minus“ Button hinzugefügt. Wenn der „Plus“ Button geklickt wird, wird der Bestellung der ausgewählte Artikel einmal hinzugefügt. Wird der „Minus“ Button geklickt, wird der ausgewählte Artikel einmal von der Bestellung entfernt.
Erwartetes Testergebnis	Alle Artikel werden hinzugefügt oder abgezogen.
Tatsächliches Testergebnis	Die ausgewählten Artikel werden der Bestellung hinzugefügt beziehungsweise abgezogen.
Sonderfall 1	Es soll ein Artikel hinzugefügt werden, der die Menge „0“ besitzt. Dieser Artikel befindet sich nicht mehr im Lager.
Erwartetes Testergebnis	Der Artikel lässt sich der Bestellung nicht mehr hinzufügen.
Tatsächliches Testergebnis	Der Artikel lässt sich der Bestellung nicht mehr hinzufügen.
Sonderfall 2	Von der Bestellung soll durch Klicken auf den „Minus“ Button ein Artikel entfernt werden, der in der Bestellung nicht mehr vorhanden ist.
Erwartetes Testergebnis	In der Bestellung wird der Artikel weiterhin mit „0“ angegeben und ist in der Bestellung somit nicht vorhanden.
Tatsächliches Testergebnis	In der Bestellung wird der Artikel weiterhin mit „0“ angegeben und ist in der Bestellung somit nicht vorhanden.
Sonderfall 3	Von der Bestellung soll durch Klicken auf den „Minus“ Button ein Artikel entfernt werden, der im Vorfeld schon bestellt wurde.
Erwartetes Testergebnis	Die Anzahl der Artikel die bereits bestellt wurden, wird nicht unterschritten.

Tatsächliches Testergebnis	Die Anzahl der Artikel die bereits bestellt wurden, wird nicht unterschritten.
Test bestanden	Normalablauf: Ja Sonderfall 1: Ja Sonderfall 2: Ja Sonderfall 3: Ja

Gesamtpreis darstellen

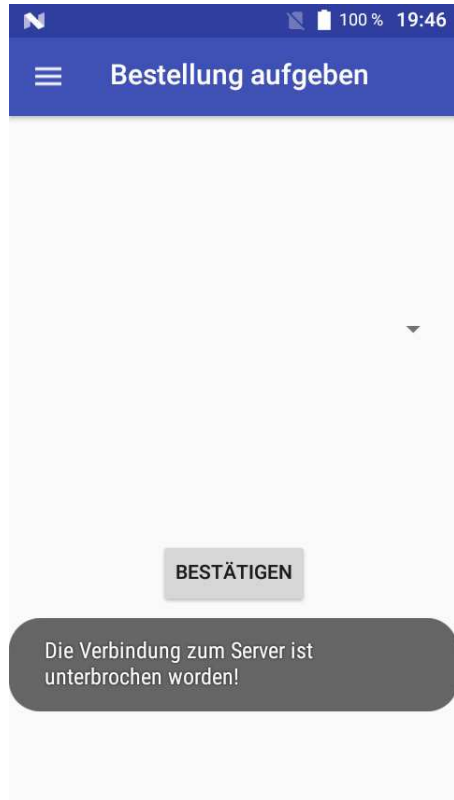
Anwendungsfall	Den aktuellen Gesamtpreis einer Bestellung anzeigen.												
Verwendete Methode	<i>updateSum (boolean isAdd, String itemName)</i> (Klasse: "ItemSelect")												
Normalablauf	Immer wenn ein Artikel einer Bestellung hinzugefügt oder abgezogen wird, errechnet die Applikation die Gesamtsumme der Bestellung und zeigt diese an.												
Erwartetes Testergebnis	<div>Die aktuelle Summe wird dargestellt.</div> <div>Mit folgenden Artikeln aus der Datenbank:</div> <div>2-mal Pommes</div> <div>1-mal Rote Wurst</div> <table><thead><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr></thead><tbody><tr><td>1</td><td>Pommes</td><td>2.5</td><td>1</td></tr><tr><td>2</td><td>Rote Wurst</td><td>3.5</td><td>1</td></tr></tbody></table> <div>Erwarteter Gesamtpreis der Bestellung: 8.50 €</div>	itemID	name	retailprice	available	1	Pommes	2.5	1	2	Rote Wurst	3.5	1
itemID	name	retailprice	available										
1	Pommes	2.5	1										
2	Rote Wurst	3.5	1										
Tatsächliches Testergebnis	Der von der Applikation errechnete Gesamtpreis:												

	
Sonderfall	Keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Bestellung abschicken

Anwendungsfall	Eine von der Bedienung zusammengestellte Bestellung soll an die Küche geschickt werden.
Verwendete Methode	<code>doInBackground()</code> (Klasse: "ItemSelect")
Normalablauf	Die Bedienung stellt für den Kunden eine Bestellung zusammen. Anschließend klickt sie auf den Button „BESTELLEN“ und sendet der Küche die Bestellung.

Erwartetes Testergebnis	<p>In der Datenbank wird eine neue Bestellung erstellt. Diese beinhaltet den ausgewählten Tisch, die ausgewählten Artikel, den Gesamtpreis, die Bestellungs-ID, das Datum inklusive Uhrzeit wann die Bestellung erstellt wurde und ob die Bestellung bereits bezahlt wurde.</p> <p>Mit folgenden Daten wird getestet:</p> <ul style="list-style-type: none">• 2-mal Pommes, 1-mal Rote Wurst• Preis: 8.50 €• Tisch-ID: 1• Datum und Uhrzeit: 02.01.2018, 18:49 Uhr• Nur bestellt nicht bezahlt																																																																																										
Tatsächliches Testergebnis	<p>Die Datenbank sieht wie folgt aus:</p> <table><thead><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr></thead><tbody><tr><td>63</td><td>10:10:10:10:10:10:10:10:10:10:</td><td>110</td><td>2018-01-02 17:28:45</td><td>2</td><td>1</td></tr><tr><td>64</td><td>10:10:10:</td><td>33</td><td>2018-01-02 18:05:57</td><td>1</td><td>1</td></tr><tr><td>65</td><td>10:10:10:10:</td><td>44</td><td>2018-01-02 18:06:18</td><td>1</td><td>1</td></tr><tr><td>66</td><td>8:8:8:8:8:8:8:8:8:8:8:8:8:</td><td>75</td><td>2018-01-02 18:08:51</td><td>1</td><td>1</td></tr><tr><td>67</td><td>10:10:10:10:</td><td>44</td><td>2018-01-02 18:09:57</td><td>2</td><td>1</td></tr><tr><td>68</td><td>8:8:8:8:8:7:7:7:7:7:7:10:</td><td>85</td><td>2018-01-02 18:10:43</td><td>2</td><td>1</td></tr><tr><td>69</td><td>4:4:4:4:4:4:4:11:11:11:11:4:4:</td><td>140.5</td><td>2018-01-02 18:13:00</td><td>1</td><td>1</td></tr><tr><td>70</td><td>3:3:3:</td><td>9</td><td>2018-01-02 18:13:12</td><td>1</td><td>1</td></tr><tr><td>71</td><td>4:4:4:4:4:4:4:4:4:4:4:4:4:</td><td>67.5</td><td>2018-01-02 18:14:59</td><td>1</td><td>1</td></tr><tr><td>72</td><td>2:2:2:2:2:1:1:1:1:1:8:8:8:8:</td><td>55</td><td>2018-01-02 18:15:46</td><td>1</td><td>1</td></tr><tr><td>73</td><td>1:2:2:1:</td><td>12</td><td>2018-01-02 18:33:17</td><td>1</td><td>1</td></tr><tr><td>74</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:48:21</td><td>1</td><td>1</td></tr><tr><td>75</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:49:47</td><td>1</td><td>0</td></tr><tr><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td></tr></tbody></table>	orderID	itemIDs	price	date	tableID	paid	63	10:10:10:10:10:10:10:10:10:10:	110	2018-01-02 17:28:45	2	1	64	10:10:10:	33	2018-01-02 18:05:57	1	1	65	10:10:10:10:	44	2018-01-02 18:06:18	1	1	66	8:8:8:8:8:8:8:8:8:8:8:8:8:	75	2018-01-02 18:08:51	1	1	67	10:10:10:10:	44	2018-01-02 18:09:57	2	1	68	8:8:8:8:8:7:7:7:7:7:7:10:	85	2018-01-02 18:10:43	2	1	69	4:4:4:4:4:4:4:11:11:11:11:4:4:	140.5	2018-01-02 18:13:00	1	1	70	3:3:3:	9	2018-01-02 18:13:12	1	1	71	4:4:4:4:4:4:4:4:4:4:4:4:4:	67.5	2018-01-02 18:14:59	1	1	72	2:2:2:2:2:1:1:1:1:1:8:8:8:8:	55	2018-01-02 18:15:46	1	1	73	1:2:2:1:	12	2018-01-02 18:33:17	1	1	74	1:1:2:	8.5	2018-01-02 18:48:21	1	1	75	1:1:2:	8.5	2018-01-02 18:49:47	1	0	NULL	NULL	NULL	NULL	NULL	NULL
orderID	itemIDs	price	date	tableID	paid																																																																																						
63	10:10:10:10:10:10:10:10:10:10:	110	2018-01-02 17:28:45	2	1																																																																																						
64	10:10:10:	33	2018-01-02 18:05:57	1	1																																																																																						
65	10:10:10:10:	44	2018-01-02 18:06:18	1	1																																																																																						
66	8:8:8:8:8:8:8:8:8:8:8:8:8:	75	2018-01-02 18:08:51	1	1																																																																																						
67	10:10:10:10:	44	2018-01-02 18:09:57	2	1																																																																																						
68	8:8:8:8:8:7:7:7:7:7:7:10:	85	2018-01-02 18:10:43	2	1																																																																																						
69	4:4:4:4:4:4:4:11:11:11:11:4:4:	140.5	2018-01-02 18:13:00	1	1																																																																																						
70	3:3:3:	9	2018-01-02 18:13:12	1	1																																																																																						
71	4:4:4:4:4:4:4:4:4:4:4:4:4:	67.5	2018-01-02 18:14:59	1	1																																																																																						
72	2:2:2:2:2:1:1:1:1:1:8:8:8:8:	55	2018-01-02 18:15:46	1	1																																																																																						
73	1:2:2:1:	12	2018-01-02 18:33:17	1	1																																																																																						
74	1:1:2:	8.5	2018-01-02 18:48:21	1	1																																																																																						
75	1:1:2:	8.5	2018-01-02 18:49:47	1	0																																																																																						
NULL	NULL	NULL	NULL	NULL	NULL																																																																																						
Sonderfall 1	<p>Wenn an diesem Tisch bereits eine Bestellung besteht, die bis zu diesem Zeitpunkt noch nicht bezahlt wurde. Die Bestellung wird um die neuen Artikel erweitert.</p>																																																																																										
Erwartetes Testergebnis	<p>Es wird in der Datenbank die Bestellung aktualisiert. Mit dem neuen Datum/Uhrzeit und den bereits bestehenden Artikeln und den neu hinzugefügten Artikeln. Der Preis wird ebenfalls aktualisiert. Der Test wird mit der gleichen Bestellung die im Normalfall getestet wurde, gestartet.</p> <p>Es soll eine zusätzliche Rote Wurst bestellt werden.</p>																																																																																										
Tatsächliches Testergebnis	<p>Die Datenbank sieht wie folgt aus:</p>																																																																																										

	<table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>64</td><td>10:10:10:</td><td>33</td><td>2018-01-02 18:05:57</td><td>1</td><td>1</td></tr><tr><td>65</td><td>10:10:10:10:</td><td>44</td><td>2018-01-02 18:06:18</td><td>1</td><td>1</td></tr><tr><td>66</td><td>8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:</td><td>75</td><td>2018-01-02 18:08:51</td><td>1</td><td>1</td></tr><tr><td>67</td><td>10:10:10:10:</td><td>44</td><td>2018-01-02 18:09:57</td><td>2</td><td>1</td></tr><tr><td>68</td><td>8:8:8:8:8:7:7:7:7:7:7:7:10:</td><td>85</td><td>2018-01-02 18:10:43</td><td>2</td><td>1</td></tr><tr><td>69</td><td>4:4:4:4:4:4:4:11:11:11:11:4:4:</td><td>140.5</td><td>2018-01-02 18:13:00</td><td>1</td><td>1</td></tr><tr><td>70</td><td>3:3:3:</td><td>9</td><td>2018-01-02 18:13:12</td><td>1</td><td>1</td></tr><tr><td>71</td><td>4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:</td><td>67.5</td><td>2018-01-02 18:14:59</td><td>1</td><td>1</td></tr><tr><td>72</td><td>2:2:2:2:2:1:1:1:1:1:8:8:8:8:8:</td><td>55</td><td>2018-01-02 18:15:46</td><td>1</td><td>1</td></tr><tr><td>73</td><td>1:2:2:1:</td><td>12</td><td>2018-01-02 18:33:17</td><td>1</td><td>1</td></tr><tr><td>74</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:48:21</td><td>1</td><td>1</td></tr><tr><td>75</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:50:26</td><td>1</td><td>1</td></tr><tr><td>76</td><td>1:1:2:2:</td><td>12</td><td>2018-01-02 18:55:42</td><td>1</td><td>0</td></tr><tr><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td></tr></table>	orderID	itemIDs	price	date	tableID	paid	64	10:10:10:	33	2018-01-02 18:05:57	1	1	65	10:10:10:10:	44	2018-01-02 18:06:18	1	1	66	8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:	75	2018-01-02 18:08:51	1	1	67	10:10:10:10:	44	2018-01-02 18:09:57	2	1	68	8:8:8:8:8:7:7:7:7:7:7:7:10:	85	2018-01-02 18:10:43	2	1	69	4:4:4:4:4:4:4:11:11:11:11:4:4:	140.5	2018-01-02 18:13:00	1	1	70	3:3:3:	9	2018-01-02 18:13:12	1	1	71	4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:	67.5	2018-01-02 18:14:59	1	1	72	2:2:2:2:2:1:1:1:1:1:8:8:8:8:8:	55	2018-01-02 18:15:46	1	1	73	1:2:2:1:	12	2018-01-02 18:33:17	1	1	74	1:1:2:	8.5	2018-01-02 18:48:21	1	1	75	1:1:2:	8.5	2018-01-02 18:50:26	1	1	76	1:1:2:2:	12	2018-01-02 18:55:42	1	0	NULL	NULL	NULL	NULL	NULL	NULL
orderID	itemIDs	price	date	tableID	paid																																																																																						
64	10:10:10:	33	2018-01-02 18:05:57	1	1																																																																																						
65	10:10:10:10:	44	2018-01-02 18:06:18	1	1																																																																																						
66	8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:8:	75	2018-01-02 18:08:51	1	1																																																																																						
67	10:10:10:10:	44	2018-01-02 18:09:57	2	1																																																																																						
68	8:8:8:8:8:7:7:7:7:7:7:7:10:	85	2018-01-02 18:10:43	2	1																																																																																						
69	4:4:4:4:4:4:4:11:11:11:11:4:4:	140.5	2018-01-02 18:13:00	1	1																																																																																						
70	3:3:3:	9	2018-01-02 18:13:12	1	1																																																																																						
71	4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:4:	67.5	2018-01-02 18:14:59	1	1																																																																																						
72	2:2:2:2:2:1:1:1:1:1:8:8:8:8:8:	55	2018-01-02 18:15:46	1	1																																																																																						
73	1:2:2:1:	12	2018-01-02 18:33:17	1	1																																																																																						
74	1:1:2:	8.5	2018-01-02 18:48:21	1	1																																																																																						
75	1:1:2:	8.5	2018-01-02 18:50:26	1	1																																																																																						
76	1:1:2:2:	12	2018-01-02 18:55:42	1	0																																																																																						
NULL	NULL	NULL	NULL	NULL	NULL																																																																																						
Sonderfall 2	Die Verbindung zum Server wird während eine Bestellung erstellt wird unterbrochen.																																																																																										
Erwartetes Testergebnis	Die Bestellung wird nicht gespeichert und eine Fehlermeldung wird angezeigt.																																																																																										
Tatsächliches Testergebnis	<div>Das Smartphone gibt folgende Fehlermeldung aus:</div> <div></div>																																																																																										
Test bestanden	Normalfall: Ja Sonderfall 1: Ja Sonderfall 2: Ja																																																																																										

Rechnung für den Kunden anfordern

Anwen- dungsfall	Eine von der Bedienung zusammengestellte Bestellung soll an die Küche geschickt werden und direkt vom Kunden bezahlt werden.																																																																														
Verwendete Methode	<code>doInBackground()</code> (Klasse: "ItemSelect")																																																																														
Normalab- lauf	Die Bedienung stellt für den Kunden eine Bestellung zusammen. Anschließend klickt sie auf den Button „BEZAHLEN“ und sendet der Küche die Bestellung. Am Bon-Drucker werden zwei Belege ausgedruckt. Einen für die Küche und einen für den Kunden.																																																																														
Erwartetes Testergeb- nis	<p>In der Datenbank wird eine neue Bestellung erstellt. Diese beinhaltet den ausgewählten Tisch, die ausgewählten Artikel, den Gesamtpreis, die Bestellungs-ID, das Datum inklusive Uhrzeit wann die Bestellung erstellt wurde und ob die Bestellung bereits bezahlt wurde.</p> <p>Mit folgenden Daten wird getestet:</p> <ul style="list-style-type: none">• 2-mal Pommies, 1-mal Rote Wurst• Preis: 8.50 €• Tisch-ID: 1• Datum und Uhrzeit: 03.01.2018, 15:45 Uhr• bezahlt																																																																														
Tatsächli- ches Testergeb- nis	<p>Die Datenbank sieht wie folgt aus:</p> <table><thead><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr></thead><tbody><tr><td>75</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:50:26</td><td>1</td><td>1</td></tr><tr><td>76</td><td>1:1:2:2:1:8:</td><td>19.5</td><td>2018-01-02 19:07:40</td><td>1</td><td>1</td></tr><tr><td>77</td><td>10:10:10:</td><td>33</td><td>2018-01-02 19:16:17</td><td>1</td><td>1</td></tr><tr><td>78</td><td>11:11:11:11:</td><td>100</td><td>2018-01-02 19:22:13</td><td>1</td><td>1</td></tr><tr><td>79</td><td>1:1:</td><td>5</td><td>2018-01-02 19:24:35</td><td>1</td><td>1</td></tr><tr><td>80</td><td>11:11:11:11:11:</td><td>125</td><td>2018-01-03 10:45:16</td><td>1</td><td>1</td></tr><tr><td>81</td><td>8:8:8:8:8:8:8:8:8:8:8:</td><td>70</td><td>2018-01-03 13:29:56</td><td>2</td><td>1</td></tr><tr><td>82</td><td>8:8:8:8:8:10:10:</td><td>47</td><td>2018-01-03 13:38:57</td><td>1</td><td>1</td></tr><tr><td>83</td><td>10:10:</td><td>22</td><td>2018-01-03 15:38:56</td><td>2</td><td>1</td></tr><tr><td>84</td><td>8:8:</td><td>10</td><td>2018-01-03 15:39:03</td><td>1</td><td>1</td></tr><tr><td>85</td><td>1:1:1:1:1:</td><td>12.5</td><td>2018-01-03 15:39:27</td><td>1</td><td>1</td></tr><tr><td>87</td><td>1:1:2:</td><td>8.5</td><td>2018-01-03 15:45:04</td><td>1</td><td>1</td></tr></tbody></table>	orderID	itemIDs	price	date	tableID	paid	75	1:1:2:	8.5	2018-01-02 18:50:26	1	1	76	1:1:2:2:1:8:	19.5	2018-01-02 19:07:40	1	1	77	10:10:10:	33	2018-01-02 19:16:17	1	1	78	11:11:11:11:	100	2018-01-02 19:22:13	1	1	79	1:1:	5	2018-01-02 19:24:35	1	1	80	11:11:11:11:11:	125	2018-01-03 10:45:16	1	1	81	8:8:8:8:8:8:8:8:8:8:8:	70	2018-01-03 13:29:56	2	1	82	8:8:8:8:8:10:10:	47	2018-01-03 13:38:57	1	1	83	10:10:	22	2018-01-03 15:38:56	2	1	84	8:8:	10	2018-01-03 15:39:03	1	1	85	1:1:1:1:1:	12.5	2018-01-03 15:39:27	1	1	87	1:1:2:	8.5	2018-01-03 15:45:04	1	1
orderID	itemIDs	price	date	tableID	paid																																																																										
75	1:1:2:	8.5	2018-01-02 18:50:26	1	1																																																																										
76	1:1:2:2:1:8:	19.5	2018-01-02 19:07:40	1	1																																																																										
77	10:10:10:	33	2018-01-02 19:16:17	1	1																																																																										
78	11:11:11:11:	100	2018-01-02 19:22:13	1	1																																																																										
79	1:1:	5	2018-01-02 19:24:35	1	1																																																																										
80	11:11:11:11:11:	125	2018-01-03 10:45:16	1	1																																																																										
81	8:8:8:8:8:8:8:8:8:8:8:	70	2018-01-03 13:29:56	2	1																																																																										
82	8:8:8:8:8:10:10:	47	2018-01-03 13:38:57	1	1																																																																										
83	10:10:	22	2018-01-03 15:38:56	2	1																																																																										
84	8:8:	10	2018-01-03 15:39:03	1	1																																																																										
85	1:1:1:1:1:	12.5	2018-01-03 15:39:27	1	1																																																																										
87	1:1:2:	8.5	2018-01-03 15:45:04	1	1																																																																										
Sonderfall 1	Wenn an diesem Tisch bereits eine Bestellung besteht, die bis zu diesem Zeitpunkt noch nicht bezahlt wurde. Die Bestellung wird um die neuen Artikel erweitert.																																																																														

Erwartetes Testergebnis	<p>Es wird in der Datenbank die Bestellung aktualisiert. Mit dem neuen Datum/Uhrzeit und den bereits bestehenden Artikeln und den neu hinzugefügten Artikeln. Der Preis wird ebenfalls aktualisiert. Der Test wird mit der gleichen Bestellung die im Normalfall getestet wurde, gestartet.</p> <p>Es soll eine zusätzliche Rote Wurst bestellt werden.</p>																																																																																										
Tatsächliches Testergebnis	<p>Die Datenbank sieht wie folgt aus.</p> <table><thead><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr></thead><tbody><tr><td>75</td><td>1:1:2:</td><td>8.5</td><td>2018-01-02 18:50:26</td><td>1</td><td>1</td></tr><tr><td>76</td><td>1:1:2:2:1:8:</td><td>19.5</td><td>2018-01-02 19:07:40</td><td>1</td><td>1</td></tr><tr><td>77</td><td>10:10:10:</td><td>33</td><td>2018-01-02 19:16:17</td><td>1</td><td>1</td></tr><tr><td>78</td><td>11:11:11:11:</td><td>100</td><td>2018-01-02 19:22:13</td><td>1</td><td>1</td></tr><tr><td>79</td><td>1:1:</td><td>5</td><td>2018-01-02 19:24:35</td><td>1</td><td>1</td></tr><tr><td>80</td><td>11:11:11:11:11:</td><td>125</td><td>2018-01-03 10:45:16</td><td>1</td><td>1</td></tr><tr><td>81</td><td>8:8:8:8:8:8:8:8:8:8:8:8:</td><td>70</td><td>2018-01-03 13:29:56</td><td>2</td><td>1</td></tr><tr><td>82</td><td>8:8:8:8:8:10:10:</td><td>47</td><td>2018-01-03 13:38:57</td><td>1</td><td>1</td></tr><tr><td>83</td><td>10:10:</td><td>22</td><td>2018-01-03 15:38:56</td><td>2</td><td>1</td></tr><tr><td>84</td><td>8:8:</td><td>10</td><td>2018-01-03 15:39:03</td><td>1</td><td>1</td></tr><tr><td>85</td><td>1:1:1:1:1:</td><td>12.5</td><td>2018-01-03 15:39:27</td><td>1</td><td>1</td></tr><tr><td>87</td><td>1:1:2:</td><td>8.5</td><td>2018-01-03 15:45:04</td><td>1</td><td>1</td></tr><tr><td>88</td><td>1:1:2:2:</td><td>12</td><td>2018-01-03 15:45:31</td><td>1</td><td>1</td></tr><tr><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td><td>NULL</td></tr></tbody></table>	orderID	itemIDs	price	date	tableID	paid	75	1:1:2:	8.5	2018-01-02 18:50:26	1	1	76	1:1:2:2:1:8:	19.5	2018-01-02 19:07:40	1	1	77	10:10:10:	33	2018-01-02 19:16:17	1	1	78	11:11:11:11:	100	2018-01-02 19:22:13	1	1	79	1:1:	5	2018-01-02 19:24:35	1	1	80	11:11:11:11:11:	125	2018-01-03 10:45:16	1	1	81	8:8:8:8:8:8:8:8:8:8:8:8:	70	2018-01-03 13:29:56	2	1	82	8:8:8:8:8:10:10:	47	2018-01-03 13:38:57	1	1	83	10:10:	22	2018-01-03 15:38:56	2	1	84	8:8:	10	2018-01-03 15:39:03	1	1	85	1:1:1:1:1:	12.5	2018-01-03 15:39:27	1	1	87	1:1:2:	8.5	2018-01-03 15:45:04	1	1	88	1:1:2:2:	12	2018-01-03 15:45:31	1	1	NULL	NULL	NULL	NULL	NULL	NULL
orderID	itemIDs	price	date	tableID	paid																																																																																						
75	1:1:2:	8.5	2018-01-02 18:50:26	1	1																																																																																						
76	1:1:2:2:1:8:	19.5	2018-01-02 19:07:40	1	1																																																																																						
77	10:10:10:	33	2018-01-02 19:16:17	1	1																																																																																						
78	11:11:11:11:	100	2018-01-02 19:22:13	1	1																																																																																						
79	1:1:	5	2018-01-02 19:24:35	1	1																																																																																						
80	11:11:11:11:11:	125	2018-01-03 10:45:16	1	1																																																																																						
81	8:8:8:8:8:8:8:8:8:8:8:8:	70	2018-01-03 13:29:56	2	1																																																																																						
82	8:8:8:8:8:10:10:	47	2018-01-03 13:38:57	1	1																																																																																						
83	10:10:	22	2018-01-03 15:38:56	2	1																																																																																						
84	8:8:	10	2018-01-03 15:39:03	1	1																																																																																						
85	1:1:1:1:1:	12.5	2018-01-03 15:39:27	1	1																																																																																						
87	1:1:2:	8.5	2018-01-03 15:45:04	1	1																																																																																						
88	1:1:2:2:	12	2018-01-03 15:45:31	1	1																																																																																						
NULL	NULL	NULL	NULL	NULL	NULL																																																																																						
Sonderfall 2	Die Verbindung zum Server wird während eine Bestellung erstellt wird unterbrochen.																																																																																										
Erwartetes Testergebnis	Die Bestellung wird nicht gespeichert und eine Fehlermeldung wird angezeigt.																																																																																										
Tatsächliches Testergebnis	<p>Das Smartphone gibt folgende Fehlermeldung aus:</p> <div><div><div><div></div><div>Bestellung aufgeben</div></div><div></div><div>BESTÄTIGEN</div><div>Die Verbindung zum Server ist unterbrochen worden!</div></div></div>																																																																																										

Test bestanden	Normalfall: Ja Sonderfall 1: Ja Sonderfall 2: Ja
-------------------	--

6 Fazit

Zusammenfassend kann gesagt werden, dass die Grundstruktur des Systems entwickelt wurde. Mit dem System kann der normale Tagesablauf in einer Gastronomie bewältigt werden.

Mit Hilfe des Datenbank Managers kann unter anderem der Bestand, sowie der Wareneingang und Warenausgang überprüft und bearbeitet werden. Die Android-Anwendung hilft dem Personal/den Bedienungsen dabei, Bestellungen der Kunden aufzunehmen und abzurechnen.

Das System weist jedoch noch Potential auf, die Anwendungen benutzerfreundlicher zu gestalten. Durch Implementierung verschiedener zusätzlicher Funktionen kann das System den Arbeitsalltag im Gastronomie-Bereich noch vereinfachen.

Hierbei sind verschiedene Funktionen mit unterschiedlichem Nutzen für den Anwender zu nennen. Beispielsweise kann das System dahingehend erweitert werden, dass die gesammelten Daten, unter anderem von den verschiedenen Bestellungen, grafisch (in Form eines Diagramms) dargestellt werden können. Dadurch kann der Nutzer sich zu jedem Zeitpunkt einen Überblick darüber verschaffen, welche Artikel, zu welcher Uhrzeit, gut beziehungsweise weniger gut, verkauft werden.

Des Weiteren kann das System noch um eine Login-Funktion erweitert werden. Dann kann nicht mehr jeder auf das System zugreifen, sondern lediglich die vom System-Administrator authentifizierte Personen.

Dies sind nur zwei Möglichkeiten die Struktur zu erweitern. Andere Funktionen wären zum Beispiel: In der Android-Applikation kann die Rechnung getrennt werden, die Artikel in der Applikation nach Getränken und Speisen sortiert, der jeweiligen Bestellung ein Kundenwunsch beigelegt werden und weiteres.

7 Anhang

7.1 Installationsanweisung

Im Folgenden wird beschrieben, wie man das Datenbank-System auf einem Computer installiert und die Android Applikation auf einem Android fähigem Smartphone installiert.

7.1.1 Datenbank-System

Die folgenden Schritte müssen durchgeführt werden, um das Datenbank-System auf einem Computer zu installieren und betreiben.

1. Wenn nicht vorhanden, JRE (Java Runtime Environment) installieren.
2. Installation des MySQL Community Servers
Download des MySQL-installers von:
<https://dev.mysql.com/downloads/windows/installer/5.5.html>
Folgende Komponenten installieren:
 - MySQL-Server
 - MySQL-Workbench
 - MySQL-Notifier
3. Importieren der Datenbank Strukturen
 - MySQL-Workbench öffnen: Management -> Data Import/Restore
 - "Import from Dump Project Folder":
Die Datei "Datenbank Import/Dump20171128.sql" auswählen
 - Im Drop-Down-Menü "Dump Structure Only" auswählen
 - Importieren mit "Start Import"
4. Anlegen eines neuen Users für den Database-Service
 - MySQL-Workbench: Management -> Users and Privileges
 - Den folgenden User anlegen:
 - o Login Name: DatabaseService
 - o Password: password
 - o Im Tab "Administrative Roles": "DBManager" auswählen
 - Anwenden mit "Apply"

5. Installieren des Druckertreibers

- Download des Treibers: https://download.epson-biz.com/modules/pos/index.php?page=single_soft&cid=5131&pcat=3&scat=31
- APD_507_T88V.exe im Ordner "Druckertreiber APD_507_T88V_EWM" starten
- Installationsanweisungen folgen

6. Download des aktuellsten DatabaseSystems

- <https://github.com/Kassensystem/DatabaseSystem/releases/latest>
- Zum Download auf "Source Code (zip)" klicken
- Entpacken der Dateien

7. Download der aktuellsten ManagerApplication

- <https://github.com/Kassensystem/ManagerApplication/releases/latest>
- Zum Download auf "Source Code (zip)" klicken
- Entpacken der Dateien

8. Anpassen der Firewall

- Um dem Server eine Kommunikation im lokalen Netzwerk zu ermöglichen die folgenden Änderungen durchführen:
 - o Windows-Firewall öffnen
 - o Firewall komplett deaktivieren

9. Starten des Servers

- In entpackten Dateien des DatabaseSystems:
 - o Datei "start.bat" starten
 - o Das Kommandozeilenfenster geöffnet lassen
 - o Zum Beenden des Servers das Fenster schließen.

10. Starten der ManagerApplication

- In entpackten Dateien der ManagerApplication:
 - o Datei "start.bat" starten
 - o Oder "kassensystem_manager.exe" im Pfad: "\\out\\artifacts\\kassensystem_managerApplication\\bundles\\kassensystem_managerApplication"

7.1.2 Android Applikation

Die folgenden Schritte müssen durchgeführt werden, um die Applikation auf einem Smartphone zu installieren und zu betreiben.

1. Voraussetzung: Betriebssystem: Android, Version > 5
2. Installation von Apps aus unbekannten Quellen zulassen:
Einstellungen öffnen → Unterpunkt *Sicherheit* auswählen → *Unbekannte Herkunft* muss aktivieren
3. Download der APK auf das Smartphone:
<https://github.com/Nunay/Kassensytem-AndroidApplikation>
4. Heruntergeladene Datei öffnen und installieren
Ordner *Dateien* öffnen → *app-release.apk* öffnen → installieren
5. Applikation starten

8 Testdokumentation

Im Folgenden sind die Tests der öffentlichen Methoden der Software-Module dokumentiert. Es wird jeweils der Anwendungsfall und die getestete Methode beschrieben, außerdem Normalablauf und Sonderfälle, bspw. einem Laufzeitfehler wie Verbindungsprobleme oder falsche Eingaben. Anschließend wird das zu erwartenden und das tatsächliche Testergebnis für den Normalablauf und Sonderfall dokumentiert. In einigen Fällen wurden die Testergebnisse gekürzt.

8.1 Datenbank-Modul

8.1.1 DatabaseService-Klasse

8.1.1.1 Verbindung zur Datenbank

Anwendungsfall	Der DatabaseService verbindet sich mit der MySQL-Datenbank.
Verwendete Methode	<i>Void</i> <i>connect()</i> Stellt eine Verbindung zur MySQL-Datenbank her.
Normalablauf	Wenn die MySQL-Datenbank definitionsgemäß installiert ist und läuft, wird mit dieser eine Verbindung hergestellt.
Erwartetes Testergebnis	Ausgabe der erfolgreichen Verbindungsaufnahme in der Konsole.
Tatsächliches Testergebnis	<i>2017-11-26 13:45:00.113 MYSQL-Info Connecting database...</i> <i>2017-11-26 13:45:00.883 MYSQL-Info Database connected!</i>
Sonderfall	Die MySQL-Datenbank wurde nicht definitionsgemäß installiert oder läuft nicht auf dem Computer.
Erwartetes Testergebnis	Ausgabe der Fehlermeldung in der Konsole.
Tatsächliches Testergebnis	<i>2017-11-26 13:47:33.331 MYSQL-Info Connecting database...</i> <i>2017-11-26 13:47:35.933 MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i>
Test bestanden	Normalablauf: Ja Sonderfall: Ja

8.1.1.2 Abrufen von Datenbankinhalten

Artikel-Daten

Anwendungsfall	Abfragen von Artikel-Daten aus der Datenbank.																																								
Verwendete Methode	<i>java.util.ArrayList<Item></i> <i>getAllItems()</i> Fragt die Artikel der Datenbank ab.																																								
Normalablauf	Alle Artikel werden als Item-Objekte aus der MySQL-Datenbank abgefragt und anschließend zu Testzwecken ausgegeben.																																								
Erwartetes Testergebnis	Rückgabe einer Liste von Item-Objekten. Validierung über eine Konsolenausgabe der folgenden Datenbankinhalte: <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>1</td><td>Pizza</td><td>7</td><td>1</td></tr><tr><td>2</td><td>Pommes</td><td>3.5</td><td>1</td></tr><tr><td>3</td><td>Nudeln</td><td>11</td><td>1</td></tr><tr><td>4</td><td>Lasagne</td><td>7.5</td><td>1</td></tr><tr><td>5</td><td>Kaffee</td><td>2.3</td><td>1</td></tr><tr><td>6</td><td>Cola</td><td>1.5</td><td>0</td></tr><tr><td>9</td><td>Bier</td><td>3.3</td><td>0</td></tr><tr><td>10</td><td>Brot</td><td>5</td><td>0</td></tr></table>					itemID	name	retailprice	available	1	Pizza	7	1	2	Pommes	3.5	1	3	Nudeln	11	1	4	Lasagne	7.5	1	5	Kaffee	2.3	1	6	Cola	1.5	0	9	Bier	3.3	0	10	Brot	5	0
itemID	name	retailprice	available																																						
1	Pizza	7	1																																						
2	Pommes	3.5	1																																						
3	Nudeln	11	1																																						
4	Lasagne	7.5	1																																						
5	Kaffee	2.3	1																																						
6	Cola	1.5	0																																						
9	Bier	3.3	0																																						
10	Brot	5	0																																						
Tatsächliches Testergebnis	<i>2017-11-26 14:01:13.697 MySQL-Info Connecting database...</i> <i>2017-11-26 14:01:14.654 MySQL-Info Database connected!</i> <i>2017-11-26 14:01:14.654 MySQL-Info Getting Items from MySQL-Data-</i> <i>base.</i> <i>-----All-Items-Test-----</i> <i>1 Pizza 7.0 4 true</i> <i>2 Pommes 3.5 7 true</i> <i>3 Nudeln 11.0 4 true</i> <i>4 Lasagne 7.5 6 true</i> <i>5 Kaffee 2.3 45 true</i> <i>6 Cola 1.5 6 false</i> <i>9 Bier 3.3 10 false</i> <i>10 Brot 5.0 10 false</i>																																								
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																																								
Erwartetes Testergebnis	Fehlerausgabe über den Verbindungsfehler in der Konsole.																																								
Tatsächliches Testergebnis	<i>2017-11-26 14:30:59.934 MySQL-Info Connecting database...</i> <i>2017-11-26 14:31:02.756 MySQL-ERROR Verbindung zur Datenbank fehlge-</i> <i>schlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehl-</i> <i>geschlagen!</i>																																								

Test bestanden	Normalablauf: Ja Sonderfall: Ja
-------------------	--

Anwendungsfall	Abfragen eines Artikels mit einer bestimmten ID.								
Verwendete Methode	<i>Item</i> <i>getItemById(int itemID)</i> Liefert ein Item in Abhängigkeit von einer ID								
Normalablauf	Artikel mit der ID existiert in der Datenbank und wird als Item-Objekt zurückgeliefert.								
Erwartetes Testergebnis	Rückgabe des Item-Objekts. Validierung über eine Konsolenausgabe des folgenden Artikels: <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>18</td><td>Steak</td><td>15</td><td>1</td></tr></table>	itemID	name	retailprice	available	18	Steak	15	1
itemID	name	retailprice	available						
18	Steak	15	1						
Tatsächliches Testergebnis	<i>2017-11-26 15:02:28.831 MySQL-Info Connecting database...</i> <i>2017-11-26 15:02:29.549 MySQL-Info Database connected!</i> <i>2017-11-26 15:02:29.549 MySQL-Info Getting Order with ID 18.</i> <i>-----Item-By-ID-Test-----</i> <i>18 Steak 15.0 17 true</i>								
Sonderfall	1. Keine Verbindung zur MySQL-Datenbank aufbaubar. 2. Die angegebene ID existiert nicht. 3. Es wurde keine ID angegeben.								
Erwartetes Testergebnis	1. Fehlerausgabe über den Verbindungsfehler in der Konsole. 2. Fehlermeldung, die eine unbekannte ID meldet. 3. Fehlermeldung, die eine fehlende ID meldet.								
Tatsächliches Testergebnis	1. <i>2017-11-26 15:36:24.184 MySQL-Info Connecting database...</i> <i>2017-11-26 15:36:27.124 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2. <i>2017-11-26 15:41:14.551 MySQL-Info Connecting database...</i> <i>2017-11-26 15:41:15.418 MySQL-Info Database connected!</i> <i>-----Item-By-ID-Test-----</i> <i>2017-11-26 15:41:15.418 MySQL-Info Getting Item with ID 7.</i> <i>2017-11-26 15:41:15.489 MySQL-ERROR Item with ID 7 doesn't exist in the database.</i> <i>Exception in thread "main" java.lang.NullPointerException: Item-ID 7 not found.</i>								

	<p>3. 2017-11-26 15:42:19.738 MySQL-Info Connecting database... 2017-11-26 15:42:20.552 MySQL-Info Database connected! Exception in thread "main" java.lang.NullPointerException: No Item-ID given. -----Item-By-ID-Test----- 2017-11-26 15:42:20.553 MySQL-ERROR Item-ID may not be null.</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Bestellungs-Daten

Anwendungsfall	Abfragen von allen Bestellungen aus der Datenbank.																																										
Verwendete Methode	<i>java.util.ArrayList<Order></i> <i>getAllOrders()</i> Fragt die Bestellungen der Datenbank ab.																																										
Normalablauf	Alle Bestellungen werden als Order-Objekte aus der MySQL-Datenbank abgefragt und anschließend zu Testzwecken ausgegeben.																																										
Erwartetes Testergebnis	Rückgabe einer Liste von Order-Objekten. Validierung über eine Konsolenausgabe der folgenden Datenbankinhalte: <table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>1</td><td>2;1;</td><td>2.3</td><td>2017-10-19 15:40:25</td><td>5</td><td>1</td></tr><tr><td>3</td><td>1;3;</td><td>18</td><td>2017-10-23 10:44:48</td><td>6</td><td>1</td></tr><tr><td>4</td><td>1;2;3;4;5;</td><td>31.3</td><td>2017-11-05 15:38:42</td><td>3</td><td>0</td></tr><tr><td>5</td><td>2;3;</td><td>9.8</td><td>2017-11-08 23:31:33</td><td>5</td><td>1</td></tr><tr><td>12</td><td>4;5;</td><td>9.8</td><td>2017-11-14 21:32:22</td><td>5</td><td>1</td></tr><tr><td>17</td><td>3;5;</td><td>33.99</td><td>2017-11-12 14:42:34</td><td>10</td><td>1</td></tr></table>	orderID	itemIDs	price	date	tableID	paid	1	2;1;	2.3	2017-10-19 15:40:25	5	1	3	1;3;	18	2017-10-23 10:44:48	6	1	4	1;2;3;4;5;	31.3	2017-11-05 15:38:42	3	0	5	2;3;	9.8	2017-11-08 23:31:33	5	1	12	4;5;	9.8	2017-11-14 21:32:22	5	1	17	3;5;	33.99	2017-11-12 14:42:34	10	1
orderID	itemIDs	price	date	tableID	paid																																						
1	2;1;	2.3	2017-10-19 15:40:25	5	1																																						
3	1;3;	18	2017-10-23 10:44:48	6	1																																						
4	1;2;3;4;5;	31.3	2017-11-05 15:38:42	3	0																																						
5	2;3;	9.8	2017-11-08 23:31:33	5	1																																						
12	4;5;	9.8	2017-11-14 21:32:22	5	1																																						
17	3;5;	33.99	2017-11-12 14:42:34	10	1																																						
Tatsächliches Testergebnis	<i>2017-11-26 14:27:10.445 MySQL-Info Connecting database...</i> <i>2017-11-26 14:27:11.128 MySQL-Info Database connected!</i> <i>2017-11-26 14:27:11.128 MySQL-Info Getting Orders from MySQL-Database.</i> <i>-----All-Orders-Test-----</i> <table><tr><td><i>1</i></td><td><i>2;1;</i></td><td><i>2.3</i></td><td><i>19.10.2017 15:40:25</i></td><td><i>5</i></td><td><i>true</i></td></tr><tr><td><i>3</i></td><td><i>1;3;</i></td><td><i>18.0</i></td><td><i>23.10.2017 10:44:48</i></td><td><i>6</i></td><td><i>true</i></td></tr><tr><td><i>4</i></td><td><i>1;2;3;4;5;</i></td><td><i>31.3</i></td><td><i>05.11.2017 15:38:42</i></td><td><i>3</i></td><td><i>false</i></td></tr><tr><td><i>5</i></td><td><i>2;3;</i></td><td><i>9.8</i></td><td><i>08.11.2017 23:31:33</i></td><td><i>5</i></td><td><i>true</i></td></tr><tr><td><i>12</i></td><td><i>4;5;</i></td><td><i>9.8</i></td><td><i>14.11.2017 21:32:22</i></td><td><i>5</i></td><td><i>true</i></td></tr><tr><td><i>17</i></td><td><i>3;5;</i></td><td><i>33.99</i></td><td><i>12.11.2017 14:42:34</i></td><td><i>10</i></td><td><i>true</i></td></tr></table>	<i>1</i>	<i>2;1;</i>	<i>2.3</i>	<i>19.10.2017 15:40:25</i>	<i>5</i>	<i>true</i>	<i>3</i>	<i>1;3;</i>	<i>18.0</i>	<i>23.10.2017 10:44:48</i>	<i>6</i>	<i>true</i>	<i>4</i>	<i>1;2;3;4;5;</i>	<i>31.3</i>	<i>05.11.2017 15:38:42</i>	<i>3</i>	<i>false</i>	<i>5</i>	<i>2;3;</i>	<i>9.8</i>	<i>08.11.2017 23:31:33</i>	<i>5</i>	<i>true</i>	<i>12</i>	<i>4;5;</i>	<i>9.8</i>	<i>14.11.2017 21:32:22</i>	<i>5</i>	<i>true</i>	<i>17</i>	<i>3;5;</i>	<i>33.99</i>	<i>12.11.2017 14:42:34</i>	<i>10</i>	<i>true</i>						
<i>1</i>	<i>2;1;</i>	<i>2.3</i>	<i>19.10.2017 15:40:25</i>	<i>5</i>	<i>true</i>																																						
<i>3</i>	<i>1;3;</i>	<i>18.0</i>	<i>23.10.2017 10:44:48</i>	<i>6</i>	<i>true</i>																																						
<i>4</i>	<i>1;2;3;4;5;</i>	<i>31.3</i>	<i>05.11.2017 15:38:42</i>	<i>3</i>	<i>false</i>																																						
<i>5</i>	<i>2;3;</i>	<i>9.8</i>	<i>08.11.2017 23:31:33</i>	<i>5</i>	<i>true</i>																																						
<i>12</i>	<i>4;5;</i>	<i>9.8</i>	<i>14.11.2017 21:32:22</i>	<i>5</i>	<i>true</i>																																						
<i>17</i>	<i>3;5;</i>	<i>33.99</i>	<i>12.11.2017 14:42:34</i>	<i>10</i>	<i>true</i>																																						
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																																										

Erwartetes Testergebnis	Fehlerausgabe über den Verbindungsfehler in der Konsole.
Tatsächliches Testergebnis	<p>2017-11-26 14:31:39.702 <i>MYSQL-Info Connecting database...</i></p> <p>2017-11-26 14:31:42.346 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i></p> <p><i>java.lang.ExceptionInInitializerError</i></p> <p><i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Anwendungsfall	Abfragen einer Order mit einer bestimmten ID.												
Verwendete Methode	<i>Order</i> <i>getOrderById(int orderID)</i> Liefert eine Bestellung in Abhängigkeit von einer ID.												
Normalablauf	Bestellung mit der ID existiert in der Datenbank und wird als Order-Objekt zurückgeliefert.												
Erwartetes Testergebnis	Rückgabe des Order-Objekts. Validierung über eine Konsolenausgabe der folgenden Bestellung: <table><tr><td>orderID</td><td>itemIDs</td><td>price</td><td>date</td><td>tableID</td><td>paid</td></tr><tr><td>47</td><td>1;5;18;6;6;6;6;</td><td>30.3</td><td>2017-11-23 15:33:09</td><td>14</td><td>0</td></tr></table>	orderID	itemIDs	price	date	tableID	paid	47	1;5;18;6;6;6;6;	30.3	2017-11-23 15:33:09	14	0
orderID	itemIDs	price	date	tableID	paid								
47	1;5;18;6;6;6;6;	30.3	2017-11-23 15:33:09	14	0								
Tatsächliches Testergebnis	<i>2017-11-26 15:47:07.692 MYSQL-Info Connecting database...</i> <i>2017-11-26 15:47:08.487 MYSQL-Info Database connected!</i> <i>-----Order-By-ID-Test-----</i> <i>2017-11-26 15:47:08.487 MYSQL-Info Getting Order with ID 47.</i> <i>2017-11-26 15:47:08.487 MYSQL-Info Getting Orders from MySQL-Database.</i> <i>47 1;5;18;6;6;6;6; 30.3 23.11.2017 15:33:09 14 false</i>												
Sonderfall	1. Keine Verbindung zur MySQL-Datenbank aufbaubar. 2. Die angegebene ID existiert nicht. 3. Es wurde keine ID angegeben.												
Erwartetes Testergebnis	1. Fehlerausgabe über den Verbindungsfehler in der Konsole. 2. Fehlermeldung, die eine unbekannte ID meldet. 3. Fehlermeldung, die eine fehlende ID meldet.												
Tatsächliches Testergebnis	1. <i>2017-11-26 15:26:59.049 MYSQL-Info Connecting database...</i>												

	<p>2017-11-26 15:27:01.781 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2. 2017-11-26 15:28:04.033 <i>MYSQL-Info Connecting database...</i> 2017-11-26 15:28:04.745 <i>MYSQL-Info Database connected!</i> -----<i>Order-By-ID-Test</i>----- 2017-11-26 15:28:04.745 <i>MYSQL-Info Getting Order with ID 7.</i> <i>Exception in thread "main" java.lang.NullPointerException: Order-ID 7 not found.</i> 2017-11-26 15:28:04.789 <i>MYSQL-ERROR Order with ID 7 doesn't exist in the database.</i> 3. 2017-11-26 15:29:08.224 <i>MYSQL-Info Connecting database...</i> 2017-11-26 15:29:09.066 <i>MYSQL-Info Database connected!</i> <i>Exception in thread "main" java.lang.NullPointerException: No Order-ID given.</i> -----<i>Order-By-ID-Test</i>----- 2017-11-26 15:29:09.067 <i>MYSQL-ERROR Order-ID may not be null.</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Tisch-Daten

Anwendungsfall	Abfragen von Tisch-Daten aus der Datenbank.
Verwendete Methode	<i>java.util.ArrayList<Table></i> <i>getAllTables()</i> Fragt die Tische der Datenbank ab
Normalablauf	Alle Tische werden als Table-Objekte aus der MySQL-Datenbank abgefragt und anschließend zu Testzwecken ausgegeben.

Erwartetes Testergebnis	<p>Rückgabe einer Liste von Table-Objekten. Validierung über eine Konsolenausgabe der folgenden Datenbankinhalte:</p> <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>1</td><td>A1</td><td>1</td></tr><tr><td>2</td><td>A2</td><td>1</td></tr><tr><td>3</td><td>A3</td><td>1</td></tr><tr><td>4</td><td>B1</td><td>1</td></tr><tr><td>5</td><td>B2</td><td>1</td></tr><tr><td>6</td><td>B3</td><td>1</td></tr><tr><td>7</td><td>C1</td><td>1</td></tr><tr><td>8</td><td>C2</td><td>1</td></tr></table>	tableID	name	available	1	A1	1	2	A2	1	3	A3	1	4	B1	1	5	B2	1	6	B3	1	7	C1	1	8	C2	1
tableID	name	available																										
1	A1	1																										
2	A2	1																										
3	A3	1																										
4	B1	1																										
5	B2	1																										
6	B3	1																										
7	C1	1																										
8	C2	1																										
Tatsächliches Testergebnis	<p>2017-11-26 14:28:33.563 MYSQL-Info Connecting database... 2017-11-26 14:28:34.333 MYSQL-Info Database connected! 2017-11-26 14:28:34.333 MYSQL-Info Getting Tables from MySQL-Data- base. -----All-Tables-Test----- 1 A1 2 A2 3 A3 4 B1 5 B2 6 B3 7 C1 8 C2</p>																											
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																											
Erwartetes Testergebnis	Fehlerausgabe über den Verbindungsfehler in der Konsole.																											
Tatsächliches Testergebnis	<p>2017-11-26 14:30:16.657 MYSQL-Info Connecting database... 2017-11-26 14:30:19.492 MYSQL-ERROR Verbindung zur Datenbank fehlge- schlagen! java.lang.ExceptionInInitializerError Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehl- geschlagen!</p>																											
Test bestanden	Normalablauf: Ja Sonderfall: Ja																											

Usecase	Abfragen eines Tisches mit einer bestimmten ID.
Verwendete Methode	<p><i>Table</i> <i>getTableById(int tableID)</i></p> <p>Liefert eine Table in Abhängigkeit von einer ID.</p>

Normalablauf	Bestellung mit der ID existiert in der Datenbank und wird als Order-Objekt zurückgeliefert.						
Erwartetes Testergebnis	Rückgabe des Table-Objekts. Validierung über eine Konsolenausgabe des folgenden Tisches: <table><tr><td>tableID</td><td>name</td><td>available</td></tr><tr><td>9</td><td>C3</td><td>1</td></tr></table>	tableID	name	available	9	C3	1
tableID	name	available					
9	C3	1					
Tatsächliches Testergebnis	<pre>2017-11-26 15:53:21.071 MySQL-Info Connecting database... 2017-11-26 15:53:22.259 MySQL-Info Database connected! -----Table-By-ID-Test----- 2017-11-26 15:53:22.259 MySQL-Info Getting Table with ID 9. 9 C3 true</pre>						
Sonderfall	<ol style="list-style-type: none">Keine Verbindung zur MySQL-Datenbank aufbaubar.Die angegebene ID existiert nicht.Es wurde keine ID angegeben.						
Erwartetes Testergebnis	<ol style="list-style-type: none">Fehlerausgabe über den Verbindungsfehler in der Konsole.Fehlermeldung, die eine unbekannte ID meldet.Fehlermeldung, die eine fehlende ID meldet.						
Tatsächliches Testergebnis	<pre>1. 2017-11-26 16:04:47.801 MySQL-Info Connecting database... 2017-11-26 16:04:50.394 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen! java.lang.ExceptionInInitializerError Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen! 2. 2017-11-26 16:05:57.778 MySQL-Info Connecting database... 2017-11-26 16:05:58.569 MySQL-Info Database connected! -----Table-By-ID-Test----- 2017-11-26 16:05:58.569 MySQL-Info Getting Table with ID 16. 2017-11-26 16:05:58.569 MySQL-Info Getting Tables from MySQL-Database. 2017-11-26 16:05:58.600 MySQL-ERROR Table with ID 16 doesn't exist in the database. Exception in thread "main" java.lang.NullPointerException: Table-ID 16 not found. 3. 2017-11-26 16:05:16.382 MySQL-Info Connecting database... Exception in thread "main" java.lang.NullPointerException: No Table-ID given. 2017-11-26 16:05:17.155 MySQL-Info Database connected! -----Table-By-ID-Test----- 2017-11-26 16:05:17.155 MySQL-ERROR Table-ID may not be null.</pre>						

Test bestanden	Normalablauf: Ja Sonderfall: Ja
-------------------	--

Wareneingangs-Daten

Anwendungsfall	Abfragen von Wareneingangs-Daten aus der Datenbank.																											
Verwendete Methode	<code>java.util.ArrayList<Itemdelivery></code> <code>getAllItemdeliveries()</code> Fragt die Wareneingaenge der Datenbank ab.																											
Normalablauf	Alle Wareneingänge werden als Itemdelivery-Objekte aus der MySQL-Datenbank abgefragt und anschließend ausgegeben.																											
Erwartetes Testergebnis	Rückgabe einer Liste von Itemdelivery-Objekten. Validierung über eine Konsolenausgabe der folgenden Datenbankinhalte: <table><tr><th>itemdeliveryID</th><th>itemID</th><th>quantity</th></tr><tr><td>1</td><td>1</td><td>10</td></tr><tr><td>2</td><td>2</td><td>10</td></tr><tr><td>3</td><td>3</td><td>10</td></tr><tr><td>4</td><td>4</td><td>10</td></tr><tr><td>5</td><td>5</td><td>10</td></tr><tr><td>6</td><td>6</td><td>10</td></tr><tr><td>7</td><td>9</td><td>10</td></tr><tr><td>8</td><td>10</td><td>10</td></tr></table>	itemdeliveryID	itemID	quantity	1	1	10	2	2	10	3	3	10	4	4	10	5	5	10	6	6	10	7	9	10	8	10	10
itemdeliveryID	itemID	quantity																										
1	1	10																										
2	2	10																										
3	3	10																										
4	4	10																										
5	5	10																										
6	6	10																										
7	9	10																										
8	10	10																										
Tatsächliches Testergebnis	<pre>2017-11-26 14:34:40.782 MySQL-Info Connecting database... 2017-11-26 14:34:41.664 MySQL-Info Database connected! 2017-11-26 14:34:41.664 MySQL-Info Getting Itemdeliveries from MySQL- Database. -----All-Itemdeliveries-Test----- 1 1 10 2 2 10 3 3 10 4 4 10 5 5 10 6 6 10 7 9 10 8 10 10</pre>																											
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																											
Erwartetes Testergebnis	Fehlerausgabe über den Verbindungsfehler in der Konsole.																											

Tatsächliches Testergebnis	2017-11-26 14:52:19.976 <i>MYSQL-Info Connecting database...</i> 2017-11-26 14:52:22.625 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i>
Test bestanden	Normalablauf: Ja Sonderfall: Ja

Anwendungsfall	Abfragen eines Wareneingangs mit einer bestimmten ID.						
Verwendete Methode	<i>Itemdelivery getItemdeliveryById(int itemdeliveryID)</i> Liefert eine Itemdelivery in Abhängigkeit von einer ID.						
Normalablauf	Wareneingang mit der ID existiert in der Datenbank und wird als Itemdelivery-Objekt zurückgeliefert.						
Erwartetes Testergebnis	Rückgabe des Itemdelivery-Objekts. Validierung über eine Konsolenausgabe des folgenden Wareneingangs: <table><tr><td>itemdeliveryID</td><td>itemID</td><td>quantity</td></tr><tr><td>23</td><td>26</td><td>500</td></tr></table>	itemdeliveryID	itemID	quantity	23	26	500
itemdeliveryID	itemID	quantity					
23	26	500					
Tatsächliches Testergebnis	<i>2017-11-26 19:15:24.532 MYSQL-Info Connecting database...</i> <i>2017-11-26 19:15:25.265 MYSQL-Info Database connected!</i> <i>-----Itemdelivery-By-ID-Test-----</i> <i>2017-11-26 19:15:25.265 MYSQL-Info Getting Itemdeliveries from MySQL-Database.</i> <i>23 26 500</i>						
Sonderfall	1. Keine Verbindung zur MySQL-Datenbank aufbaubar. 2. Die angegebene ID existiert nicht. 3. Es wurde keine ID angegeben.						
Erwartetes Testergebnis	1. Fehlerausgabe über den Verbindungsfehler in der Konsole. 2. Fehlermeldung, die eine unbekannte ID meldet. 3. Fehlermeldung, die eine fehlende ID meldet.						
Tatsächliches Testergebnis	1. <i>2017-11-26 19:31:18.553 MYSQL-Info Connecting database...</i> <i>2017-11-26 19:31:21.384 MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2.						

	<p>2017-11-26 19:32:30.124 MySQL-Info Connecting database...</p> <p>2017-11-26 19:32:30.821 MySQL-Info Database connected!</p> <p>-----Itemdelivery-By-ID-Test-----</p> <p>2017-11-26 19:32:30.821 MySQL-Info Getting Itemdeliveries from MySQL-Database.</p> <p>Exception in thread "main" java.lang.NullPointerException</p> <p>3.</p> <p>2017-11-26 19:35:13.716 MySQL-Info Connecting database...</p> <p>2017-11-26 19:35:14.612 MySQL-Info Database connected!</p> <p>-----Itemdelivery-By-ID-Test-----</p> <p>2017-11-26 19:35:14.612 MySQL-ERROR Itemdelivery-ID may not be null.</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

8.1.1.3 Hinzufügen von Datenbankinhalten

Artikel hinzufügen

Anwendungsfall	Hinzufügen eines neuen Artikels														
Verwendete Methode	<code>void addItem(Item item)</code> Fügt der Datenbank einen neuen Artikel hinzu.														
Normalablauf	Ein neuer Eintrag in der Datenbanktabelle Item wird erzeugt. Für die angegebene Anzahl des Artikels wird außerdem ein neuer Wareneingang angelegt.														
Erwartetes Testergebnis	Erfolgreiches Erstellen eines neuen Item- und Wareneingangseintrags in der Datenbank. Validierung über einen Abgleich der Datenbank nach einem neu erzeugten Item mit folgenden Daten: <code>String name = "Pizza"; double price = 5.99; int quantity = 15; boolean available = true;</code>														
Tatsächliches Testergebnis	Konsolenausgabe: <code>2017-11-27 13:01:57.813 MySQL-Info Connecting database...</code> <code>2017-11-27 13:01:58.610 MySQL-Info Database connected!</code> <code>-----Add-Item-Test-----</code> <code>0 Pizza 5.99 15 true</code> <code>2017-11-27 13:01:58.610 MySQL-Info Adding Item to MySQL-Database.</code> <code>2017-11-27 13:01:58.641 MySQL-Info Adding Itemdelivery to MySQL-Database.</code> Neue Datenbankeinträge: Artikel-Tabelle: <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>37</td><td>Pizza</td><td>5.99</td><td>1</td></tr></table> Wareneingangs-Tabelle: <table><tr><th>itemdeliveryID</th><th>itemID</th><th>quantity</th></tr><tr><td>47</td><td>37</td><td>15</td></tr></table>	itemID	name	retailprice	available	37	Pizza	5.99	1	itemdeliveryID	itemID	quantity	47	37	15
itemID	name	retailprice	available												
37	Pizza	5.99	1												
itemdeliveryID	itemID	quantity													
47	37	15													
Sonderfall	1. Es besteht keine Verbindung zur MySQL-Datenbank. 2. Eines der Attribute wurde nicht gesetzt. 3. Es wurde dem neuen Artikel eine ID vergeben.														
Erwartetes Testergebnis	1. Die Fehlerhafte Verbindungsaufnahme wird in einer Konsolenausgabe gemeldet. 2. Das fehlende Attribut wird als fehlend gemeldet und kein Artikel hinzugefügt.														

	3. Es wird rückgemeldet, dass die ID nicht vom Anwender, sondern von dem Datenbank-Server gesetzt wird.
Tatsächliches Testergebnis	<p>1.</p> <p>2017-11-27 13:20:20.457 MYSQL-Info Connecting database...</p> <p>2017-11-27 13:20:23.051 MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</p> <p>java.lang.ExceptionInInitializerError</p> <p>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</p> <p>2.</p> <p>2017-11-27 13:26:07.176 MYSQL-Info Connecting database...</p> <p>2017-11-27 13:26:07.895 MYSQL-Info Database connected!</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Der Artikel ist unvollständig! Folgende Attribute fehlen: Name Available</p> <p>-----Add-Item-Test-----</p> <p>0 5.99 15 false</p> <p>2017-11-27 13:26:07.895 MYSQL-Info Adding Item to MySQL-Database.</p> <p>2017-11-27 13:26:07.895 MYSQL-ERROR Name is missing.</p> <p>2017-11-27 13:26:07.895 MYSQL-ERROR New Item cannot be set unavailable.</p> <p>2017-11-27 13:26:07.895 MYSQL-ERROR Item was not added to the Database!</p> <p>3.</p> <p>2017-11-27 13:29:38.309 MYSQL-Info Connecting database...</p> <p>2017-11-27 13:29:39.231 MYSQL-Info Database connected!</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Es darf keine ID übergeben werden. Die ID wird vom Datenbank-Server gewählt!</p> <p>-----Add-Item-Test-----</p> <p>4 5.99 20 false</p> <p>2017-11-27 13:29:39.231 MYSQL-Info Adding Item to MySQL-Database.</p> <p>2017-11-27 13:29:39.231 MYSQL-ERROR ID may not be set by the user.</p> <p>2017-11-27 13:29:39.231 MYSQL-ERROR Item was not added to the Database!</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Bestellung hinzufügen

Anwendungsfall	Hinzufügen einer neuen Bestellung.																																				
Verwendete Methode	<code>void addOrder(Order order)</code> Fügt der Datenbank eine neue Bestellung hinzu.																																				
Normalablauf	Es wird eine neue gültige Bestellung der Datenbank-Tabelle Order hinzugefügt.																																				
Erwartetes Testergebnis	<p>Erfolgreiches Erstellen eines neuen Order-Eintrags in der Datenbank. Validierung über einen Abgleich der Datenbank nach einer neu erzeugten Order mit folgenden Daten:</p> <pre>String itemIDs = "12;13;15"; int tableID = 8; double price = 3.3; boolean paid = true;</pre> <p>Außerdem wird die Bestellung für die Küche ausgedruckt, wenn <code>paid = false</code> gesetzt wird. Wenn <code>paid = true</code> gesetzt, wird einmal ein Küchenbeleg und zusätzlich ein Kundenbeleg ausgedruckt.</p>																																				
Tatsächliches Testergebnis	<p>Konsolenausgabe:</p> <pre>2017-11-27 15:52:30.328 MySQL-Info Connecting database... 2017-11-27 15:52:31.063 MySQL-Info Database connected! -----Add-Order-Test----- 0 12;13;15 3.3 27.11.2017 15:52:31 8 true Neuer Datenbankeintrag:</pre> <table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>58</td><td>12;13;15</td><td>3.3</td><td>2017-11-27 15:52:31</td><td>8</td><td>1</td></tr></table> <p>Beide Belege wurden ausgedruckt.</p> <div><div><p>-----Küchenbeleg-----</p><p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p><p>Ihre Bestellung:</p><table><tr><td>Paprikaauflauf</td><td>13,30 EUR</td></tr><tr><td>Paprika</td><td>0,50 EUR</td></tr><tr><td>Apfel</td><td>0,75 EUR</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>Summe</td><td>3,30 EUR</td></tr><tr><td>inkl. MWST 19%</td><td>0,66 EUR</td></tr></table><p>Sie saßen an Tisch C2. Vielen Dank für Ihren Besuch! 27.11.2017 15:52:31</p></div><div><p>-----Kundenbeleg-----</p><p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p><p>Ihre Bestellung:</p><table><tr><td>Paprikaauflauf</td><td>13,30 EUR</td></tr><tr><td>Paprika</td><td>0,50 EUR</td></tr><tr><td>Apfel</td><td>0,75 EUR</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>Summe</td><td>3,30 EUR</td></tr><tr><td>inkl. MWST 19%</td><td>0,66 EUR</td></tr></table><p>Sie saßen an Tisch C2. Vielen Dank für Ihren Besuch! 27.11.2017 15:52:31</p></div></div>	orderID	itemIDs	price	date	tableID	paid	58	12;13;15	3.3	2017-11-27 15:52:31	8	1	Paprikaauflauf	13,30 EUR	Paprika	0,50 EUR	Apfel	0,75 EUR	<hr/>		Summe	3,30 EUR	inkl. MWST 19%	0,66 EUR	Paprikaauflauf	13,30 EUR	Paprika	0,50 EUR	Apfel	0,75 EUR	<hr/>		Summe	3,30 EUR	inkl. MWST 19%	0,66 EUR
orderID	itemIDs	price	date	tableID	paid																																
58	12;13;15	3.3	2017-11-27 15:52:31	8	1																																
Paprikaauflauf	13,30 EUR																																				
Paprika	0,50 EUR																																				
Apfel	0,75 EUR																																				
<hr/>																																					
Summe	3,30 EUR																																				
inkl. MWST 19%	0,66 EUR																																				
Paprikaauflauf	13,30 EUR																																				
Paprika	0,50 EUR																																				
Apfel	0,75 EUR																																				
<hr/>																																					
Summe	3,30 EUR																																				
inkl. MWST 19%	0,66 EUR																																				

Sonderfall	<ol style="list-style-type: none"> 1. Verbindungsproblem mit dem MySQL-Server 2. Eines der Attribute wurde nicht gesetzt. 3. Eine der Item-IDs existiert nicht in der Datenbank. 4. Die Table-ID existiert nicht in der Datenbank. 5. Es wurde eine Order-ID gesetzt.
Erwartetes Testergebnis	<ol style="list-style-type: none"> 1. Das Verbindungsproblem wird gemeldet. 2. Die fehlenden Attribute (Artikel und Tisch) werden in einer Meldung zurückgegeben. 3. Die nichtexistierende Item-ID wird gemeldet. 4. Die nichtexistierende Table-ID wird gemeldet. 5. Es wird rückgemeldet, dass die ID nicht vom Anwender, sondern von dem Datenbank-Server gesetzt wird.
Tatsächliches Testergebnis	<ol style="list-style-type: none"> 1. 2017-11-27 15:56:38.190 <i>MYSQL-Info Connecting database...</i> 2017-11-27 15:56:40.769 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2. 2017-11-28 13:27:44.886 <i>MYSQL-Info Connecting database...</i> 2017-11-28 13:27:45.624 <i>MYSQL-Info Database connected!</i> -----Add-Order-Test----- 0 0.0 28.11.2017 13:27:45 0 true 2017-11-28 13:27:45.625 <i>MYSQL-Info Adding Order to MySQL-Database.</i> 2017-11-28 13:27:45.715 <i>MYSQL-ERROR Item-IDs missing.</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Die Bestellung ist unvollständig! Die folgenden Parameter fehlen: Artikel Tisch</i> 2017-11-28 13:27:45.715 <i>MYSQL-ERROR Table-ID missing.</i> 2017-11-28 13:27:45.715 <i>MYSQL-ERROR Order was not added to the database!</i> 3. 2017-11-28 13:39:13.824 <i>MYSQL-Info Connecting database...</i> 2017-11-28 13:39:14.574 <i>MYSQL-Info Database connected!</i> -----Add-Order-Test----- 0 15;7;13 0.0 28.11.2017 13:39:14 8 true 2017-11-28 13:39:14.575 <i>MYSQL-Info Adding Order to MySQL-Database.</i> 2017-11-28 13:39:14.677 <i>MYSQL-ERROR One or multiple Item-IDs do not exist in the database!</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Eine oder mehrere angegebene Artikel-IDs existieren nicht in der Datenbank.</i>

	<p>2017-11-28 13:39:14.677 MYSQL-ERROR Order was not added to the Database!</p> <p>4.</p> <p>2017-11-28 13:45:11.977 MYSQL-Info Connecting database...</p> <p>2017-11-28 13:45:12.718 MYSQL-Info Database connected!</p> <p>-----Add-Order-Test-----</p> <p>0 15;13 0.0 28.11.2017 13:45:12 20 true</p> <p>2017-11-28 13:45:12.719 MYSQL-Info Adding Order to MySQL-Database. Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Die angegebene Table-ID existiert nicht in der Datenbank!</p> <p>2017-11-28 13:45:12.817 MYSQL-ERROR The Table-ID does not exist in the database!</p> <p>2017-11-28 13:45:12.818 MYSQL-ERROR Order was not added to the database!</p> <p>5.</p> <p>2017-11-28 13:42:13.595 MYSQL-Info Connecting database...</p> <p>2017-11-28 13:42:14.348 MYSQL-Info Database connected!</p> <p>-----Add-Order-Test-----</p> <p>4 15;13 0.0 28.11.2017 13:42:14 8 true</p> <p>2017-11-28 13:42:14.350 MYSQL-Info Adding Order to MySQL-Database. Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Es darf keine ID übergeben werden. Die ID wird vom Datenbank-Server gewählt!</p> <p>2017-11-28 13:42:14.429 MYSQL-ERROR ID may not be set by the user.</p> <p>2017-11-28 13:42:14.429 MYSQL-ERROR Order was not added to the Database!</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Tisch hinzufügen

Anwendungsfall	Hinzufügen eines neuen Tisches zur Datenbank.
Verwendete Methode	<code>void addTable(Table table)</code> Fügt der Datenbank einen neuen Tisch hinzu.
Normalablauf	Es wird ein neuer gültiger Tisch der Datenbank-Tabelle Table hinzugefügt.
Erwartetes Testergebnis	<p>Erfolgreiches Erstellen eines neuen Tisch-Eintrags in der Datenbank. Validierung über einen Abgleich der Datenbank nach einem neu erzeugten Tisch mit folgenden Daten:</p> <pre>String name = "Tisch5"; boolean available = true;</pre>

Tatsächliches Testergebnis	<p><i>Konsolenausgabe:</i></p> <p>2017-11-28 14:19:37.356 <i>MYSQL-Info Connecting database...</i> 2017-11-28 14:19:38.197 <i>MYSQL-Info Database connected!</i> -----Add-Table-Test----- 0 <i>Tisch5 true</i> 2017-11-28 14:19:38.198 <i>MYSQL-Info Adding Table to MySQL-Database.</i> <i>Datenbankeintrag:</i></p> <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>17</td><td>Tisch5</td><td>1</td></tr></table>	tableID	name	available	17	Tisch5	1
tableID	name	available					
17	Tisch5	1					
Sonderfall	<ol style="list-style-type: none">1. Verbindungsproblem mit dem MySQL-Server2. Eines der Attribute wurde nicht gesetzt.3. Es wurde eine Table-ID gesetzt.						
Erwartetes Testergebnis	<ol style="list-style-type: none">1. Das Verbindungsproblem wird gemeldet.2. Das fehlende Attribut (Name) wird in einer Meldung zurückgegeben.3. Es wird rückgemeldet, dass die ID nicht vom Anwender, sondern von dem Datenbank-Server gesetzt wird.						
Tatsächliches Testergebnis	<ol style="list-style-type: none">1. 2017-11-28 14:23:37.119 <i>MYSQL-Info Connecting database...</i> 2017-11-28 14:23:39.710 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i>2. 2017-11-28 14:25:30.090 <i>MYSQL-Info Connecting database...</i> 2017-11-28 14:25:31.071 <i>MYSQL-Info Database connected!</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Der Tisch ist unvollständig! Die folgenden Parameter fehlen: Tisch</i> -----Add-Table-Test----- 0 <i>true</i> 2017-11-28 14:25:31.072 <i>MYSQL-Info Adding Table to MySQL-Database.</i> 2017-11-28 14:25:31.072 <i>MYSQL-ERROR Table missing.</i> 2017-11-28 14:25:31.072 <i>MYSQL-ERROR Table was not added to the database!</i>3. 2017-11-28 14:27:04.905 <i>MYSQL-Info Connecting database...</i> 2017-11-28 14:27:05.617 <i>MYSQL-Info Database connected!</i>						

	<p><i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Es darf keine ID übergeben werden. Die ID wird vom Datenbank-Server gewählt!</i></p> <p>-----Add-Table-Test-----</p> <p>5 Tisch10 true</p> <p>2017-11-28 14:27:05.618 MySQL-Info Adding Table to MySQL-Database.</p> <p>2017-11-28 14:27:05.618 MySQL-ERROR ID may not be set by the user.</p> <p>2017-11-28 14:27:05.618 MySQL-ERROR Table was not added to the Database!</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Wareneingang hinzufügen

Anwendungsfall	Hinzufügen eines neuen Wareneingangs zu Datenbank.						
Verwendete Methode	<i>void addItemdelivery(Itemdelivery itemdelivery)</i> Fügt der Datenbank einen neuen Wareneingang hinzu						
Normalablauf	Es wird ein neuer gültiger Wareneingang der Datenbank-Tabelle Itemdelivery hinzugefügt.						
Erwartetes Testergebnis	Erfolgreiches Erstellen eines neuen Wareneingags-Eintrags in der Datenbank. Validierung über einen Abgleich der Datenbank nach einem neu erzeugten Wareneingang mit folgenden Daten: <code>int itemID = 5;</code> <code>int quantity = 20;</code>						
Tatsächliches Testergebnis	<i>Konsolenausgabe:</i> <i>2017-11-28 14:34:26.274 MySQL-Info Connecting database...</i> <i>2017-11-28 14:34:26.995 MySQL-Info Database connected!</i> <i>-----Add-Itemdelivery-Test-----</i> <i>0 5 20</i> <i>2017-11-28 14:34:26.996 MySQL-Info Adding Itemdelivery to MySQL-Database.</i> <i>Datenbankeintrag:</i> <table><tr><th>itemdeliveryID</th><th>itemID</th><th>quantity</th></tr><tr><td>48</td><td>5</td><td>20</td></tr></table>	itemdeliveryID	itemID	quantity	48	5	20
itemdeliveryID	itemID	quantity					
48	5	20					
Sonderfall	1. Verbindungsproblem mit dem MySQL-Server 2. Eines der Attribute wurde nicht gesetzt. 3. Die gesetzte Item-ID existiert nicht in der Datenbank.						

	4. Es wurde eine Table-ID gesetzt.
Erwartetes Testergebnis	<p>1. Das Verbindungsproblem wird gemeldet.</p> <p>2. Das fehlende Attribut (Item-ID, Anzahl) wird in einer Meldung zurückgegeben.</p> <p>3. Es wird rückgemeldet, dass die Item-ID in der Datenbank nicht existiert.</p> <p>4. Es wird rückgemeldet, dass die ID nicht vom Anwender, sondern von dem Datenbank-Server gesetzt wird.</p>
Tatsächliches Testergebnis	<p>1.</p> <p>2017-11-28 14:38:43.712 MySQL-Info Connecting database...</p> <p>2017-11-28 14:38:46.334 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</p> <p>java.lang.ExceptionInInitializerError</p> <p>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</p> <p>2.</p> <p>2017-11-28 14:39:31.260 MySQL-Info Connecting database...</p> <p>2017-11-28 14:39:32.024 MySQL-Info Database connected!</p> <p>-----Add-Itemdelivery-Test-----</p> <p>0 0 0</p> <p>2017-11-28 14:39:32.025 MySQL-Info Adding Itemdelivery to MySQL-Database.</p> <p>2017-11-28 14:39:32.135 MySQL-ERROR Item-ID missing</p> <p>2017-11-28 14:39:32.135 MySQL-ERROR Quantity missing.</p> <p>2017-11-28 14:39:32.135 MySQL-ERROR Itemdelivery was not added to the database!</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Der Wareneingang ist unvollständig! Die folgenden Parameter fehlen: Artikel Anzahl</p> <p>3.</p> <p>2017-11-28 14:41:52.971 MySQL-Info Connecting database...</p> <p>2017-11-28 14:41:54.023 MySQL-Info Database connected!</p> <p>-----Add-Itemdelivery-Test-----</p> <p>0 7 50</p> <p>2017-11-28 14:41:54.024 MySQL-Info Adding Itemdelivery to MySQL-Database.</p> <p>2017-11-28 14:41:54.109 MySQL-ERROR The Item-IDs does not exist in the Database!</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Die angegebene Artikel-ID existiert nicht in der Datenbank.</p> <p>2017-11-28 14:41:54.109 MySQL-ERROR Itemdelivery was not added to the Database!</p>

	<p>4.</p> <p>2017-11-28 14:43:46.465 <i>MYSQL-Info Connecting database...</i></p> <p>2017-11-28 14:43:47.428 <i>MYSQL-Info Database connected!</i></p> <p>-----Add-Itemdelivery-Test-----</p> <p>5 5 50</p> <p>2017-11-28 14:43:47.430 <i>MYSQL-Info Adding Itemdelivery to MySQL-Database.</i></p> <p>2017-11-28 14:43:47.506 <i>MYSQL-ERROR ID may not be set by the user.</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Es darf keine ID übergeben werden. Die ID wird vom Datenbank-Server gewählt!</i></p> <p>2017-11-28 14:43:47.506 <i>MYSQL-ERROR Itemdelivery was not added to the Database!</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

8.1.1.4 Aktualisieren von Datenbankinhalten

Artikel aktualisieren

Anwendungsfall	Bearbeiten der Daten eines Artikels.								
Verwendete Methode	<code>void updateItem(int itemID, Item item)</code> Aktualisiert die Daten eines Artikels								
Normalablauf	Der richtige Artikel wird mit den Daten des neuen Items aktualisiert.								
Erwartetes Testergebnis	<div>Der Artikel mit der ID 16 wird aktualisiert.</div> <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>16</td><td>Kuchen</td><td>2.99</td><td>1</td></tr></table> <div>Die folgenden Daten werden zum Aktualisieren übergeben:</div> <pre>int ID = 16; String name = "Kirschkuchen"; double retailprice = 3.49; boolean available = false;</pre>	itemID	name	retailprice	available	16	Kuchen	2.99	1
itemID	name	retailprice	available						
16	Kuchen	2.99	1						
Tatsächliches Testergebnis	<div>Konsolenausgabe:</div> <pre>2017-11-28 16:09:28.476 MySQL-Info Connecting database... 2017-11-28 16:09:29.222 MySQL-Info Database connected! -----Update-Item-Test----- 16 Kirschkuchen 3.49 0 false 2017-11-28 16:09:29.223 MySQL-Info Updating Item with ID 16. Datenbankeintrag:</pre> <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>16</td><td>Kirschkuchen</td><td>3.49</td><td>0</td></tr></table>	itemID	name	retailprice	available	16	Kirschkuchen	3.49	0
itemID	name	retailprice	available						
16	Kirschkuchen	3.49	0						
Sonderfall	<div>1. Es besteht keine Verbindung zur Datenbank.</div> <div>2. Die zu aktualisierende ID existiert nicht in der Datenbank.</div> <div>3. Die übergebenden Daten sind nicht vollständig.</div> <div>4. Es wurde keine zu aktualisierende ID übergeben.</div>								
Erwartetes Testergebnis	<div>1. Verbindungsfehlermeldung wird ausgegeben.</div> <div>2. Eine nichtexistierende ID 14 wird gemeldet.</div> <div>3. Die fehlenden Attribute werden gemeldet.</div> <div>4. Eine fehlende ID wird gemeldet.</div>								
Tatsächliches	<div>1.</div>								

Testergebnis	<p>2017-11-28 16:13:03.562 <i>MYSQL-Info Connecting database...</i> 2017-11-28 16:13:06.420 <i>MYSQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2.</p> <p>2017-11-28 16:13:59.112 <i>MYSQL-Info Connecting database...</i> 2017-11-28 16:13:59.924 <i>MYSQL-Info Database connected!</i> -----Update-Item-Test----- 14 <i>Kirschkuchen</i> 3.49 0 <i>false</i> 2017-11-28 16:13:59.926 <i>MYSQL-Info Updating Item with ID 14.</i> 2017-11-28 16:14:00.021 <i>MYSQL-ERROR Item with ID 14 does not exist in the database!</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Artikel mit der ID 14 existiert nicht in der Datenbank!</i> 3.</p> <p>2017-11-28 16:16:08.972 <i>MYSQL-Info Connecting database...</i> 2017-11-28 16:16:09.796 <i>MYSQL-Info Database connected!</i> -----Update-Item-Test----- 16 0.0 0 <i>false</i> 2017-11-28 16:16:09.798 <i>MYSQL-Info Updating Item with ID 16.</i> 2017-11-28 16:16:09.889 <i>MYSQL-ERROR Name is missing.</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Der Artikel ist unvollständig! Folgende Attribute fehlen: Name</i> 2017-11-28 16:16:09.889 <i>MYSQL-ERROR Item was not added to the Database!</i> 4.</p> <p>2017-11-28 16:17:43.228 <i>MYSQL-Info Connecting database...</i> 2017-11-28 16:17:44.036 <i>MYSQL-Info Database connected!</i> -----Update-Item-Test----- 0 <i>Kirschkuchen</i> 3.49 0 <i>false</i> 2017-11-28 16:17:44.037 <i>MYSQL-Info Updating Item with ID 0.</i> 2017-11-28 16:17:44.119 <i>MYSQL-ERROR Item-ID may not be null.</i> <i>Exception in thread "main" java.lang.NullPointerException: No Item-ID given.</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Tisch aktualisieren

Anwendungsfall	Bearbeiten der Daten eines Tisches.
----------------	-------------------------------------

Verwendete Methode	<code>void updateTable(int tableID, Table table)</code> Aktualisiert die Daten eines Tisches.						
Normalablauf	Der richtige Tisch wird mit den Daten des neuen Table aktualisiert.						
Erwartetes Testergebnis	<p>Der Tisch mit der ID 17 wird aktualisiert:</p> <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>17</td><td>Tisch5</td><td>1</td></tr></table> <p>Die folgenden Daten werden zum Aktualisieren übergeben:</p> <pre>int ID = 17; String name = "Tisch10"; boolean available = false;</pre>	tableID	name	available	17	Tisch5	1
tableID	name	available					
17	Tisch5	1					
Tatsächliches Testergebnis	<p>Konsolenausgabe:</p> <pre>2017-11-28 16:22:02.501 MySQL-Info Connecting database... 2017-11-28 16:22:03.488 MySQL-Info Database connected! -----Update-Table-Test----- 17 Tisch10false 2017-11-28 16:22:03.489 MySQL-Info Updating Table with ID 17. 2017-11-28 16:22:03.489 MySQL-Info Getting Tables from MySQL-Data- base. Datenbankeintrag:</pre> <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>17</td><td>Tisch10</td><td>0</td></tr></table>	tableID	name	available	17	Tisch10	0
tableID	name	available					
17	Tisch10	0					
Sonderfall	<ol style="list-style-type: none">1. Es besteht keine Verbindung zur Datenbank.2. Die zu aktualisierende ID existiert nicht in der Datenbank.3. Die übergebenden Daten sind nicht vollständig.						
Erwartetes Testergebnis	<ol style="list-style-type: none">1. Verbindungsfehlermeldung wird ausgegeben.2. Eine nichtexistierende ID 16 wird gemeldet.3. Die fehlenden Attribute werden gemeldet.						
Tatsächliches Testergebnis	<ol style="list-style-type: none">1.<pre>2017-11-28 16:24:24.418 MySQL-Info Connecting database... 2017-11-28 16:24:27.133 MySQL-ERROR Verbindung zur Datenbank fehlge- schlagen! java.lang.ExceptionInInitializerError Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehl- geschlagen!</pre>2.<pre>2017-11-28 16:25:22.870 MySQL-Info Connecting database... 2017-11-28 16:25:23.770 MySQL-Info Database connected!</pre>						

	<p>-----Update-Table-Test-----</p> <p>16 <i>Tisch10 false</i></p> <p>2017-11-28 16:25:23.771 <i>MYSQL-Info Updating Table with ID 16.</i></p> <p>2017-11-28 16:25:23.803 <i>MYSQL-ERROR Table with ID 16 does not exist in the database!</i></p> <p><i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Tisch mit der ID 16 existiert nicht in der Datenbank!</i></p> <p>3.</p> <p>2017-11-28 16:26:41.177 <i>MYSQL-Info Connecting database...</i></p> <p>2017-11-28 16:26:41.912 <i>MYSQL-Info Database connected!</i></p> <p>-----Update-Table-Test-----</p> <p>17 <i>false</i></p> <p>2017-11-28 16:26:41.913 <i>MYSQL-Info Updating Table with ID 17.</i></p> <p>2017-11-28 16:26:41.940 <i>MYSQL-ERROR Table missing.</i></p> <p><i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Der Tisch ist unvollständig! Die folgenden Parameter fehlen: Tisch</i></p> <p>2017-11-28 16:26:41.940 <i>MYSQL-ERROR Table was not added to the database!</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Bestellung aktualisieren

Anwendungsfall	Bearbeiten der Daten einer Bestellung.												
Verwendete Methode	<i>void updateOrder(int orderID, Order order)</i> Aktualisiert die Daten einer Bestellung.												
Normalablauf	Die richtige Bestellung wird mit den Daten der neuen Order aktualisiert.												
Erwartetes Testergebnis	<p>Die Bestellung mit der ID 45 wird aktualisiert:</p> <table border="1"><thead><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr></thead><tbody><tr><td>45</td><td>16;5;4;</td><td>12.79</td><td>2017-11-22 22:18:14</td><td>1</td><td>0</td></tr></tbody></table> <p>Die folgenden Daten werden zum Aktualisieren übergeben:</p> <pre>int ID = 45; String itemIDs = "16;5;4;25"; double price = 13.29; int tableID = 5; boolean paid = true;</pre>	orderID	itemIDs	price	date	tableID	paid	45	16;5;4;	12.79	2017-11-22 22:18:14	1	0
orderID	itemIDs	price	date	tableID	paid								
45	16;5;4;	12.79	2017-11-22 22:18:14	1	0								

	Wenn paid = false gesetzt ist, wird nur ein Küchenbeleg mit den hinzugefügten Artikeln ausgedruckt. Wenn paid = true ist, wird zusätzlich ein vollständiger Kundenbeleg ausgedruckt.																						
Tatsächliches Testergebnis	<p>Konsolenausgabe:</p> <pre>2017-11-28 16:47:08.875 MySQL-Info Connecting database... 2017-11-28 16:47:09.744 MySQL-Info Database connected! -----Update-Order-Test----- 45 16;5;4;25 13.29 28.11.2017 16:47:09 5 true 2017-11-28 16:47:09.746 MySQL-Info Updating Order with ID 45.</pre> <p>Datenbankeintrag:</p> <table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>45</td><td>16;5;4;25</td><td>13.29</td><td>2017-11-28 16:47:10</td><td>5</td><td>1</td></tr></table> <p>Die Belege wurden beide korrekt ausgedruckt:</p> <div><div><p>-----Kundenbeleg-----</p><p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p><p>Ihre Bestellung:</p><table><tr><td>Kirschkuchen</td><td>3,49 EUR</td></tr><tr><td>Kaffee</td><td>2,30 EUR</td></tr><tr><td>Lasagne</td><td>7,50 EUR</td></tr><tr><td>Cola</td><td>1,50 EUR</td></tr></table><hr/><p>Summe 13,29 EUR inkl. MWST 19% 2,64 EUR</p><p>Sie saßen an Tisch B2. Vielen Dank für Ihren Besuch! 28.11.2017 16:47:09</p></div><div><p>-----Küchenbeleg-----</p><p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p><p>Ihre Bestellung:</p><table><tr><td>Cola</td><td>1,50 EUR</td></tr></table><hr/><p>Summe 13,29 EUR inkl. MWST 19% 2,64 EUR</p><p>Sie saßen an Tisch A1. Vielen Dank für Ihren Besuch! 28.11.2017 16:47:09</p></div></div>	orderID	itemIDs	price	date	tableID	paid	45	16;5;4;25	13.29	2017-11-28 16:47:10	5	1	Kirschkuchen	3,49 EUR	Kaffee	2,30 EUR	Lasagne	7,50 EUR	Cola	1,50 EUR	Cola	1,50 EUR
orderID	itemIDs	price	date	tableID	paid																		
45	16;5;4;25	13.29	2017-11-28 16:47:10	5	1																		
Kirschkuchen	3,49 EUR																						
Kaffee	2,30 EUR																						
Lasagne	7,50 EUR																						
Cola	1,50 EUR																						
Cola	1,50 EUR																						
Sonderfall	<ol style="list-style-type: none">1. Es besteht keine Verbindung zur Datenbank.2. Die zu aktualisierende ID existiert nicht in der Datenbank.3. Die übergebenden Daten sind nicht vollständig.4. Eine der Item-IDs existiert nicht in der Datenbank.5. Die Table-ID existiert nicht in der Datenbank.																						
Erwartetes Testergebnis	<ol style="list-style-type: none">1. Verbindungsfehlermeldung wird ausgegeben.2. Eine nichtexistierende ID 2 wird gemeldet.3. Die fehlenden Attribute werden gemeldet.4. Es wird eine Fehlermeldung über eine nichtexistierende Item-ID ausgegeben.5. Es wird eine Fehlermeldung über eine nichtexistierende Table-ID ausgegeben.																						

<p>Tatsächliches Testergebnis</p>	<pre> 1. 2017-11-28 16:52:06.673 MySQL-Info Connecting database... 2017-11-28 16:52:09.581 MySQL-ERROR Verbindung zur Datenbank fehlge- schlagen! java.lang.ExceptionInInitializerError Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehl- geschlagen! 2. 2017-11-28 16:57:23.970 MySQL-Info Connecting database... 2017-11-28 16:57:24.707 MySQL-Info Database connected! -----Update-Order-Test----- 2 16;5;4;25 13.29 28.11.2017 16:57:24 5 true 2017-11-28 16:57:24.709 MySQL-Info Updating Order with ID 2. 2017-11-28 16:57:24.799 MySQL-Info Getting Tables from MySQL-Data- base. 2017-11-28 16:57:24.804 MySQL-ERROR Order with ID 2 does not exist in the database! dhw.sa.kassensystem_rest.exceptions.DataException: Bestellung mit der ID 2 existiert nicht in der Datenbank! 3. 2017-11-28 16:58:50.037 MySQL-Info Connecting database... 2017-11-28 16:58:50.795 MySQL-Info Database connected! -----Update-Order-Test----- 45 13.29 28.11.2017 16:58:50 0 true 2017-11-28 16:58:50.797 MySQL-Info Updating Order with ID 45. Exception in thread "main" dhw.sa.kassensystem_rest.excepti- ons.DataException: Die Bestellung ist unvollständig! Die folgenden Parame- ter fehlen: Artikel Tisch 2017-11-28 16:58:50.878 MySQL-ERROR Item-IDs missing. 2017-11-28 16:58:50.878 MySQL-ERROR Table-ID missing. 2017-11-28 16:58:50.878 MySQL-ERROR Order was not added to the data- base! 4. 2017-11-28 17:00:18.817 MySQL-Info Connecting database... 2017-11-28 17:00:19.541 MySQL-Info Database connected! -----Update-Order-Test----- 45 16;5;4;25;7 13.29 28.11.2017 17:00:19 5 true 2017-11-28 17:00:19.543 MySQL-Info Updating Order with ID 45. Exception in thread "main" dhw.sa.kassensystem_rest.excepti- ons.DataException: Eine oder mehrere angegebene Artikel-IDs existieren nicht in der Datenbank. 2017-11-28 17:00:19.645 MySQL-ERROR One or multiple Item-IDs do not exist in the database! 2017-11-28 17:00:19.645 MySQL-ERROR Order was not added to the Data- base! 5. 2017-11-28 17:01:49.394 MySQL-Info Connecting database... 2017-11-28 17:01:50.265 MySQL-Info Database connected! -----Update-Order-Test----- </pre>
---------------------------------------	--

	<p>45 16;5;4;25 13.29 28.11.2017 17:01:50 18 true</p> <p>2017-11-28 17:01:50.267 MYSQL-Info Updating Order with ID 45.</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataEx- ception: Die angegebene Table-ID existiert nicht in der Datenbank!</p> <p>2017-11-28 17:01:50.384 MYSQL-ERROR The Table-ID does not exist in the database!</p> <p>2017-11-28 17:01:50.384 MYSQL-ERROR Order was not added to the data- base!</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

8.1.1.5 Löschen von Datenbankinhalten

Das Löschen der Datenbankinhalte Artikel und Tisch funktioniert über ein Aktualisieren des Datenbankeintrags mit Setzen der Verfügbarkeit auf falsch. Dies wurde mit dem Testen des Aktualisierens schon abgedeckt. Daher muss nur das Löschen von Bestellungen und Wareneingängen getestet werden.

Bestellung löschen

Anwendungsfall	Löschen einer Bestellung																	
Verwendete Methode	<i>void deleteOrder(int orderID)</i> Loescht eine Bestellung aus der Datenbank																	
Normalablauf	Die Bestellung wird aus der Datenbank gelöscht.																	
Erwartetes Testergebnis	Die folgende Bestellung existiert nicht mehr in der Datenbank: <table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>52</td><td>13;16;18;</td><td>18.49</td><td>2017-11-23 15:50:20</td><td>1</td><td>0</td></tr></table>						orderID	itemIDs	price	date	tableID	paid	52	13;16;18;	18.49	2017-11-23 15:50:20	1	0
orderID	itemIDs	price	date	tableID	paid													
52	13;16;18;	18.49	2017-11-23 15:50:20	1	0													
Tatsächliches Testergebnis	<i>Konsolenausgabe:</i> <i>2017-11-28 17:33:35.528 MySQL-Info Connecting database...</i> <i>2017-11-28 17:33:36.341 MySQL-Info Database connected!</i> <i>-----Delete-Order-Test-----</i> <i>2017-11-28 17:33:36.342 MySQL-Info Deleting Order with ID 52.</i> <i>Datenbankeintrag:</i> <i>Wurde gelöscht.</i>																	
Sonderfall	1. Es besteht keine Verbindung zu Datenbank. 2. Es existiert keine Bestellung mit der angegebenen ID.																	
Erwartetes Testergebnis	1. Ein Verbindungsfehler wird gemeldet 2. Es wird gemeldet, dass die ID nicht in der Datenbank existiert.																	
Tatsächliches Testergebnis	1. <i>2017-11-28 17:35:02.744 MySQL-Info Connecting database...</i> <i>2017-11-28 17:35:05.441 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i> <i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> 2. <i>2017-11-28 17:35:37.840 MySQL-Info Connecting database...</i> <i>2017-11-28 17:35:38.855 MySQL-Info Database connected!</i>																	

	<p>-----Delete-Order-Test-----</p> <p>2017-11-28 17:35:38.855 MYSQL-Info Deleting Order with ID 59. Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Bestellung mit der ID 59 existiert nicht in der Datenbank! Es konnte nichts gelöscht werden</p> <p>2017-11-28 17:35:38.906 MYSQL-ERROR Order with ID 59 does not exist in the database! Nothing was deleted.</p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Wareneingang löschen

Anwendungsfall	Löschen eines Wareneingangs								
Verwendete Methode	<i>void deleteItemdelivery(int itemdeliveryID)</i> Löscht einen Wareneingang aus der Datenbank								
Normalablauf	Der Wareneingang wird aus der Datenbank gelöscht.								
Erwartetes Testergebnis	Der folgende Wareneingang existiert nicht mehr in der Datenbank: <table><tr><td>itemdeliveryID</td><td>itemID</td><td>quantity</td></tr><tr><td>23</td><td>26</td><td>500</td></tr></table>			itemdeliveryID	itemID	quantity	23	26	500
itemdeliveryID	itemID	quantity							
23	26	500							
Tatsächliches Testergebnis	<i>Konsolenausgabe:</i> <i>2017-11-28 17:40:31.660 MySQL-Info Connecting database...</i> <i>2017-11-28 17:40:32.503 MySQL-Info Database connected!</i> <i>-----Delete-Itemdelivery-Test-----</i> <i>2017-11-28 17:40:32.503 MySQL-Info Deleting Itemdelivery with ID 23.</i> <i>Datenbankeintrag:</i> <i>Wurde gelöscht.</i>								
Sonderfall	1. Es besteht keine Verbindung zu Datenbank. 2. Es existiert keine Bestellung mit der angegebenen ID.								
Erwartetes Testergebnis	1. Ein Verbindungsfehler wird gemeldet 2. Es wird gemeldet, dass die ID nicht in der Datenbank existiert.								
Tatsächliches Testergebnis	1. <i>2017-11-28 17:41:09.252 MySQL-Info Connecting database...</i> <i>2017-11-28 17:41:11.894 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen!</i>								

	<p><i>java.lang.ExceptionInInitializerError</i> <i>Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen!</i> <i>2.</i></p> <p><i>2017-11-28 17:39:15.810 MYSQL-Info Connecting database...</i> <i>2017-11-28 17:39:16.613 MYSQL-Info Database connected!</i> <i>-----Delete-Itemdelivery-Test-----</i> <i>2017-11-28 17:39:16.613 MYSQL-Info Deleting Itemdelivery with ID 59.</i> <i>2017-11-28 17:39:16.647 MYSQL-ERROR Itemdelivery with ID 59 does not exist in the database! Nothing was deleted.</i> <i>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Wareneingang mit der ID 59 existiert nicht in der Datenbank! Es konnte nichts gelöscht werden.</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

8.1.1.6 Drucken einer Bestellung

Anwendungsfall	Drucken einer Bestellung im Kassensystem-Manager.
Verwendete Methode	<i>void printOrderById(int orderID)</i> Ausdrucken einer Bestellung in Abhängigkeit von einer ID.
Normalablauf	Die Bestellung mit der angegebenen ID wird ausgedruckt.
Erwartetes Testergebnis	Ein Ausdruck der Bestellung mit den folgenden Daten:
Tatsächliches Testergebnis	<p>Die Bestellung wurde ausgedruckt:</p> <pre> -----Kundenbeleg----- Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655 Ihre Bestellung: Kirschkuchen 3,49 EUR Kaffee 2,30 EUR Lasagne 7,50 EUR Cola 1,50 EUR Summe 13,29 EUR inkl. MWST 19% 2,64 EUR Sie saßen an Tisch B2. Vielen Dank für Ihren Besuch! 28.11.2017 16:47:10 </pre>
Sonderfall	<ol style="list-style-type: none"> 1. Es besteht keine Verbindung zur Datenbank. 2. Es existiert in der Datenbank keine Bestellung mit der angegebenen ID. 3. Der Drucker ist nicht erreichbar.
Erwartetes Testergebnis	<ol style="list-style-type: none"> 1. Der Verbindungsfehler wird gemeldet. 2. Es wird gemeldet, dass die ID nicht in der Datenbank existiert. 3. Die Bestellung wird mit dem nächsten Anschalten des Druckers ausgedruckt.
Tatsächliches Testergebnis	<ol style="list-style-type: none"> 1. <pre> 2017-11-28 17:41:09.252 MySQL-Info Connecting database... 2017-11-28 17:41:11.894 MySQL-ERROR Verbindung zur Datenbank fehlgeschlagen! java.lang.ExceptionInInitializerError Caused by: java.lang.IllegalStateException: Verbindung zur Datenbank fehlgeschlagen! </pre>

	<p>2.</p> <p>2017-11-28 17:57:54.594 MySQL-Info Connecting database...</p> <p>2017-11-28 17:57:55.445 MySQL-Info Database connected!</p> <p>Exception in thread "main" dhw.sa.kassensystem_rest.exceptions.DataException: Bestellung mit der ID 60 existiert nicht in der Datenbank! Es kann nichts gedruckt werden.</p> <p>2017-11-28 17:57:55.533 MySQL-ERROR Order with ID 60 does not exist in the database!</p> <p>3.</p> <p>Nach einem erneuten Anschalten des Druckers wird die Bestellung ausgedruckt:</p> <p>-----Kundenbeleg-----</p> <p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p> <p>Ihre Bestellung:</p> <table> <tr> <td>Kirschkuchen</td> <td>3,49 EUR</td> </tr> <tr> <td>Kaffee</td> <td>2,30 EUR</td> </tr> <tr> <td>Lasagne</td> <td>7,50 EUR</td> </tr> <tr> <td>Cola</td> <td>1,50 EUR</td> </tr> </table> <hr/> <p>Summe 13,29 EUR inkl. MWST 19% 2,64 EUR</p> <p>Sie saßen an Tisch B2. Vielen Dank für Ihren Besuch! 28.11.2017 16:47:10</p>	Kirschkuchen	3,49 EUR	Kaffee	2,30 EUR	Lasagne	7,50 EUR	Cola	1,50 EUR
Kirschkuchen	3,49 EUR								
Kaffee	2,30 EUR								
Lasagne	7,50 EUR								
Cola	1,50 EUR								
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>								

8.1.2 Server-Modul

8.1.2.1 Abrufen von Datenbankinhalten

Im Folgenden werden die Abrufsfunktionen des Rest-API-Controllers getestet. Dafür wird die entsprechende URL in einen Browser angesprochen und die Rückmeldung überprüft. Für diesen Test muss auf demselben Rechner der Controller laufen und mit der URL „localhost:8080“ erreichbar sein.

Artikel-Daten

Anwendungsfall	Abfragen von Artikel-Daten aus der Datenbank.																																							
Verwendete Methode	<i>java.util.ArrayList<Item></i> <i>getAllItems()</i> Durch das Ansprechen des Pfades ".../api/items" können die Artikel der Datenbank abgefragt werden.																																							
Normalablauf	Alle Artikel werden als Item-Objekte aus der MySQL-Datenbank abgefragt und über Http als JSON-Datei übertragen																																							
Erwartetes Testergebnis	<div>Rückgabe eines JSON-Textes mit folgenden Inhalten:</div> <table><tr><th>itemID</th><th>name</th><th>retailprice</th><th>available</th></tr><tr><td>1</td><td>Pizza</td><td>7</td><td>1</td></tr><tr><td>2</td><td>Pommes</td><td>3.5</td><td>1</td></tr><tr><td>3</td><td>Nudeln</td><td>11</td><td>1</td></tr><tr><td>4</td><td>Lasagne</td><td>7.5</td><td>1</td></tr><tr><td>5</td><td>Kaffee</td><td>2.3</td><td>1</td></tr><tr><td>6</td><td>Cola</td><td>1.5</td><td>0</td></tr><tr><td>9</td><td>Bier</td><td>3.3</td><td>0</td></tr><tr><td>10</td><td>Brot</td><td>5</td><td>0</td></tr></table>				itemID	name	retailprice	available	1	Pizza	7	1	2	Pommes	3.5	1	3	Nudeln	11	1	4	Lasagne	7.5	1	5	Kaffee	2.3	1	6	Cola	1.5	0	9	Bier	3.3	0	10	Brot	5	0
itemID	name	retailprice	available																																					
1	Pizza	7	1																																					
2	Pommes	3.5	1																																					
3	Nudeln	11	1																																					
4	Lasagne	7.5	1																																					
5	Kaffee	2.3	1																																					
6	Cola	1.5	0																																					
9	Bier	3.3	0																																					
10	Brot	5	0																																					
Tatsächliches Testergebnis	<pre>[{"itemID":1,"name":"Pizza","retailprice":7.0,"quantity":4,"available":true}, {"itemID":2,"name":"Pommes","retailprice":3.5,"quantity":7,"available":true}, {"itemID":3,"name":"Nudeln","retailprice":11.0,"quantity":4,"available":true}, {"itemID":4,"name":"Lasagne","retailprice":7.5,"quantity":6,"available":true}, {"itemID":5,"name":"Kaffee","retailprice":2.3,"quantity":66,"available":true}, ...]</pre>																																							
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																																							

Erwartetes Testergebnis	Http-Nachricht über den Fehler.
Tatsächliches Testergebnis	<i>Whitelabel Error Page</i> <i>This application has no explicit mapping for /error, so you are seeing this as a fallback.</i> <i>Tue Nov 28 19:06:04 CET 2017</i> <i>There was an unexpected error (type=Not Found, status=404).</i> <i>Verbindung zur Datenbank fehlgeschlagen!</i>
Test bestanden	Normalablauf: Ja Sonderfall: Ja

Bestellungs-Daten

Anwendungsfall	Abfragen von allen Bestellungen aus der Datenbank.																																										
Verwendete Methode	<i>java.util.ArrayList<Order></i> <i>getAllOrders()</i> Durch das ansprechen des Pfades ".../api/orders" können die Bestellungen der Datenbank abgefragt werden.																																										
Normalablauf	Alle Bestellungen werden als Order-Objekte aus der MySQL-Datenbank abgefragt und über Http als JSON-Datei übertragen.																																										
Erwartetes Testergebnis	Rückgabe eines JSON-Textes mit folgenden Inhalten: <table><tr><th>orderID</th><th>itemIDs</th><th>price</th><th>date</th><th>tableID</th><th>paid</th></tr><tr><td>1</td><td>2;1;</td><td>2.3</td><td>2017-10-19 15:40:25</td><td>5</td><td>1</td></tr><tr><td>3</td><td>1;3;</td><td>18</td><td>2017-10-23 10:44:48</td><td>6</td><td>1</td></tr><tr><td>4</td><td>1;2;3;4;5;</td><td>31.3</td><td>2017-11-05 15:38:42</td><td>3</td><td>0</td></tr><tr><td>5</td><td>2;3;</td><td>9.8</td><td>2017-11-08 23:31:33</td><td>5</td><td>1</td></tr><tr><td>12</td><td>4;5;</td><td>9.8</td><td>2017-11-14 21:32:22</td><td>5</td><td>1</td></tr><tr><td>17</td><td>3;5;</td><td>33.99</td><td>2017-11-12 14:42:34</td><td>10</td><td>1</td></tr></table>	orderID	itemIDs	price	date	tableID	paid	1	2;1;	2.3	2017-10-19 15:40:25	5	1	3	1;3;	18	2017-10-23 10:44:48	6	1	4	1;2;3;4;5;	31.3	2017-11-05 15:38:42	3	0	5	2;3;	9.8	2017-11-08 23:31:33	5	1	12	4;5;	9.8	2017-11-14 21:32:22	5	1	17	3;5;	33.99	2017-11-12 14:42:34	10	1
orderID	itemIDs	price	date	tableID	paid																																						
1	2;1;	2.3	2017-10-19 15:40:25	5	1																																						
3	1;3;	18	2017-10-23 10:44:48	6	1																																						
4	1;2;3;4;5;	31.3	2017-11-05 15:38:42	3	0																																						
5	2;3;	9.8	2017-11-08 23:31:33	5	1																																						
12	4;5;	9.8	2017-11-14 21:32:22	5	1																																						
17	3;5;	33.99	2017-11-12 14:42:34	10	1																																						
Tatsächliches Testergebnis	<pre>[{"orderID":1,"itemIDs":"2;1;","tableID":5,"price":2.3,"date":1508420425000,"paid":true,"table":5,"items":"2;1;"}, {"orderID":3,"itemIDs":"1;3;","tableID":6,"price":18.0,"date":1508748288000,"paid":true,"table":6,"items":"1;3;"}, ...]</pre>																																										
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																																										
Erwartetes Testergebnis	Http-Nachricht über den Fehler.																																										
Tatsächliches	<i>Whitelabel Error Page</i>																																										

Testergebnis	<p><i>This application has no explicit mapping for /error, so you are seeing this as a fallback.</i></p> <p><i>Tue Nov 28 19:24:36 CET 2017</i></p> <p><i>There was an unexpected error (type=Not Found, status=404).</i></p> <p><i>Verbindung zur Datenbank fehlgeschlagen!</i></p>
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

Tisch-Daten

Anwendungsfall	Abfragen von Tisch-Daten aus der Datenbank.																											
Verwendete Methode	<i>java.util.ArrayList<Table></i> <i>getAllTables()</i> Durch das Ansprechen des Pfades ".../api/tables" können die Tische der Datenbank abgefragt werden.																											
Normalablauf	Alle Tische werden als Table-Objekte aus der MySQL-Datenbank abgefragt und über Http als JSON-Datei übertragen.																											
Erwartetes Testergebnis	Rückgabe eines JSON-Textes mit folgenden Inhalten: <table><tr><th>tableID</th><th>name</th><th>available</th></tr><tr><td>1</td><td>A1</td><td>1</td></tr><tr><td>2</td><td>A2</td><td>1</td></tr><tr><td>3</td><td>A3</td><td>1</td></tr><tr><td>4</td><td>B1</td><td>1</td></tr><tr><td>5</td><td>B2</td><td>1</td></tr><tr><td>6</td><td>B3</td><td>1</td></tr><tr><td>7</td><td>C1</td><td>1</td></tr><tr><td>8</td><td>C2</td><td>1</td></tr></table>	tableID	name	available	1	A1	1	2	A2	1	3	A3	1	4	B1	1	5	B2	1	6	B3	1	7	C1	1	8	C2	1
tableID	name	available																										
1	A1	1																										
2	A2	1																										
3	A3	1																										
4	B1	1																										
5	B2	1																										
6	B3	1																										
7	C1	1																										
8	C2	1																										
Tatsächliches Testergebnis	<i>[{"tableID":1,"name":"A1","available":true}, {"tableID":2,"name":"A2","available":true}, {"tableID":3,"name":"A3","available":true}, {"tableID":4,"name":"B1","available":true}, {"tableID":5,"name":"B2","available":true}, ...]</i>																											
Sonderfall	Keine Verbindung zur MySQL-Datenbank aufbaubar.																											
Erwartetes Testergebnis	Http-Nachricht über den Fehler.																											
Tatsächliches	<i>Whitelabel Error Page</i> <i>This application has no explicit mapping for /error, so you are seeing this as a fallback.</i>																											

Testergebnis	<i>Tue Nov 28 19:27:43 CET 2017</i> <i>There was an unexpected error (type=Not Found, status=404).</i> <i>Verbindung zur Datenbank fehlgeschlagen!</i>
Test bestanden	Normalablauf: Ja Sonderfall: Ja

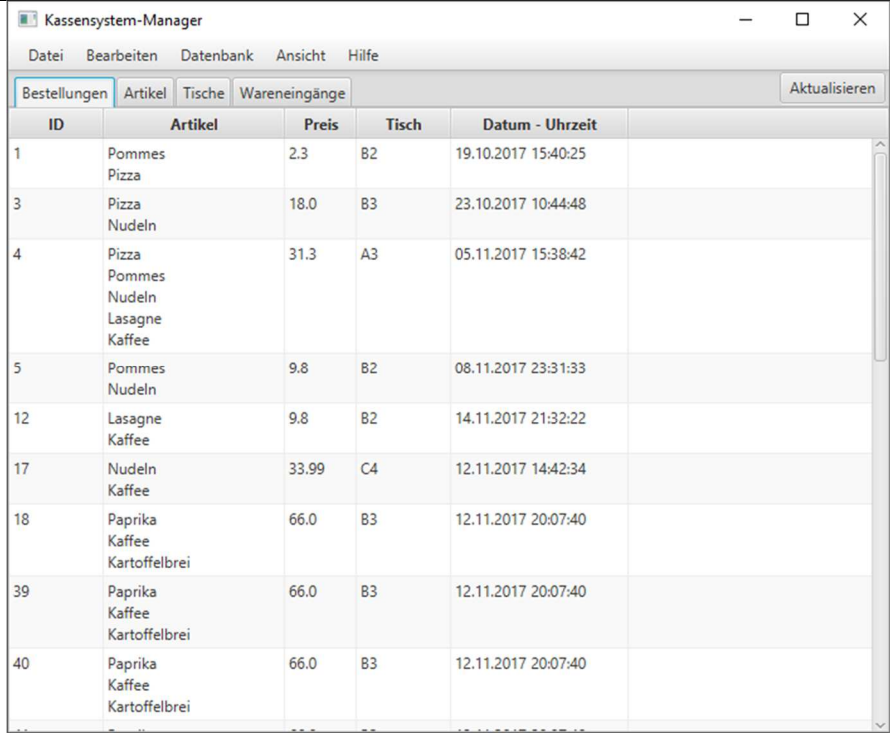
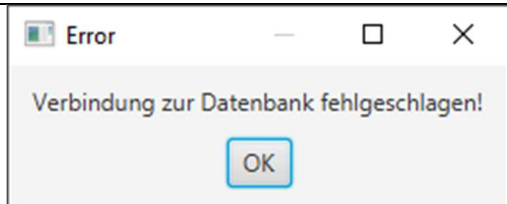
8.1.2.2 Aktualisieren und Hinzufügen einer Bestellung

Das Hinzufügen und Aktualisieren einer Bestellung kann nicht ohne weiteres ohne die Android-App getestet werden. Der Test dieser Funktionen wird mit den Tests der Android-App abgedeckt.

8.1.3 Benutzeranwendung

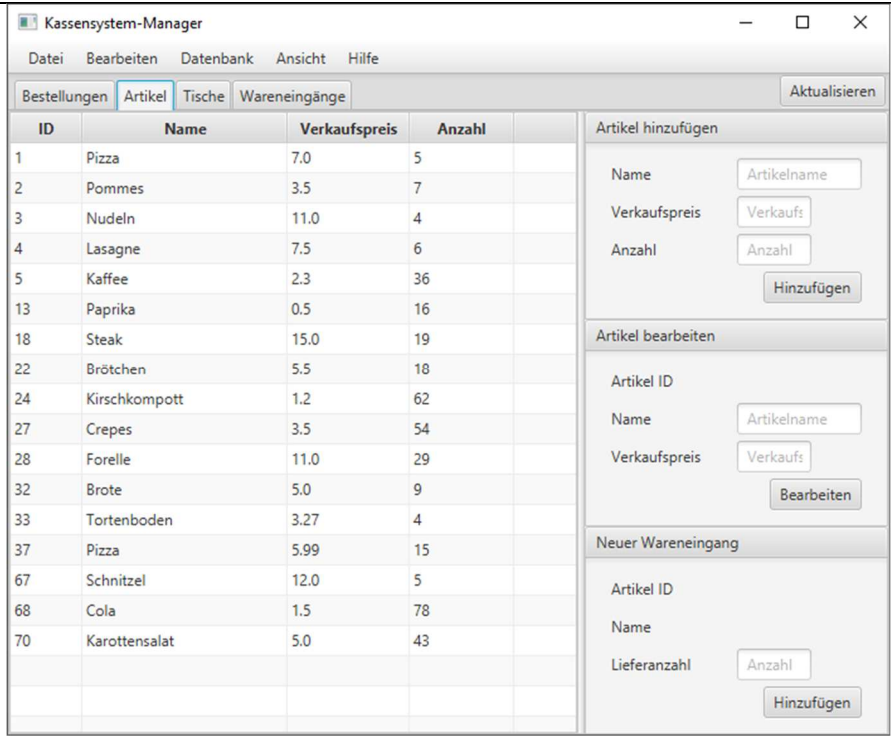
8.1.3.1 Abrufen von Datenbankinhalten

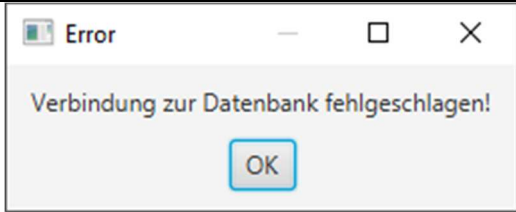
Abrufen der Bestellungen

Anwendungsfall	Einsehen aller Bestellungen in dem Kassensystem-Manager (AW 1)
Verwendete Methode	<code>void refreshOrderData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle nicht bezahlten Bestellungen in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller nicht bezahlten Bestellungen.
Tatsächliches Testergebnis	
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	

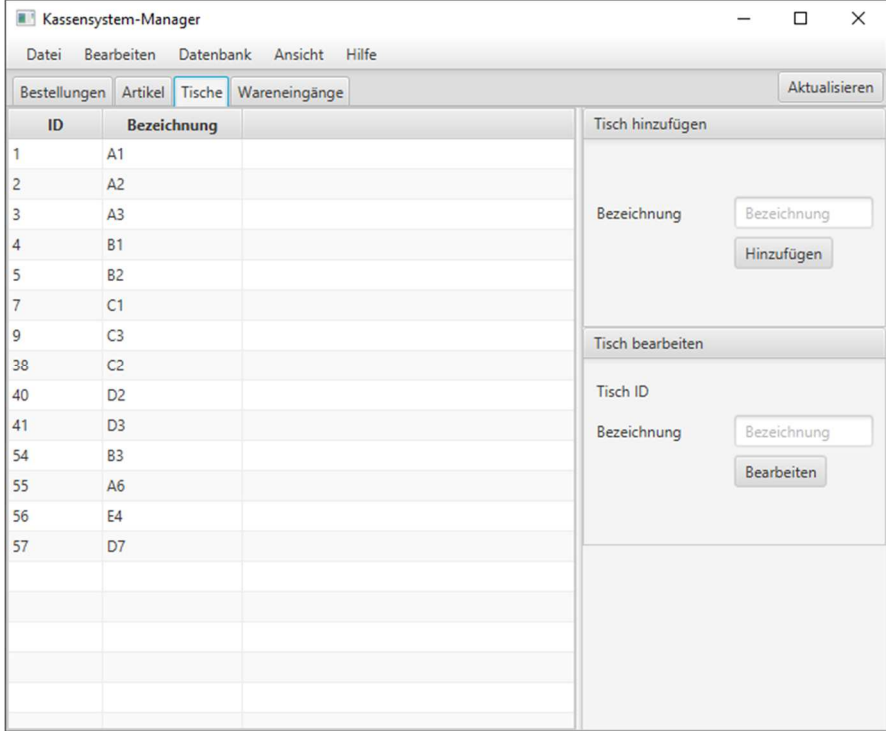
Test bestanden	Normalablauf: Ja Sonderfall: Ja
-------------------	--

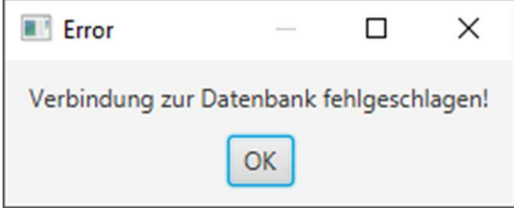
Abrufen der Artikel

Anwendungs- fall	Einsehen aller Artikel im Kassensystem-Manager (AW 3)
Verwendete Methode	<code>void refreshItemData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle verfügbaren Artikel in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller Artikel.
Tatsächliches Testergebnis	 <p>The screenshot shows the 'Kassensystem-Manager' application window. It has a menu bar with 'Datei', 'Bearbeiten', 'Datenbank', 'Ansicht', and 'Hilfe'. Below the menu is a tab bar with 'Bestellungen', 'Artikel' (selected), 'Tische', and 'Wareneingänge'. An 'Aktualisieren' button is in the top right. The main area contains a table with columns 'ID', 'Name', 'Verkaufspreis', and 'Anzahl'. The table lists 20 items including Pizza, Pommes, Nudeln, Lasagne, Kaffee, Paprika, Steak, Brötchen, Kirschkompott, Crepes, Forelle, Brote, Tortenboden, and Karottensalat. On the right sidebar, there are three sections: 'Artikel hinzufügen' with fields for Name, Verkaufspreis, and Anzahl; 'Artikel bearbeiten' with fields for Artikel ID, Name, and Verkaufspreis; and 'Neuer Wareneingang' with fields for Artikel ID, Name, and Lieferanzahl. Each section has a corresponding button (Hinzufügen or Bearbeiten).</p>
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.

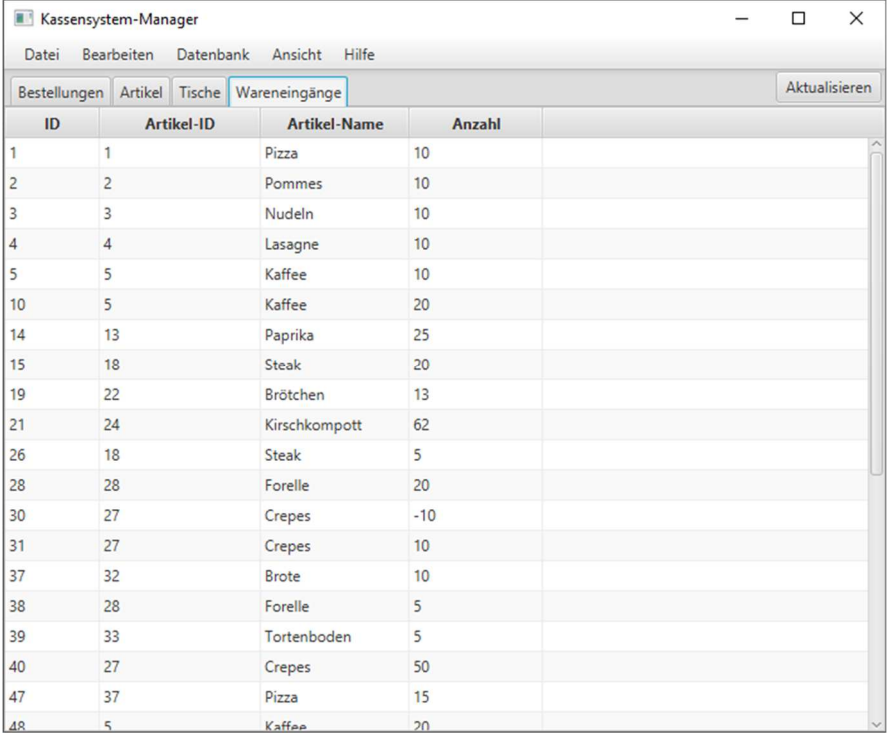
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

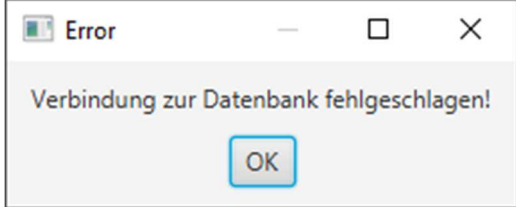
Abrufen der Tische

Anwendungsfall	Einsehen aller Tische im Kassensystem-Manager (AW 7)
Verwendete Methode	<code>void refreshTableData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle verfügbaren Tische in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller Tische.
Tatsächliches Testergebnis	
Sonderfall	Es besteht keine Verbindung zum MySQL-Server.

Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

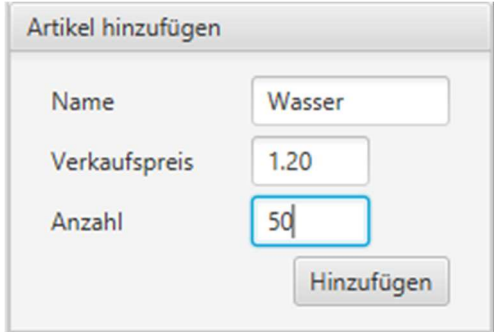
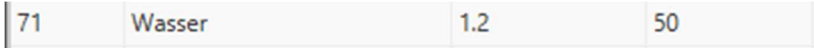
Abrufen der Wareneingänge

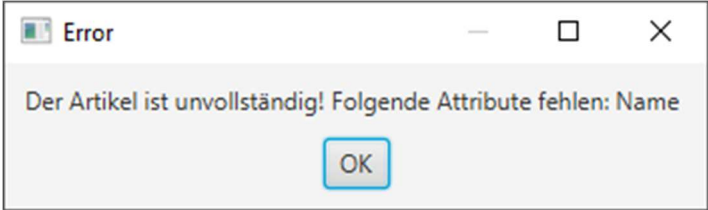
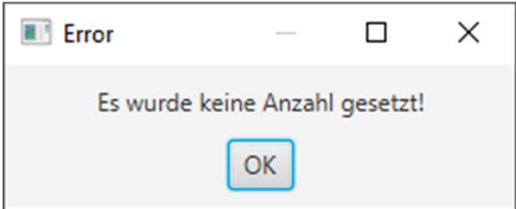
Anwendungsfall	Einsehen aller Wareneingänge in dem Kassensystem-Manager (AW 12)
Verwendete Methode	<code>void refreshItemdeliveryData()</code>
Normalablauf	Nach dem Öffnen der Anwendung werden alle Wareneingänge in einem Tab dargestellt.
Erwartetes Testergebnis	Eine tabellarische Darstellung aller Wareneingänge.
Tatsächliches Testergebnis	

Sonderfall	Es besteht keine Verbindung zum MySQL-Server.
Erwartetes Testergebnis	Es wird eine Fehlermeldung angezeigt.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

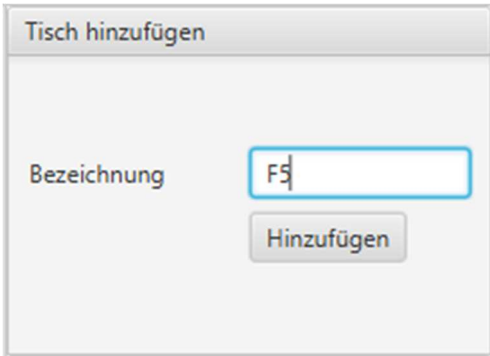
8.1.3.2 Hinzufügen von Datenbankinhalten

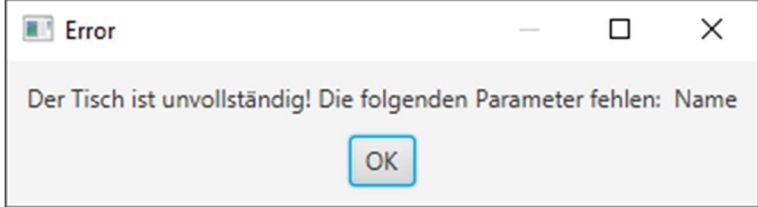
Hinzufügen von neuen Artikeln

Anwendungsfall	Hinzufügen eines neuen Artikels, der neu in das Sortiment/ die Speisekarte aufgenommen wurde (AW 5)
Verwendete Methode	<code>void addItem(ActionEvent actionEvent)</code>
Normalablauf	In die Felder werden die Daten des neuen Artikels eingegeben. Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.
Erwartetes Testergebnis	Ein neuer Artikel mit den folgenden Daten: 
Tatsächliches Testergebnis	
Sonderfall	3. Es wurde kein Name angegeben. 4. Es wurde keine Anzahl angegeben.



Erwartetes Testergebnis	<p>3. Anzeige einer Fehlermeldung mit einem fehlenden Namen.</p> <p>4. Anzeige einer Fehlermeldung mit einer fehlenden Anzahl.</p>
Tatsächliches Testergebnis	<p>1.</p>  <p>2.</p> 
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>

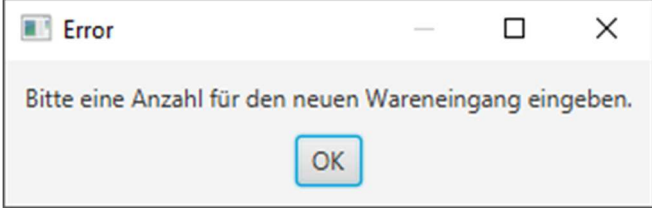
Hinzufügen von neuen Tischen

Anwendungsfall	Hinzufügen eines neuen Tisches, der neu im Geschäftsbereich eingerichtet wird (AW 9)
Verwendete Methode	<code>void addTable(ActionEvent actionEvent)</code>
Normalablauf	<p>Es werden in das Feld die Daten des neuen Tisches eingegeben.</p> <p>Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.</p>
Erwartetes Testergebnis	<p>Ein Tisch mit den folgenden Daten:</p> 
Tatsächliches	

Testergebnis	57 D7
Sonderfall	Es wurde kein Name angegeben.
Erwartetes Testergebnis	Fehlermeldung mit der Meldung eines fehlenden Namens.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

Hinzufügen von neuen Wareneingängen

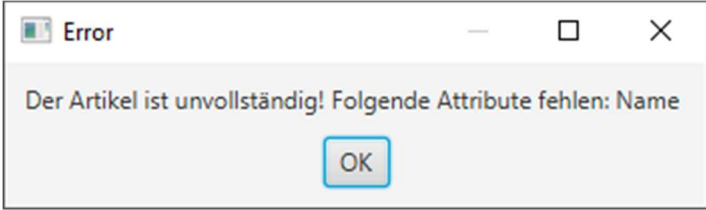
Anwendungsfall	Hinzufügen eines neuen Wareneingangs. Das wird während des Befüllens des Lagers gemacht (AW 13)
Verwendete Methode	<code>void addItemDelivery(ActionEvent actionEvent)</code>
Normalablauf	Ein Artikel wird angeklickt und anschließend die neue Anzahl eingegeben. Es wird der Datenbank ein neuer Eintrag hinzugefügt und anschließend in der tabellarischen Übersicht angezeigt.
Erwartetes Testergebnis	Ein neuer Wareneingang mit den folgenden Daten: 
Tatsächliches Testergebnis	
Sonderfall	Es wurde keine Anzahl eingegeben.

Erwartetes Testergebnis	Fehlermeldung mit der Meldung einer fehlenden Anzahl.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

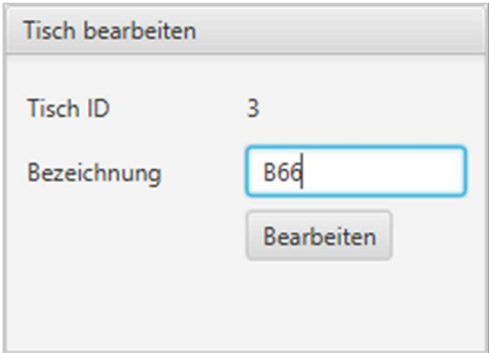
8.1.3.3 Bearbeiten von Datenbankinhalten

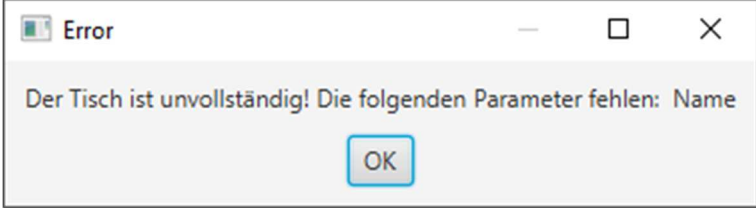
Bearbeiten von Artikeln

Anwendungsfall	Ändern der Daten eines Artikels, wie bspw. Preisänderung (AW 6)				
Verwendete Methode	<code>public void editItem(ActionEvent actionEvent)</code>				
Normalablauf	Ein Artikel wird angeklickt. Im entsprechenden Feld zum Bearbeiten des Artikels erscheinen die aktuellen Daten. Diese können bearbeitet werden. Wenn der „Bearbeiten“-Button gedrückt wird, wird der bisherige Artikel als nicht verfügbar markiert und ein neuer Datenbankeintrag mit den bearbeiteten Daten erzeugt. In der tabellarischen Übersicht wird der bearbeitete Artikel angezeigt.				
Erwartetes Testergebnis	<p>Der folgende Artikel soll aktualisiert werden:</p> <table><tr><td>3</td><td>Burger</td><td>4.5</td><td>57</td></tr></table> <p>Dieser Artikel soll mit den folgenden Daten aktualisiert werden:</p> <div><div>Artikel bearbeiten</div><div><div>Artikel ID</div><div>3</div></div><div><div>Name</div><div>VeggieBurger</div></div><div><div>Verkaufspreis</div><div>6.50</div></div><div>Bearbeiten</div></div>	3	Burger	4.5	57
3	Burger	4.5	57		
Tatsächliches Testergebnis	<p>In den Daten ist nun der folgende Artikel zu finden:</p> <table><tr><td>6</td><td>VeggieBurger</td><td>6.5</td><td>57</td></tr></table>	6	VeggieBurger	6.5	57
6	VeggieBurger	6.5	57		
Sonderfall	Es wird kein Name übergeben.				
Erwartetes Testergebnis	Eine Fehlermeldung, die einen fehlenden Namen anmerkt.				

Tatsächliches Testergebnis	
Test bestanden	<p>Normalablauf: Ja</p> <p>Sonderfall: Ja</p>


Bearbeiten von Tischen

Anwendungsfall	Ändern der Bezeichnung eines Tisches (AW 10)		
Verwendete Methode	<code>public void editTable(ActionEvent actionEvent)</code>		
Normalablauf	Ein Tisch wird angeklickt. Im entsprechenden Feld zum Bearbeiten des Tisches erscheinen die aktuellen Daten. Diese können bearbeitet werden. Wenn der „Bearbeiten“-Button gedrückt wird, wird der bisherige Tisch als nicht verfügbar markiert und ein neuer Datenbankeintrag mit den bearbeiteten Daten erzeugt. In der tabellarischen Übersicht wird der bearbeitete Tisch angezeigt.		
Erwartetes Testergebnis	<p>Der folgende Tisch soll aktualisiert werden:</p> <table border="1" data-bbox="450 1355 831 1411"> <tr> <td>3</td> <td>A3</td> </tr> </table> <p>Dieser Tisch soll mit den folgenden Daten bearbeitet werden:</p> 	3	A3
3	A3		
Tatsächliches Testergebnis	In der tabellarischen Übersicht ist nun der folgende Tisch zu finden:		

	7 B66
Sonderfall	Es wurde kein Name angegeben.
Erwartetes Testergebnis	Eine Fehlermeldung über den fehlenden Namen.
Tatsächliches Testergebnis	
Test bestanden	Normalablauf: Ja Sonderfall: Ja

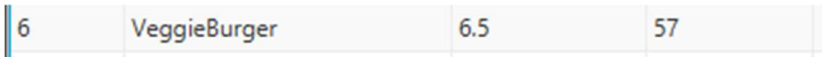
8.1.3.4 Löschen von Datenbankinhalten

Löschen von Bestellungen


Anwendungsfall	Löschen einer fehlerhaften oder überschüssigen Bestellung (AW 2)
Verwendete Methode	<code>public void deleteOrder(ActionEvent actionEvent)</code>
Normalablauf	Eine Bestellung wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag der Bestellung verschwindet aus der tabellarischen Übersicht und ist nicht mehr in der Datenbank zu finden.
Erwartetes Testergebnis	Der Eintrag der Bestellung wird aus der tabellarischen Übersicht entfernt. Die folgende Bestellung soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Artikeln

Anwendungsfall	Löschen eines Artikels der aus dem Sortiment genommen wurde (AW 4)
Verwendete Methode	<code>public void deleteItem(ActionEvent actionEvent)</code>

Normalablauf	Ein Artikel wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag der Bestellung verschwindet aus der tabellarischen Übersicht und in der Datenbank wird der Artikel als nicht verfügbar markiert.
Erwartetes Testergebnis	Der folgende Artikel soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner
Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Tischen

Anwendungsfall	Löschen eines Tisches, der von der Verkaufsfläche entfernt wurde (AW 8)
Verwendete Methode	<i>public void deleteTable(ActionEvent actionEvent)</i>
Normalablauf	Ein Tisch wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag des Tisches verschwindet aus der tabellarischen Übersicht und in der Datenbank wird der Tisch als nicht verfügbar markiert.
Erwartetes Testergebnis	Der folgende Tisch soll gelöscht werden: 
Tatsächliches Testergebnis	Der Eintrag existiert nicht mehr in der Anwendung.
Sonderfall	keiner

Erwartetes Testergebnis	--
Tatsächliches Testergebnis	--
Test bestanden	Normalablauf: Ja Sonderfall: --

Löschen von Wareneingängen

Anwendungsfall	Ein Wareneingang soll gelöscht werden, der bspw. fälschlicherweise angelegt wurde. (AW 14)								
Verwendete Methode	public void deleteItemdelivery(ActionEvent actionEvent)								
Normalablauf	Ein Wareneingang wird über einen Rechtsklick ausgewählt und der Menüeintrag zum Löschen ausgewählt. Der Eintrag des Wareneingangs verschwindet aus der tabellarischen Übersicht und ist nicht mehr in der Datenbank zu finden.								
Erwartetes Testergebnis	<p>Der folgende Wareneingang soll gelöscht werden:</p> <table><tr><td>9</td><td>5</td><td>Cola</td><td>50</td></tr></table> <p>In der tabellarischen Übersicht der Artikel wird die Anzahl des entsprechenden Artikels reduziert:</p> <table><tr><td>5</td><td>Cola</td><td>2.5</td><td>212</td></tr></table>	9	5	Cola	50	5	Cola	2.5	212
9	5	Cola	50						
5	Cola	2.5	212						
Tatsächliches Testergebnis	<p>Der Eintrag existiert nicht mehr in der Anwendung. Die Anzahl des Artikels in der Artikel-Übersicht wurde aktualisiert:</p> <table><tr><td>5</td><td>Cola</td><td>2.5</td><td>162</td></tr></table>	5	Cola	2.5	162				
5	Cola	2.5	162						
Sonderfall	keiner								
Erwartetes Testergebnis	--								
Tatsächliches Testergebnis	--								

Test bestanden	Normalablauf: Ja Sonderfall: --
-------------------	------------------------------------

8.1.3.5 Ausdrucken einer Bestellung

Anwendungsfall	Nachträgliches Ausdrucken eines Belegs, nachdem der ursprüngliche Beleg verloren gegangen oder zerstört bzw. verschmutzt wurde (AW 12)																				
Verwendete Methode	<code>public void printOrder(ActionEvent actionEvent)</code>																				
Normalablauf	Über einen Rechtsklick wird eine Bestellung ausgewählt und der Menüeintrag zum Ausdrucken angeklickt. Über den Bon-Drucker wird der Kundenbeleg ausgedruckt.																				
Erwartetes Testergebnis	Der Bon-Drucker druckt einen Kundenbeleg mit den folgenden Bestelungsdaten: <div><table><tr><td>7</td><td>Brot</td><td>14.5</td><td>A1</td><td>03.12.2017 19:51:12</td></tr><tr><td></td><td>Cola</td><td></td><td></td><td></td></tr><tr><td></td><td>Cola</td><td></td><td></td><td></td></tr><tr><td></td><td>Burger</td><td></td><td></td><td></td></tr></table></div>	7	Brot	14.5	A1	03.12.2017 19:51:12		Cola					Cola					Burger			
7	Brot	14.5	A1	03.12.2017 19:51:12																	
	Cola																				
	Cola																				
	Burger																				
Tatsächliches Testergebnis	Ein Ausdruck wurde ausgegeben: <div><p>-----Kundenbeleg-----</p><p>Restaurante Gaumenfreude Gourmetstraße 11 12345 Leckerschmeckerhausen +49 541 466 655</p><p>Ihre Bestellung:</p><table><tr><td>Brot</td><td>5,00 EUR</td></tr><tr><td>Cola</td><td>2,50 EUR</td></tr><tr><td>Cola</td><td>2,50 EUR</td></tr><tr><td>Burger</td><td>4,50 EUR</td></tr></table><hr/><table><tr><td>Summe</td><td>14,50 EUR</td></tr><tr><td>inkl. MWST 19%</td><td>2,89 EUR</td></tr></table><p>Sie saßen an Tisch A1. Vielen Dank für Ihren Besuch! 03.12.2017 19:51:12</p></div>	Brot	5,00 EUR	Cola	2,50 EUR	Cola	2,50 EUR	Burger	4,50 EUR	Summe	14,50 EUR	inkl. MWST 19%	2,89 EUR								
Brot	5,00 EUR																				
Cola	2,50 EUR																				
Cola	2,50 EUR																				
Burger	4,50 EUR																				
Summe	14,50 EUR																				
inkl. MWST 19%	2,89 EUR																				
Sonderfall	Der Drucker ist nicht angeschlossen oder abgeschaltet.																				
Erwartetes Testergebnis	Der Beleg wird ausgedruckt, sobald der Drucker erreichbar ist.																				
Tatsächliches Testergebnis	Nach dem Abschalten und wieder Anschalten des Drucker wird der Beleg wie erwartet ausgedruckt.																				
Test bestanden	Normalablauf: Ja Sonderfall: Ja																				