

Modul 326 - KKK

Projekt - KKK

Fabian Kurt

2018

Inhaltsverzeichnis

[Projektbeschrieb 2](#_Toc510871784)

[Wie starte ich das Projekt 3](#_Toc510871785)

[Use Case 4](#_Toc510871786)

[Klassendiagramm 5](#_Toc510871787)

[Kontextdiagramm 6](#_Toc510871788)

[Anforderungen 7](#_Toc510871789)

[Funktionale Anforderungen 7](#_Toc510871790)

[Nicht funktionale Anforderungen 7](#_Toc510871791)

[Verwendete Libraries 8](#_Toc510871792)

[Java 1.8 8](#_Toc510871793)

[groovy 2.4.15 8](#_Toc510871794)

[mysql connection java 5.1.43 8](#_Toc510871795)

[Testszenarien 9](#_Toc510871796)

[Implementierungsentscheidung 10](#_Toc510871797)

[In Bezug auf Abstrakte Klassen 10](#_Toc510871798)

[In Bezug auf Klassenhierarchien 10](#_Toc510871799)

[In Bezug auf Interfaces 10](#_Toc510871800)

[Zugriff auf das Datenbanksystem 11](#_Toc510871801)

[Datensicherheit & Applikationssicherheit 12](#_Toc510871802)

[Welche Fehlerzustände können auftreten 12](#_Toc510871803)

[Fazit 12](#_Toc510871804)

# Projektbeschrieb

Ich habe den Auftrag erhalten ein Projekt/Applikation zu entwickeln die verschiedene Anforderungen erfüllen muss. Zuerst musste ich mich für eine Sprache entscheiden und ich bin zum Entschluss gekommen das ich Java brauche. Danach war nur noch die Frage was für eine Anwendung ich Entwickeln werde und ich kam auf die Idee ein Kartei Kärtchen System zu entwickeln.

Ich habe das Programm wie folgt umgesetzt:  
Wen man das Programm startet befindet man sich zuerst im Startmenu. da hat man drei Auswahlmöglichkeiten.

1.“Üben“: Man kann eine Kartei auswällen und dann mit dem Üben beginnen. Es wird eine Frage gestellt und man kann darauf antworten und dann auf Prüfen drücken. Wen man die richtige Antwort eingegeben hat, erscheint eine MessageBox mit der Meldung dass die Antwort richtig sei. Wen man aber jetzt eine falsche Antwort eingegeben hat erscheint auch eine MessageBox und gibt diesmal aus das die Antwort falsch sei und was die richtige Antwort sei. Danach kommt die nächste Frage und ganz am Schluss kommt noch mal eine MessageBox die anzeigt wie viele Antworten man richtig hat und wie viele Antworten man falsch. Danach gelingt man wieder zum GUI wo man eine Kartei auswällen kann zum Üben.

2.“Bearbeiten“: Im GUI-Bearbeiten kann man: „Erstellen“ -> Man kann eine neue Kartei erstellen. Zuerst definiert man einen Namen für die Kartei. Danach kann man Kärtchen der Kartei hinzufügen. Eine Frage und eine Antwort, danach drückt man hinzufügen und kann beliebig viele Kärtchen hinzufügen. wen man fertig ist drückt man auf Fertig und man kommt wieder zurück zum GUI-Bearbeiten. Wen man eine Kartei erweitern möchte kann man sie mit der Combobox auswällen und dann bearbeiten oder wen man Sie nicht mehr brauch kann man sie auch löschen. Aber um eine Kartei nur anzuzeigen genügt es schon mit der ComboBox die gewünschte Kartei auszuwählen und dann werden alle Kärtchen angezeigt. Wen man nichts mehr bearbeiten möchte kann man auf Abbrechen drücken und geling wider zum Startmenü.

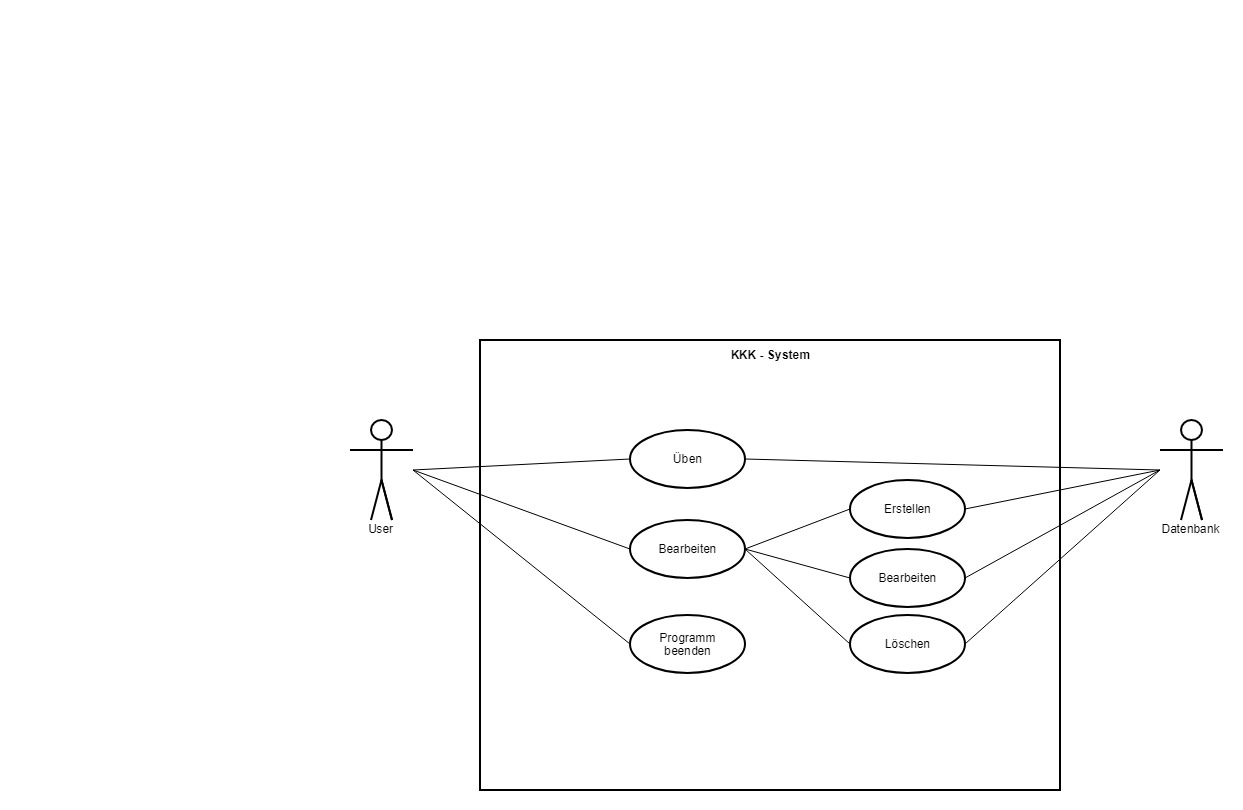
3. “Beenden“: Damit kann man die Anwendung schliessen und man wird vorher noch gefragt ob man die Anwendung wirklich beenden möchte für den Fall das man nur aus Versehen auf den Beenden Button gekommen ist. Wen man sich sicher ist kann man mit OK bestätigen und dann schliesst sich das Programm.

Ausserdem habe ich für die Datenhaltung mit MySql gearbeitet.

# Wie starte ich das Projekt

1. Führen Sie das create.sql in der MySql-Workbench aus
2. Danach öffnen Sie das Projekt in Intellij
3. In Intellij führen Sie dann die Main Methode aus.

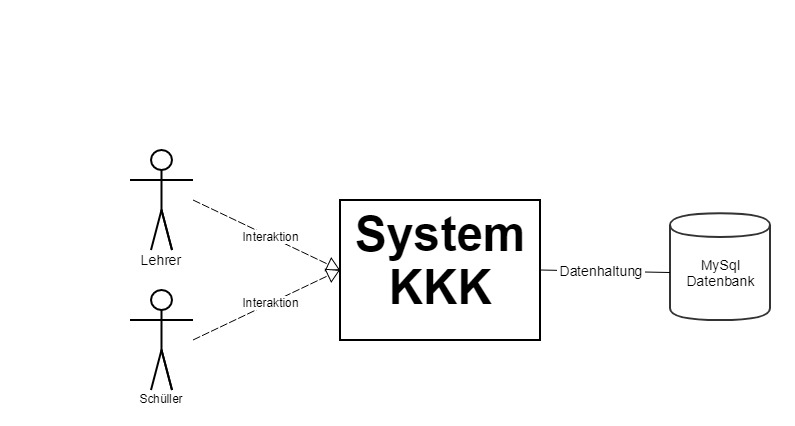
# Use Case



# Klassendiagramm

<KlassenDiagram.pdf>

# Kontextdiagramm



# Anforderungen

## Funktionale Anforderungen

|  |
| --- |
| **Programm / Applikation** |
| Fehlerlogging mittels Output-Stream korrekt umgesetzt |
| Korrekte Implementierung & Verwendung des Erzeugungsmusters: Singleton |
| Korrekte Implementierung & Verwendung des Erzeugungsmusters: Factory |
| Korrekte Implementierung & Verwendung des Verhaltensmuster: Observer |
| Korrekte Implementierung des Architekturmusters: Model-View-Controller / oder MVVM |
| Verwendung von mindestens 3 selbst erstellten Interfaces korrekt und sinngemäss für das Projekt |
| Verwendung von mindestens 3 abstrakten Klassen korrekt und sinngemäss für das Projekt |
| Verwendung von mindestens 3 Basisklassen korrekt und sinngemäss für das Projekt |
| Abstrahierung der Datenbankschnittstelle auf einzelne Entitäten korrekt und sinngemäss umgesetzt (Siehe Arbeitsblatt 02 – Datenbankanbindung) |
| Programm ist Lauffähig und stürzt nicht ab |
| Testszenarien gemäss Dokumentation implementiert und sind Funktionsbereit |
| Umsetzung mindesten zweier Entitäten mittels CRUD-Operationen |
| GUI ansprechend und sinngemäss gestaltet und umgesetzt |

## Nicht funktionale Anforderungen

|  |
| --- |
| **Programm / Applikation** |
| Applikation wird auf allen Desktopgeräten von 1280x1024 – 1920x1080 korrekt dargestellt und kann frei vergrössert & verkleinert werden |

# Verwendete Libraries

## Java 1.8



Wird verwendet für das Programmieren der Applikation.

## groovy 2.4.15



Wird verwendet für das Testen (JUnit-Tests) der Applikation.

## mysql connection java 5.1.43



Wird verwendet um eine Verbindung zur Datenbank herzustellen und mit ihr zu arbeiten.

# Testszenarien

|  |  |
| --- | --- |
| Testszenarien | |
| Testszenario 01 | Programm startet und Stürzt nicht ab |
| Testszenario 02 | Man kann zwischen den einzelnen Scenen wechseln ohne dass das Programm abstürzt. |
| Testszenario 03 | Man kann den die CRUD-Funktionalität(Button) Update(Bearbeiten) und Delete(Löschen) im der Scene ArbeitenAuswählen nur benutzten wen man eine Kartei ausgewählt hat. |
| Testszenario 04 | Man kann nur eine Übung starten wen man eine Kartei auswählt. |
| Testszenario 05 | Der User kann sich die Karteien anzeigen lassen(immer nur sie Ausgewählte). |
| Testszenario 06 | Der User kann eine neue Kartei Erstellen. |
| Testszenario 07 | Der User kann einer Kartei einen Namen geben. |
| Testszenario 08 | Einer neu erstellten Kartei können Karten hinzugefügt werden. |
| Testszenario 09 | Einer schon erstellten Kartei können später noch beliebig oft Karten hinzugefügt werden. |
| Testszenario 10 | Eine Kartei samt allen dazugehörigen Karten können gelöscht werden |
| Testszenario 11 | Alle benötigten Daten(aus MySql-Server) können korrekt aus/eingelesen werden. |
| Testszenario 12 | Die Frage wird angezeigt und nach der Beantwortung wird die nächste angezeigt wen vorhanden. |
| Testszenario 13 | Dem User wird angezeigt(MessageBox), wen er die richtige Antwort eingegeben hat. |
| Testszenario 14 | Dem User wird angezeigt(MessageBox) das die Antwort falsch sei und was die Richtige Antwort wäre, wen er eine falsche Antwort eingegeben hat. |
| Testszenario 15 | Am Ende einer Übung wird dem User angezeigt wie viel er richtig und wie viel er falsch hatte. |

# Implementierungsentscheidung

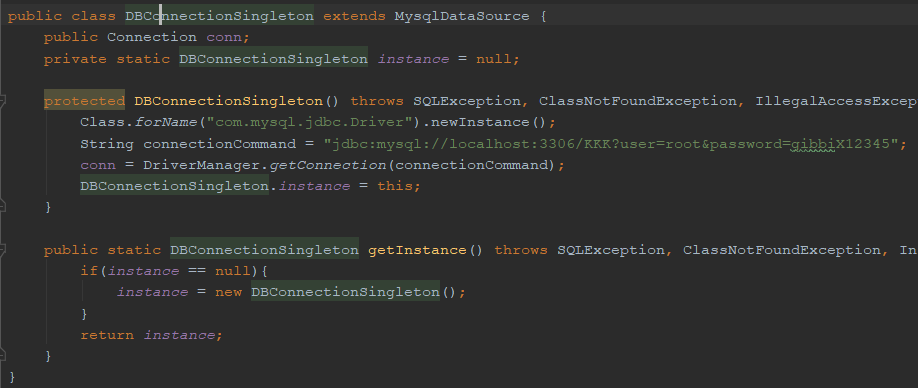
## In Bezug auf Abstrakte Klassen

## In Bezug auf Klassenhierarchien

## In Bezug auf Interfaces

# Zugriff auf das Datenbanksystem

**2.**



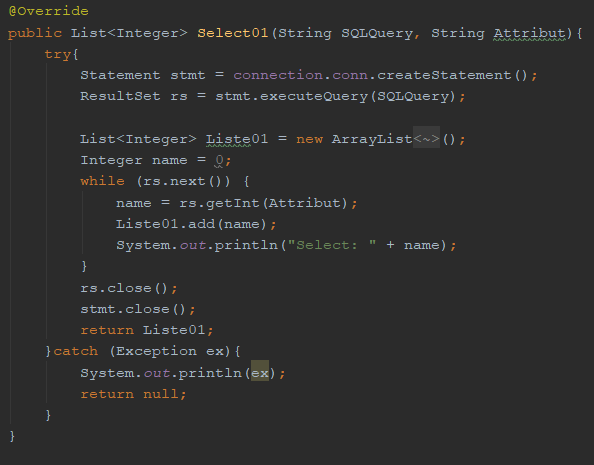
**1.**

**4.**

**3.**



**5.**



**10.**

**9.**

**8.**

**7.**

**6.**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | Es wird eine Klasse instanziiert mit einem Methoden Aufruf der eine Variable festlegt. | 6. | Es wird ein Statement deklariert und initialisiert mit der conn(Datenbank Verbindung) Variable. |
| 2. | Es werden die nötigen Treiber geladen für eine DBConnektion. | 7. | Der Resultset wird deklariert und mir der Ausgabe von dem ausgeführten stmt initialisiert. |
| 3. | Eine String Variable wird deklariert und Initialisiert mit allen wichtigen Angaben für eine Verbindung zur Datenbank | 8. | Eine String Liste „Liste01“ und eine Integer Variable „name“ werden deklariert und Initialisiert. |
| 4. | Die Instanz die = null ist, bekommt den Wert der aktuellen Instanz | 9. | In der schleifen werden alle werte von rs in die Liste „Liste01“ geladen. |
| 5. | Die Methode Slecet01 wird aufgerufen. | 10. | Beendet rs & stmt und gib Liste01 zurück. |

# Datensicherheit & Applikationssicherheit

Habe keine Datensicherheit & Applikationssicherheit integriert weil es gibt keine schützenswerte Daten. es sind alle für jeden zugänglich.

# Welche Fehlerzustände können auftreten

Keine da ich der Coode der das Programm zumabstürtztrn etc. füren könnte abgesichert habe. Entweder mit einem Try/catch oder mit throws.

# Fazit