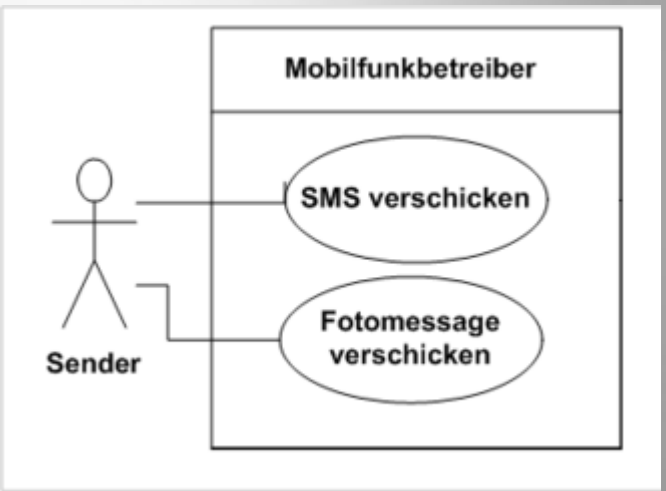
Software Engineering I

Patrick Steinhauer
Jan Dennis Bartels



Professor
Stefan Sarstedt

Assistent
Norbert Kasperczyk-Borgmann

Aufgabenblatt Nummer 3

10.12.2014

Inhaltsverzeichnis

User Story 1
Use Case : 2
3.2 Systemoperationen und Schnittstellenkategorie A 3
3.3 Sequenzdiagramm 3
3.4 Code 4
3.41 Ilernkartenkomponentenservices 4
3.42 Lernkartenkomponente 6
3.43 LernkartenKomponenteDAO 9
3.44 INutzerKomponentenServices 13
3.45 NutzerKomponente 13
3.46 INutzerKomponenteDAO 14
3.47 NutzerKomponenteDAO 15
3.48 LernkartenKomponentenTest 17
3.49 NutzerKomponentenTest 20
4.0 SQL Schema 22
4.1 Erstellung der DB 22
4.2 Inserts in die DB 24

Patrick Steinhauer
Jan Dennis Bartels

User Story

Ich als Student :

- Kann persönliche Lernkarten erstellen.
- Kann Lernkarten bearbeiten und verändern.
- Kann Lernkarten abspeichern.
- Kann Lernkarten anschauen.
- Kann Antworten zu Lernkarten abgeben.
- Kann Lernkarten zur Übung bearbeiten.
- Kann Prüfungen bearbeiten.
- Kann Antworten Hochladen und abspeichern.
- Kann Lernkarten herunterladen.
- Kann mir meinen Lernfortschritt anzeigen lassen.

Ich als Professor / Assistent :

- Kann Lernkarten erstellen und bearbeiten.
- Kann Antworten für eine Lernkarte überprüfen und korrigieren.
- Kann mir die Lernerfolge der Studierenden anschauen.
- Kann die Liste mit den Studierenden des Moduls anzeigen lassen.
- Kann sehen, welche Lernkarten ein Student bearbeitet hat.
- Kann Prüfungen erstellen.
- Kann eigene Antworten zu Lernkarten erstellen.

Ich als Administrator :

- Kann Studenten und Professoren freischalten.
- Kann Module für lernkarten vorgeben.
- Kann Lernkarten erstellen, bearbeiten.
- Kann Antworten für Lernkarten abgeben.
- Kann Antworten prüfen.

Patrick Steinhauer
Jan Dennis Bartels

Use Case :

Titel : Lernkarte anschauen
Akteur : Professor
Ziel : Der Professor schaut sich die Lernkarte eines Studenten an

Auslöser:
Student hat eine Lernkarte erstellt.

Vorbedingungen :
Benutzer muss im System angemeldet sein.
Der Student hat eine Lernkarte erstellt.

Nachbedingungen :
Die Lernkarte wurde angeschaut.

Erfolgsszenario :

1. Das System zeigt das Startmenü der SOLE Plattform an.
2. Der Professor navigiert zu dem Bereich „Neue Lernkarten“.
3. Das System zeigt eine Modulübersicht an.
4. Der Professor wählt ein Modul aus.
5. Das System zeigt alle Lernkarten an.
6. Der Professor wählt eine der vorhandenen neuen Lernkarten aus.
7. Das System zeigt die Lernkarte an.
8. Der Professor schaut sich die Lernkarte an.
9. Der Professor klickt auf bestätigen.

Fehlerfälle :

9.1. Der Professor bestätigt versehentlich die Korrektur. Es wurde keine Eingabe getätigt. Das System weist den Nutzer daraufhin.
9.2. Bei Falscher Korrektur soll der Student die Möglichkeit haben dies zu melden und erneut die Möglichkeit haben die Lernkarte in den Bereich zu korrigieren zu verschieben.

Häufigkeit :

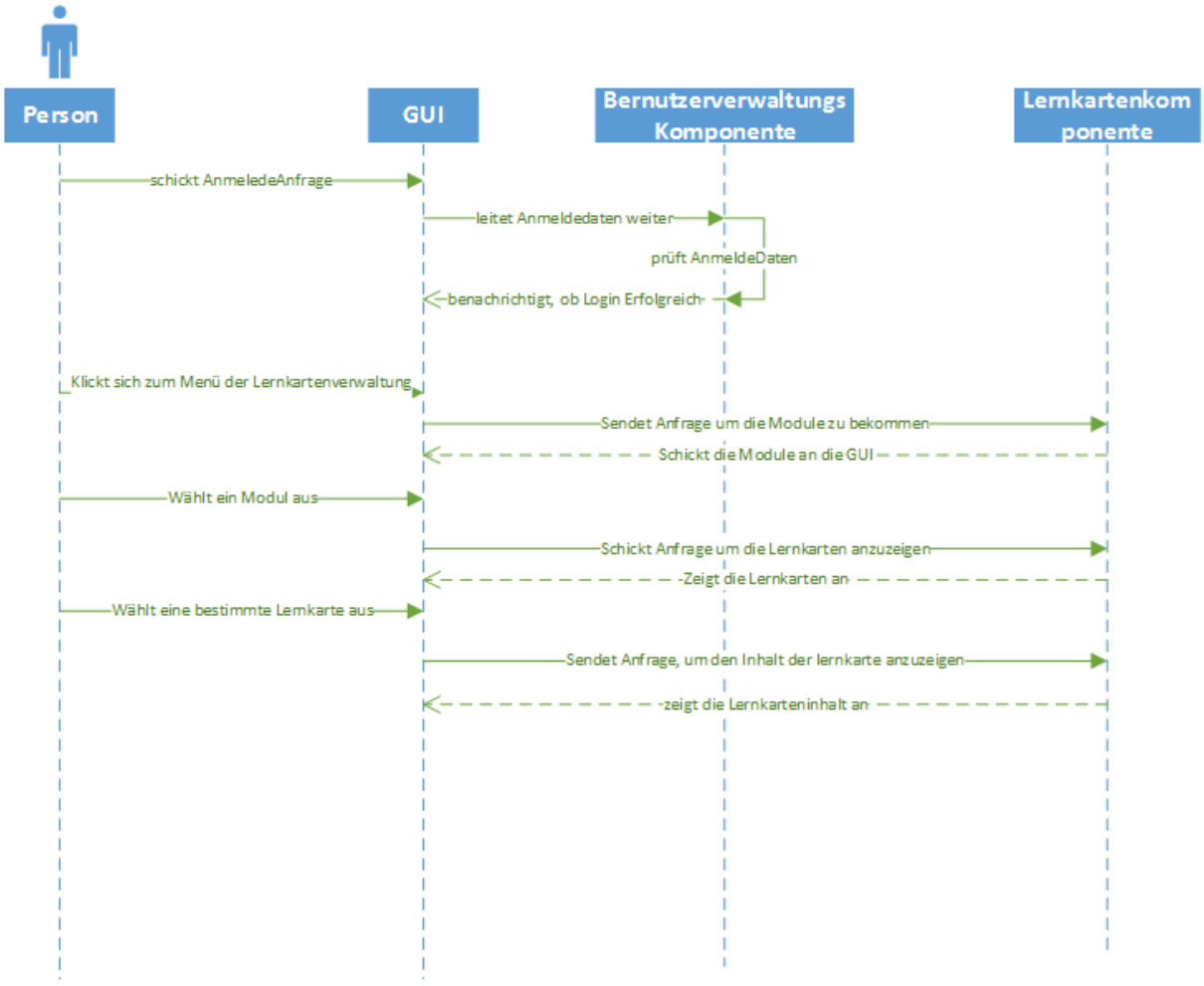
Der Professor macht dieses 20-30 mal die Woche.

3.2 Systemoperationen und Schnittstellenkategorie A

a.) Aus den User Stories:

- ILernkarte erstelleLernkarte(ILernkarte lernkarte)
- ILernkarte ändereLernkarte(ILernkarte lernkarte)
- List<ILernkarte> speichereLernkarte(ILernkarte lernkarte)
- Map<IFrage, IAntwort> erstelleAntwortZuLernkarte(ILernkarte lernkarte)
- List<ILernkarten> zeigeBearbeiteteLernkartenAn(ILernkarten lernkarten)
- ILernkarte korrigiereLernkarte(ILernkarte lernkarte)
- IModul zeigeModulUebersichtMitLernkartenAn(List<ILernkarten> lernkarten)
- List<ILernkarte> nichtKorrigierteLernkarten(List<ILernkarte> nichtKorrigiertelernkarten)
- Lernkarte zeigeLernkarteAn(ILernkarte lernkarte)

3.3 Sequenzdiagramm



3.4 Code

3.41 Ilernkartenkomponentenservices

```
package lernkartenKomponente;

import nutzerKomponente.INutzer;

import java.util.List;

/**
 * Created by patrick_steinhauer on 10.11.2014.
 */
public interface ILernkartenKomponenteServices {
    /**
     *
     * @param modulName
     * @param modulBeschreibung
     * @return Es wird eine Modul zurückgegeben, wenn eins erstellt wurde.
     */
    IModul erstelleModul(String modulName, String modulBeschreibung);
    /**
     *
     * @return Hier wird eine Liste mit vorhandenen Modulen aus der Datenbank zurückgegeben.
     */
    List<IModul> getModulListe();
    /**
     *
     * @return Hier wird eine Liste mit den vorhandenen lernkarten aus der Datenbank zurückgegeben.
     */
```

```
    List<ILernkarte> getListeDerLernkarten();

    /**

     *

     * @param modul

     * @param lernkartenName

     * @param aufgabenTyp

     * @return Hier wird eine Lernkarte zurückgegeben, wenn eine erstellt  wurde.

     */

    ILernkarte erstelleLernkarte(IModul modul, String lernkartenName, AufgabenTyp
aufgabenTyp);

    /**

     * Noch nicht implementiert.

     * @return Hier soll später eine Liste der unkorrigierten Lernkarten zurückgegeben
werden.

     */

    List<ILernkarte> getListeDerUnkorrigiertenLernkarten();

    /**

     *

     * @param lernkarte

     * @return Diese Methode soll später eine erstellte lernkarte in die Datenbank
abspeichern.

     */

    int speichereLernkarte(ILernkarte lernkarte);

    /**

     *

     * @param LernkartenName

     * @return Anhand des lernkartennamens soll hier eine Lernkarte zurückgegeben werden.

     */

    ILernkarte getLernkarte(String LernkartenName);

}
```

3.42 Lernkartenkomponente

```
package lernkartenKomponente;


import antwortKomponente.IAntwortKomponenteServices;

import com.sun.org.apache.xpath.internal.SourceTree;

import nutzerKomponente.INutzer;

import nutzerKomponente.INutzerKomponenteServices;

import persistenz.IPersistenzServices;


import java.util.ArrayList;

import java.util.Collection;

import java.util.List;


/**

 * Created by patrick_steinhauer on 10.11.2014.

 */

public class LernkartenKomponente implements ILernkartenKomponenteServices {

    private IAntwortKomponenteServices antwortKomponentenServices;

    private INutzerKomponenteServices nutzer;

    private List<ILernkarte> lernkarte;

    private List<IModul> vorhandeneModule;

    private SingleChoice singleChoice;

    private LernkartenkomponenteDAO lkDAO;


    public LernkartenKomponente(IPersistenzServices persistenzServices,
IAntwortKomponenteServices antwortKomponentenServices) {

        this.antwortKomponentenServices = antwortKomponentenServices;

        this.lkDAO = LernkartenkomponenteDAO.getInstance();

    }
```

```
    @Override
    public IModul erstelleModul(String modulName, String modulBeschreibung) {
        Modul modul = new Modul();
        modul.setModulName(modulName);
        modul.setModulBeschreibung(modulBeschreibung);
        return modul;
    }
    @Override
    public List<IModul> getModulListe() {
        return lkDAO.getModulListe();
    }
    @Override
    public List<ILernkarte> getListeDerLernkarten() {
        return lkDAO.getListeDerLernkarten();
    }
    @Override
    public ILernkarte erstelleLernkarte(IModul modul, String lernkartenName, AufgabenTyp
aufgabenTyp) {
        Lernkarte lernkarte = new Lernkarte();
//      lernkarte.setLernkartenName(lernkartenName);
        return lernkarte;
    }
    @Override
    public List<ILernkarte> getListeDerUnkorrigiertenLernkarten() {
        return null;
    }
    @Override
```

```
    public int speichereLernkarte(ILernkarte lernkarte) {
        //TODO JDBC machen speichert lernkarte und gibt einen integer mit der id zurück für den user.
        return 0;
    }
    @Override
    public ILernkarte getLernkarte(String lernkartenName) {
        return lkDAO.getLernkarte(lernkartenName);
    }
    @Override
    public String toString() {
        return "LernkartenKomponente{" +
                "antwortKomponentenServices=" + antwortKomponentenServices +
                ", nutzer=" + nutzer +
                ", lernkarte=" + lernkarte +
                ", singleChoice=" + singleChoice +
                '}';
    }
}
```

3.43 LernkartenKomponenteDAO

```
package lernkartenKomponente;


import antwortKomponente.AntwortKomponente;

import nutzerKomponente.*;

import persistenz.IPersistenzServices;

import persistenz.PersistenzServices;

import java.beans.Statement;

import java.sql.*;

import java.util.ArrayList;

import java.util.List;

/**

 * Created by patrick_steinhauer on 03.12.2014.

 */

public class LernkartenkomponenteDAO  {

   //TODO JDBC Operationen hier machen und an Lernkartenkomponente weiterreichen

   private Connection connection;

   private static LernkartenkomponenteDAO instance = null;

   private INutzerKomponentenDAO iNutzerKomponentenDAO =
NutzerKomponentenDAO.getInstance();

   public static LernkartenkomponenteDAO getInstance() {

      if(instance == null) {

         instance = new LernkartenkomponenteDAO();

      }

      return instance;

   }

   public boolean login() {

      try {

         DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
```

```
            connection = DriverManager
                    .getConnection(
                            "jdbc:oracle:thin:@oracle.informatik.haw-hamburg.de:1521/inf09",
                            "", "");
        } catch (SQLException ex) {
            System.err.println(ex.getMessage());
            return false;
        }
        return true;
    }
    public List<IModul> getModulListe() {
        List<IModul> listeMitModulen = new ArrayList<IModul>();

        try {
            login();
            String select = "SELECT * FROM MODUL";
            PreparedStatement pstmt = connection.prepareStatement(select);
            ResultSet resultset = pstmt.executeQuery();

            while(resultset.next()) {
                Modul modul = new Modul();
                modul.setModulID(resultset.getInt("MODULID"));
                modul.setModulName(resultset.getString("MODULNAME"));
                modul.setModulBeschreibung(resultset.getString("MODULBESCHREIBUNG"));
                listeMitModulen.add(modul);
            }
            resultset.close();
        } catch (SQLException e) {
```

```
            e.printStackTrace();
        }
        return listeMitModulen;
    }
    public List<ILernkarte> getListeDerLernkarten() {
        List<ILernkarte> lernkartenListe = new ArrayList<ILernkarte>();
        try {
            login();
            String select = "SELECT * FROM LERNKARTE";
            PreparedStatement pstmt = connection.prepareStatement(select);
            ResultSet resultset = pstmt.executeQuery();
            while(resultset.next()) {
                Lernkarte lernkarte = new Lernkarte();
                Modul modul = new Modul();
                lernkarte.setLernkartenID(resultset.getInt("LERNKARTENID"));
                lernkarte.setLernkartenName(resultset.getString("LERNKARTENNAME"));
                lernkarte.setUeberprueft(resultset.getString("LERNKARTEUEBERPRUEFT"));
                lernkartenListe.add(lernkarte);
            }
            resultset.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return lernkartenListe;
    }
    public Lernkarte getLernkarte(String lernkartenName) {
        Lernkarte lernkarte = new Lernkarte();
        try {
```

```
        login();
        String select = "SELECT * FROM LERNKARTE WHERE LERNKARTENNAME = '" + lernkartenName
+ "'";
        PreparedStatement pstmt = connection.prepareStatement(select);
        ResultSet resultset = pstmt.executeQuery();
        while(resultset.next()) {
           lernkarte.setLernkartenID(resultset.getInt("LERNKARTENID"));
           lernkarte.setLernkartenName(resultset.getString("LERNKARTENNAME"));
           lernkarte.setUeberprueft(resultset.getString("LERNKARTEUEBERPRUEFT"));
           lernkarte.setFrage(resultset.getString("LERNKARTENFRAGE"));
           lernkarte.setAntwort(resultset.getString("LERNKARTENANTWORT"));
           lernkarte.setUeberprueft(resultset.getString("LERNKARTEUEBERPRUEFT"));
//
lernkarte.setNutzer(iNutzerKomponentenDAO.getNutzerVonLernkarte(lernkarte.getErsteller().getNu
tzerID()));

lernkarte.setNutzer(iNutzerKomponentenDAO.getNutzerVonLernkarte(resultset.getInt("LERNKARTE
NERSTELLER")));
        }
        resultset.close();
      } catch (SQLException e) {
        e.printStackTrace();
      }
      return lernkarte;
   }
}
```

3.44 INutzerKomponentenServices

```
package nutzerKomponente;

/**
 * Created by patrick_steinhauer on 02.12.2014.
 */
public interface INutzerKomponenteServices {
    /**
     * Hier kommen noch Methoden für die Nutzerverwaltung herein.
     */
}
```

3.45 NutzerKomponente

```
package nutzerKomponente;

import antwortKomponente.IAntwortKomponenteServices;
import lernkartenKomponente.ILernkartenKomponenteServices;
import persistenz.IPersistenzServices;
/**
 * Created by patrick_steinhauer on 10.11.2014.
 */
public class Nutzerkomponente implements INutzerKomponenteServices {
    private ILernkartenKomponenteServices lernkartenKomponenteServices;
    private IAntwortKomponenteServices antwortKomponenteServices;
    private NutzerKomponentenDAO nutzerKomponentenDAO;
    public Nutzerkomponente(IPersistenzServices persistenzServices, ILernkartenKomponenteServices
lernkartenKomponenteServices, IAntwortKomponenteServices antwortKomponenteServices) {
        this.lernkartenKomponenteServices = lernkartenKomponenteServices;
        this.antwortKomponenteServices = antwortKomponenteServices;
    }
```

3.46 INutzerKomponenteDAO

```
package nutzerKomponente;

/**
 * Created by patrick_steinhauer on 10.12.2014.
 */
public interface INutzerKomponentenDAO {
    /**
     *
     * @param id
     * @return Es wird ein INutzer zurueckgegeben, der eine Lernkarte erstellt hat.
     */
    INutzer getNutzerVonLernkarte(int id);
    /**
     *
     * @return die verbindung für die Datenbank zurzeit noch doppelt und unschön weil der Login in jeder Komponente aufgerufen wird.
     */
    boolean login();
}
```

3.47 NutzerKomponenteDAO

```
package nutzerKomponente;

import lernkartenKomponente.ILernkarte;

import lernkartenKomponente.Lernkarte;

import lernkartenKomponente.Modul;

import java.sql.*;

import java.util.ArrayList;

import java.util.List;

/**

 * Created by Paddy-Gaming on 08.12.2014.

 */

public class NutzerKomponentenDAO implements INutzerKomponentenDAO{

   private Connection connection;

   private static NutzerKomponentenDAO instance = null;

   public static NutzerKomponentenDAO getInstance() {

      if(instance == null) {

         instance = new NutzerKomponentenDAO();

      }

      return instance;

   }

   @Override

   public boolean login() {

      try {

         DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());

         connection = DriverManager

               .getConnection(

                     "jdbc:oracle:thin:@oracle.informatik.haw-hamburg.de:1521/inf09",

                     "", "");
```

```
        } catch (SQLException ex) {
            System.err.println(ex.getMessage());
            return false;
        }
        return true;
    }
    @Override
    public INutzer getNutzerVonLernkarte(int id) {
        Nutzer nutzerEinerBestimmtenLernkarte = new Nutzer();
        try {
            login();
            String select = "SELECT * FROM INUTZER nutzer, LERNKARTE lk WHERE nutzer.NUTZERID = " +
id + " AND lk.LERNKARTENID = " + id + "";
            PreparedStatement pstmt = connection.prepareStatement(select);
            ResultSet resultset = pstmt.executeQuery();
            while(resultset.next()) {
                nutzerEinerBestimmtenLernkarte.setName(resultset.getString("NAME"));
//                nutzerEinerBestimmtenLernkarte.setEmail(new
EmailDatentyp(resultset.getString("EMAIL")));
                nutzerEinerBestimmtenLernkarte.setNutzerID(resultset.getInt("NUTZERID"));
            }
            resultset.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return nutzerEinerBestimmtenLernkarte;
    }}
```

3.48 LernkartenKomponentenTest

```
package tests;

import antwortKomponente.AntwortKomponente;

import antwortKomponente.IAntwortKomponenteServices;

import junit.framework.Assert;

import lernkartenKomponente.ILernkartenKomponenteServices;

import lernkartenKomponente.LernkartenKomponente;

import lernkartenKomponente.LernkartenkomponenteDAO;

import nutzerKomponente.INutzerKomponenteServices;

import nutzerKomponente.INutzerKomponentenDAO;

import nutzerKomponente.NutzerKomponentenDAO;

import nutzerKomponente.Nutzerkomponente;

import org.junit.Before;

import org.junit.Test;

import persistenz.IPersistenzServices;

/**
 * Created by patrick_steinhauer on 07.12.2014.
 */

public class LernkartenKomponentenTest {

    private IPersistenzServices iPersistenzServices = null;

    private IAntwortKomponenteServices iAntwortKomponenteServices;

    private ILernkartenKomponenteServices iLernkartenKomponenteServices;

//    private INutzerKomponenteServices iNutzerKomponenteServices;

    private INutzerKomponentenDAO iNutzerKomponentenDAO;

    private LernkartenkomponenteDAO lkdao;

    @Before

    public void setUp() {

        lkdao = new LernkartenkomponenteDAO(); //LernkartenkomponenteDAO.getInstance();
```

```
        iNutzerKomponentenDAO = NutzerKomponentenDAO.getInstance();

            iAntwortKomponenteServices = new AntwortKomponente(iPersistenzServices);

        iLernkartenKomponenteServices = new LernkartenKomponente(iPersistenzServices,
iAntwortKomponenteServices);

//      iNutzerKomponenteServices = new Nutzerkomponente(iPersistenzServices,
iLernkartenKomponenteServices, iAntwortKomponenteServices);

        iNutzerKomponentenDAO.login();

        lkdao.login();

    }

    @Test

    public void testGetLernkartenFromDatabase() {

        Assert.assertEquals(3, iLernkartenKomponenteServices.getListeDerLernkarten().size());

        Assert.assertTrue(!(iLernkartenKomponenteServices.getListeDerLernkarten().size() > 3));

    }

    @Test

    public void testGetLernkarteFromDatabase() {

        Assert.assertEquals(1, iLernkartenKomponenteServices.getLernkarte("SE-
Lernkarte01").getLernkartenNummer());

        Assert.assertEquals("SE-Lernkarte01", iLernkartenKomponenteServices.getLernkarte("SE-
Lernkarte01").getLernkartenName());

        Assert.assertEquals("WELCHE FARBE HAT MEINE UNTERHOSE?",
iLernkartenKomponenteServices.getLernkarte("SE-Lernkarte01").getLernkartenFrage());

        Assert.assertNotSame("WELCHE FARBE HAT MEINE HOSE?",
iLernkartenKomponenteServices.getLernkarte("SE-Lernkarte01").getLernkartenFrage());

    }


    /**

     * Dieser Test funktioniert leider noch nicht ganz weil mein Datenbankschema zu kompliziert war,
weswegen ich den Fehler nicht mehr gefunden habe.

     * einen User heraus holen anhand seiner lernkarten id ist jedoch möglich test dafür im
nutzerkomponentenTest
```

```
    */
    @Test
    public void testGetUserVonlernkarte() {
        Assert.assertEquals("Son-Goku", iLernkartenKomponenteServices.getLernkarte("SE-
Lernkarte02").getErsteller().getName());
    }
}
```

3.49 NutzerKomponentenTest

```
package tests;

import antwortKomponente.AntwortKomponente;

import antwortKomponente.IAntwortKomponenteServices;

import junit.framework.Assert;

import lernkartenKomponente.ILernkartenKomponenteServices;

import lernkartenKomponente.LernkartenKomponente;

import lernkartenKomponente.LernkartenkomponenteDAO;

import nutzerKomponente.INutzerKomponenteServices;

import nutzerKomponente.INutzerKomponentenDAO;

import nutzerKomponente.NutzerKomponentenDAO;

import nutzerKomponente.Nutzerkomponente;

import org.junit.Before;

import org.junit.Test;

import persistenz.IPersistenzServices;

/**

 * Created by patrick_steinhauer on 07.12.2014.

 */

public class NutzerKomponentenTest {

    private IPersistenzServices iPersistenzServices = null;

    private IAntwortKomponenteServices iAntwortKomponenteServices;

    private ILernkartenKomponenteServices iLernkartenKomponenteServices;

    private INutzerKomponenteServices iNutzerKomponenteServices;

    private INutzerKomponentenDAO iNutzerKomponentenDAO;

    private LernkartenkomponenteDAO lkdao;



    @Before
```

```
    public void setUp() {
        lkdao = new LernkartenkomponenteDAO(); //LernkartenkomponenteDAO.getInstance();
        iNutzerKomponentenDAO = NutzerKomponentenDAO.getInstance();
        iAntwortKomponenteServices = new AntwortKomponente(iPersistenzServices);
        iLernkartenKomponenteServices = new LernkartenKomponente(iPersistenzServices,
iAntwortKomponenteServices);
        iNutzerKomponenteServices = new Nutzerkomponente(iPersistenzServices,
iLernkartenKomponenteServices, iAntwortKomponenteServices);
        iNutzerKomponentenDAO.login();
        lkdao.login();
    }
    @Test
    public void testGetUserVonLernkarte() {
        org.junit.Assert.assertEquals("Son-Goku",
iNutzerKomponentenDAO.getNutzerVonLernkarte(2).getName());
    }
}
```

4.0 SQL Schema

4.1 Erstellung der DB

```
DROP TABLE LERNKARTE;

DROP TABLE INUTZER;

DROP TABLE MODUL;

-- Tabelle der Nutzer -> INUTZER weil in meiner Datenbank schon ein NUTZER war den ich noch brauchte

CREATE TABLE INUTZER

(

      NUTZERNAME          VARCHAR2(30)

,     NAME                VARCHAR2(200)

,     LERNKARTENID        NUMBER(11)

,     EMAIL               VARCHAR2(100)

,     NUTZERKENNUNG       VARCHAR2(6)

,     NUTZERID            NUMBER(11)

,     PRIMARY KEY(NUTZERID)

--,    FOREIGN KEY(LERNKARTENID) REFERENCES LERNKARTE(LERNKARTENID)

);

-- Tabelle der Lernkarte

CREATE TABLE LERNKARTE

(

      LERNKARTENID                NUMBER(11)

,     LERNKARTENNAME              VARCHAR2(100)    NOT NULL

,     LERNKARTENERSTELLER         NUMBER(11)       NOT NULL

,     LERNKARTENFRAGE             VARCHAR2(1000)

,     LERNKARTENANTWORT           VARCHAR2(1000)

,     LERNKARTENMODUL             VARCHAR2(200)    NOT NULL

,     LERNKARTEUEBERPRUEFT        VARCHAR2(10)     NOT NULL
```

```
,     PRIMARY KEY(LERNKARTENID)

,     FOREIGN KEY(LERNKARTENERSTELLER) REFERENCES INUTZER(NUTZERID)

);

-- Eine Tabelle für Module, wird jedoch nicht genutzt zur Zeit, da erst einmal nur Modul in der
Lernkarte steht

CREATE TABLE MODUL

(

     MODULID              NUMBER(11)

,    MODULNAME            VARCHAR2(100)

,    MODULBESCHREIBUNG    VARCHAR2(2000)

);
```

4.2 Inserts in die DB

```
INSERT INTO INUTZER (NUTZERNAME, NAME, LERNKARTENID, EMAIL, NUTZERKENNUNG, NUTZERID)

VALUES ('Vegeta', 'Vegeta', 1, 'saiyajinvegeta@sole.de', 'veg111', 1);


INSERT INTO INUTZER (NUTZERNAME, NAME, LERNKARTENID, EMAIL, NUTZERKENNUNG, NUTZERID)

VALUES ('Son-Goku', 'Son-Goku', 2, 'saiyajinsongoku@sole.de', 'sgk445', 2);


INSERT INTO LERNKARTE (LERNKARTENID, LERNKARTENNAME, LERNKARTENERSTELLER,
LERNKARTENFRAGE, LERNKARTENANTWORT, LERNKARTENMODUL, LERNKARTEUEBERPRUEFT)

VALUES (1, 'SE-Lernkarte01', 1, 'WELCHE FARBE HAT MEINE UNTERHOSE?', 'Weiß', 'SE', 'false');


INSERT INTO LERNKARTE (LERNKARTENID, LERNKARTENNAME, LERNKARTENERSTELLER,
LERNKARTENFRAGE, LERNKARTENANTWORT, LERNKARTENMODUL, LERNKARTEUEBERPRUEFT)

VALUES (2, 'SE-Lernkarte02', 2, 'WELCHE FARBE hat mein tshirt?', 'Rot', 'SE3', 'false');


INSERT INTO LERNKARTE (LERNKARTENID, LERNKARTENNAME, LERNKARTENERSTELLER,
LERNKARTENFRAGE, LERNKARTENANTWORT, LERNKARTENMODUL, LERNKARTEUEBERPRUEFT)

VALUES (3, 'BS-Lernkarte01', 1, 'Was ist ein MUTEX?', 'Wissen wir nicht', 'BS', 'false');
```