

# Schulverwaltungsprogramm

## (SYSTEMTEST-DOKUMENT)

Istanbul.Montag: 02.06.2023  
Projektbetreuer: **Dr. Ömer Karacan**  
Projektkoordinator: **Arş. Gör. Fulya Yenilmez**

Gruppenmitglieder:

- Muhammed Alobayd
- Mohamad Eyad Abras



Muhammed Alobayd  
TMS 4  
e170501109@stud.tau.edu.tr



Mohamad Eyad Abras  
TMS 4  
e170501104@stud.tau.edu.tr

1. Systemüberblick
2. Systemtestfälle
3. Rückverfolgbarkeit der Anforderungen



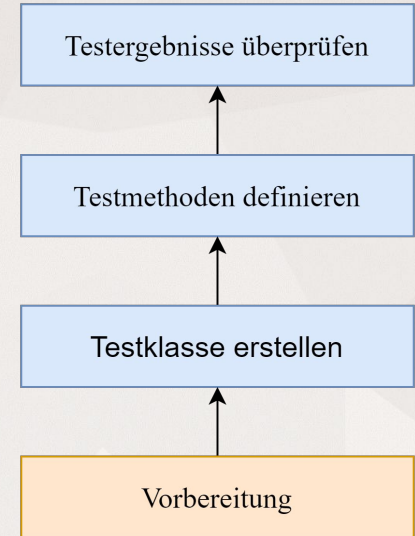
- ❖ Unsere Software ist eine Schulverwaltungssoftware, die in Java entwickelt wurde.
- ❖ Das Hauptprogramm besteht aus mehreren Klassen und Dateien. Die zentrale Klasse des Projekts ist "Hauptseite", eine Java-Klasse, die die Benutzeroberfläche bereitstellt. Diese Klasse erbt von der Klasse "javax.swing.JFrame" und enthält verschiedene Komponenten wie Labels, Textfelder und Schaltflächen.
- ❖ Um eine Verbindung mit der MySQL-Datenbank herzustellen, wird die JDBC (Java Database Connectivity)-API verwendet. Dafür werden die entsprechenden Klassen importiert und der JDBC-Treiber geladen. Anschließend wird eine Verbindung zur Datenbank hergestellt, indem die erforderlichen Verbindungsdaten angegeben werden. Also das Laden der Klasse registriert auch den Treiber im DriverManager, der in der Lage ist, auf die Datenbank mit folgenden Code zuzugreifen:

```
Class.forName("com.mysql.jdbc.Driver");  
Connection conn=  
DriverManager.getConnection("jdbc:mysql://localhost:3306/sms",  
"root", "");
```



- ❖ Die Testumgebung umfasst ein System unter Test (SUT), das als "Verarbeiten" bezeichnet wird, und ein Testtool, das für die Durchführung der Tests verwendet wird. Das SUT, also das System unter Test, ist die Anwendung "editteacher -editstudent ". Es handelt sich um eine Java Swing-Anwendung für die Bearbeitung von Lehrer-studentsdaten. Die Anwendung bietet eine grafische Benutzeroberfläche (GUI) mit verschiedenen Textfeldern und Schaltflächen, um Informationen wie ID, Name, Spezialisierung und zugeordnetes Fach eines Lehrers zu bearbeiten. Die Anwendung ist in der Lage auf eine MySQL-Datenbank zuzugreifen, um die Änderungen zu speichern.
- ❖ Das verwendete Testtool ist nicht explizit angegeben, aber basierend auf dem Codeausschnitt könnte es sich um eine Java-Testbibliothek oder ein Framework handeln, das für die Testautomatisierung verwendet wird. Natürlich ist es möglich verschiedene Java-Test Frameworks wie JUnit oder TestNG verwendet werden, aber wir haben für JUnit Test eine Unterscheidung getroffen. Der Grund dafür liegt daran, dass es einfacher als anderen die Tests ist, zu verwenden. Dieses Test Tool ermöglicht das Erstellen und Ausführen von Tests sowie die Überprüfung der erwarteten Ergebnisse. Die Funktionalität automatisiert zu testen.

1. Es wird sichergestellt, dass erforderliche Test-Framework in deinem Projekt eingerichtet ist. also In Java ist JUnit das gängigste Test-Framework und Entwicklungsumgebung ist konfiguriert, um JUnit-Tests ausführen zu können
2. Es wird eine separate Testklasse für die zu testende Klasse oder Komponente dargestellt.
3. Es werden Testmethoden innerhalb der Testklasse definiert und jede Testmethode sollte eine spezifische Funktion oder ein spezifisches Verhalten der zu testenden Klasse überprüfen. Darüber hinaus wird die Annotation '@Test' über die Testmethode verwendet, um JUnit als Testfall zu erkennen.
4. Es werden die Testergebnisse für alle Methoden überprüft, um sicherzustellen, dass alle Tests erfolgreich waren. Wenn ein Test fehlschlägt, überprüfe die Fehlermeldung, um den Grund des Fehlers zu identifizieren



## ❖ Der Code #1

```
@Test
public void testInsertDataIntoDatabase() throws ClassNotFoundException {
    String hh="John" ;
    System.out.println("hh:");
    // Mocking the input values for the test
    String t1Value = "1";
    String t2Value = "Muhammed";
    String t3Value = "alobayd";
    String t4Value = "12";
    String t5Value = "09";
    String t6Value = "a";
    String t7Value = "Byzkos";

    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/sms", user: "root", password: "");
        String sql= "insert into muhm values(?,?, ?,?, ?,?, ?,?)";

        PreparedStatement pstmt= conn.prepareStatement(string:sql);
        pstmt.setString(1, string:t1Value);
        pstmt.setString(2, string:t2Value);
        pstmt.setString(3, string:t3Value);
        pstmt.setString(4, string:t4Value);
        pstmt.setString(5, string:t5Value);
        pstmt.setString(6, string:t6Value);
        pstmt.setString(7, string:t7Value);

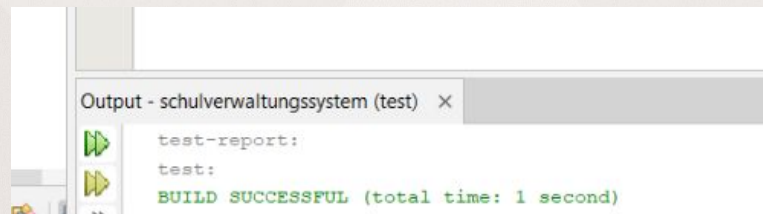
        pstmt.executeUpdate();
    } catch (SQLException e) {
        Assertions.fail("Exception occurred: " + e.getMessage());
    }
    try {
        // Connect to the database
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/sms", user: "root", password: "");
        String selectSql = "SELECT * FROM muhm WHERE id = " + t1Value + "";
        PreparedStatement stmt= conn.prepareStatement(string:selectSql);

        // Execute the insertion query

        // Verify that the data was inserted correctly
        ResultSet rs = stmt.executeQuery(string:selectSql);
        if (rs.next()) {
            String retrievedT2Value = rs.getString(string:"t2");
            String retrievedT3Value = rs.getString(string:"t3");
            String retrievedT4Value = rs.getString(string:"t4");
            String retrievedT5Value = rs.getString(string:"t5");
            String retrievedT6Value = rs.getString(string:"t6");
            String retrievedT7Value = rs.getString(string:"t7");

            Assertions.assertEquals(expected: t2Value, actual:retrievedT2Value);
            Assertions.assertEquals(expected: t3Value, actual:retrievedT3Value);
            Assertions.assertEquals(expected: t4Value, actual:retrievedT4Value);
            Assertions.assertEquals(expected: t5Value, actual:retrievedT5Value);
            Assertions.assertEquals(expected: t6Value, actual:retrievedT6Value);
            Assertions.assertEquals(expected: t7Value, actual:retrievedT7Value);
        }
    }
}
```

## ❖ Die Ausgabe



```
// Fail the test if no data was found
Assertions.fail(message:"No data found in the database");
}

// Close the database connection
conn.close();
} catch (ClassNotFoundException | SQLException e) {
    Assertions.fail("Exception occurred: " + e.getMessage());
}
}
```

<i>Test Schritt</i>	
<b>Name der Systemtestfall-ID</b>	einen neuen Student einfügen
<b>Post-Bedingung</b>	
<b>Testdaten</b>	<pre>String t1Value = "1"; String t2Value = "Muhammed"; String t3Value = "alobayd"; String t4Value = "12"; String t5Value = "09"; String t6Value = "a"; String t7Value = "Baykoz";</pre>
<b>erwartetes Ergebnis</b>	1, Muhammed, Alobayd, 12, 09, a, Beyloz
<b>tatsächliche Ergebnis</b>	1, Muhammed, Alobayd, 12, 09, a, Beyloz
<b>Urteil (Bestanden/Fall)</b>	Pass



## ❖ Der Code #2

```
@Test
public void testJButtonActionPerformed() {
    // Prepare test data
    String id = "15667";
    String name = "Ali";
    String spezialisierung = "Algebra";
    String fach = "Mathematik";

    // Call the method to be tested
    jButtonActionPerformed(id, name, spezialisierung, fach);

    // Check if the data was stored correctly in the database
    assertTrue(condition:checkIfDataExists(id, name, spezialisierung, fach));
}

private void jButtonActionPerformed(String id, String name, String spezialisierung, String fach) {
    try {
        Class.forName(classname:"com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection(url:"jdbc:mysql://localhost:3306/sms", user:"root", password:"");
        String sql = "insert into lehrer values (?, ?, ?, ?)";
        PreparedStatement ptst = conn.prepareStatement(string:sql);
        ptst.setString(1, string:id);
        ptst.setString(2, string:name);
        ptst.setString(3, string:spezialisierung);
        ptst.setString(4, string:fach);
        ptst.executeUpdate();
        conn.close();
    } catch (Exception e) {
        fail("Exception occurred: " + e.getMessage());
    }
}

private boolean checkIfDataExists(String id, String name, String spezialisierung, String fach) {
    try {
        Class.forName(classname:"com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection(url:"jdbc:mysql://localhost:3306/sms", user:"root", password:"");
        String sql = "select * from lehrer where id = ?";
        PreparedStatement ptst = conn.prepareStatement(string:sql);
        ptst.setString(1, string:id);
        ResultSet rs = ptst.executeQuery();
        if (rs.next()) {
            String storedId = rs.getString(string:"id");
            String storedName = rs.getString(string:"name");
            String storedSpezialisierung = rs.getString(string:"spezialisierung");
            String storedFach = rs.getString(string:"fach");
            conn.close();
            return id.equals(anoObject: storedId) && name.equals(anoObject: storedName) &&
                spezialisierung.equals(anoObject: storedSpezialisierung) && fach.equals(anoObject: storedFach);
        }
        conn.close();
    } catch (ClassNotFoundException | SQLException e) {
        fail("Exception occurred: " + e.getMessage());
    }
    return false;
}
```

## ❖ Die Ausgabe

Output - schulverwaltungssystem (test-single) X



test-single:

BUILD SUCCESSFUL (total time: 1 second)

<i>Test Schritt</i>	
<b>Name der Systemtestfall-ID</b>	einen neuen Lehren einfügen.
<b>Post-Bedingung</b>	
<b>Testdaten</b>	String id = "15667"; String name = "Ali"; String spezialisierung = "Algebra"; String fach = "Mathematik";
<b>erwartetes Ergebnis</b>	15667,Ali, Algebra, Mathematik
<b>tatsächliche Ergebnis</b>	15667,Ali, Algebra, Mathematik
<b>Urteil (Bestanden/Fall)</b>	Pass

## ❖ Der Code #3

```
@Test
public void testNotenberechnung() {
    // Testfall 1: Gesamtnote über 80
    double ph = 90.0;
    double mt = 85.0;
    double ch = 88.0;
    double expected1 = 87.67; // Erwartete Gesamtnote
    String expectedGrade1 = "A+"; // Erwartete Note

    double result1 = berechneGesamtnote(ph, mt, ch);
    String resultGrade1 = ermittelNote(gesamtnote: result1);

    Assertions.assertEquals(expected: expected1, actual: result1, delta: 0.01);
    Assertions.assertEquals(expected: expectedGrade1, actual: resultGrade1);

}

// Hilfsmethode zur Berechnung der Gesamtnote
private double berechneGesamtnote(double ph, double mt, double ch) {
    return ((ph + mt + ch) * 100) / 300;
}

// Hilfsmethode zur Ermittlung der Note basierend auf der Gesamtnote
private String ermittelNote(double gesamtnote) {
    if (gesamtnote >= 80) {
        return "A+";
    } else if (gesamtnote >= 70 && gesamtnote < 80) {
        return "A";
    } else if (gesamtnote >= 60 && gesamtnote < 70) {
        return "B";
    } else if (gesamtnote >= 50 && gesamtnote < 60) {
        return "C";
    } else if (gesamtnote >= 40 && gesamtnote < 50) {
        return "D";
    } else {
        return "Fail";
    }
}
```

## ❖ Die Ausgabe

Output - schulverwaltungssystem (test-single) X



test-single:

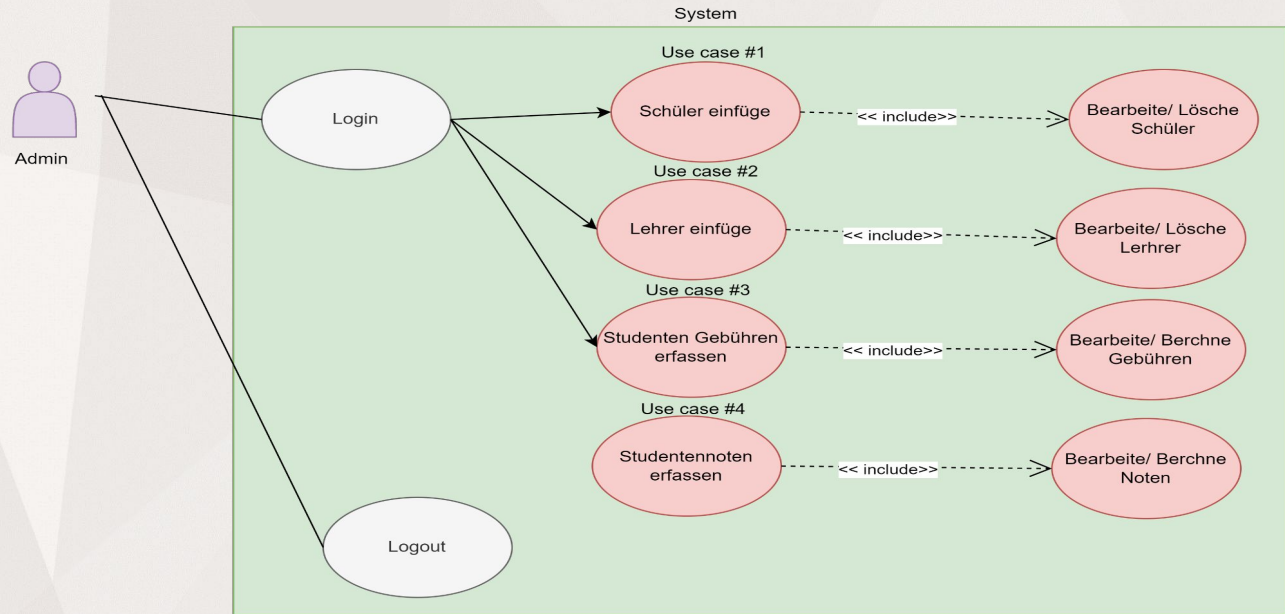


BUILD SUCCESSFUL (total time: 0 seconds)

<i>Test Schritt</i>	
<b>Name der Systemtestfall-ID</b>	Die gesamten Noten für den Studenten berechnen, ob er bestanden oder nicht ist.
<b>Post-Bedingung</b>	
<b>Testdaten</b>	Physik = 90.0; Mathematik = 85.0; Programmierung = 88.0;
<b>erwartetes Ergebnis</b>	90, 85, 88, A
<b>tatsächliche Ergebnis</b>	90, 85, 88, A
<b>Urteil (Bestanden/Fall)</b>	Pass



- ❖ Unsere Software bietet dem Administrator eine Vielzahl von Szenarien und Funktionen, um die Informationen zu verwalten. Der Admin hat die Möglichkeit, verschiedene Aktionen im Zusammenhang durchzuführen. Hier sind einige Beispiele dafür, was der Admin tun kann:



# Rückverfolgbarkeit der Anforderungen

## ❖ USE CASE #Login:

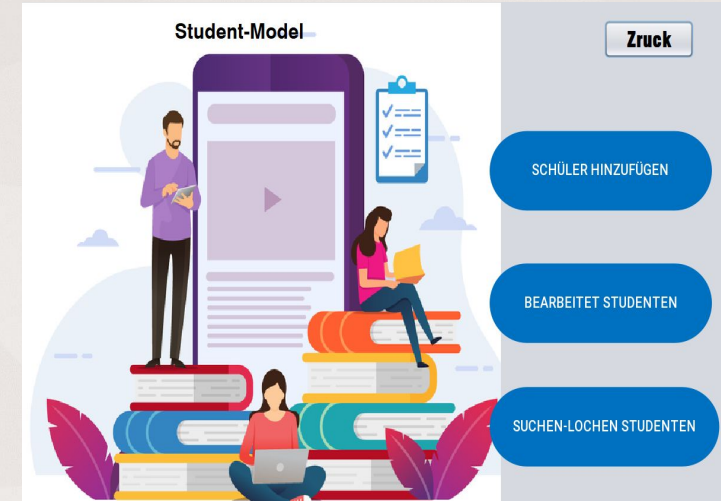
- Nachdem das Programm geöffnet ist, werden die einzelnen Datensätze im ResultSet überprüft. Es wird alle Benutzerdaten aus der Tabelle "userlogin" abgerufen. Der Benutzername und das Passwort werden mit den eingegebenen Werten verglichen. Wenn sie übereinstimmen, wird ein neues Fenster (Klasse "Hallo") angezeigt. Andernfalls wird eine Fehlermeldung angezeigt, dass der Benutzername oder das Passwort falsch ist.
- Die Methode "jButton2ActionPerformed" wird aufgerufen, wenn die zweite Schaltfläche (jButton2) geklickt wird. Jetzt wird überprüft, ob der eingegebene Geheimcode dem Wert im Textfeld "admin@123" entspricht. Wenn ja, wird eine entsprechende Meldung angezeigt.



# Rückverfolgbarkeit der Anforderungen

## ❖ USE CASE #Studentennoten -Module:

- Der Administrator kann neue Studenten dem System hinzufügen. Dabei werden Informationen wie Name, Kontaktdaten und Zugangsdaten für den Studenten festgelegt. Außerdem ist es möglich, die Details zu bearbeiten, z.B. um geänderte Kontaktdaten oder Fachgebiete zu aktualisieren.



**Zurück**

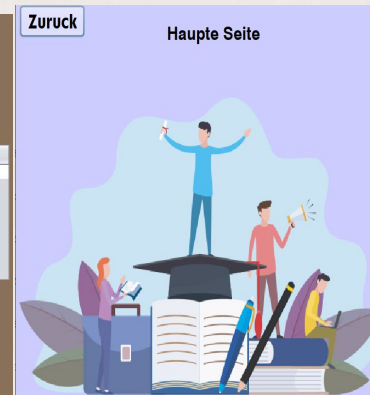
Suchen und Lochen von den Studierenden

SUCHEN

ID	Vorname	Nachname	Telefonnummer	Telefonnummer fuer Elt.	Klasse	Adresse
7050145	muhammed	alabayd	564380	42141	143431	233

Geben Sie bitte die ID von den Studenten an

Lochen



**Zurück**

STUDENTM-MODEL


LEHRER-MODEL

GEBÜHREN\_EINREICHUNGSMODEL

NOTEN

**Zurück**

Studentenanmeldung



ID

Vorname

Nachname

Telefonnummer

F-Telefonnummer

Klasse

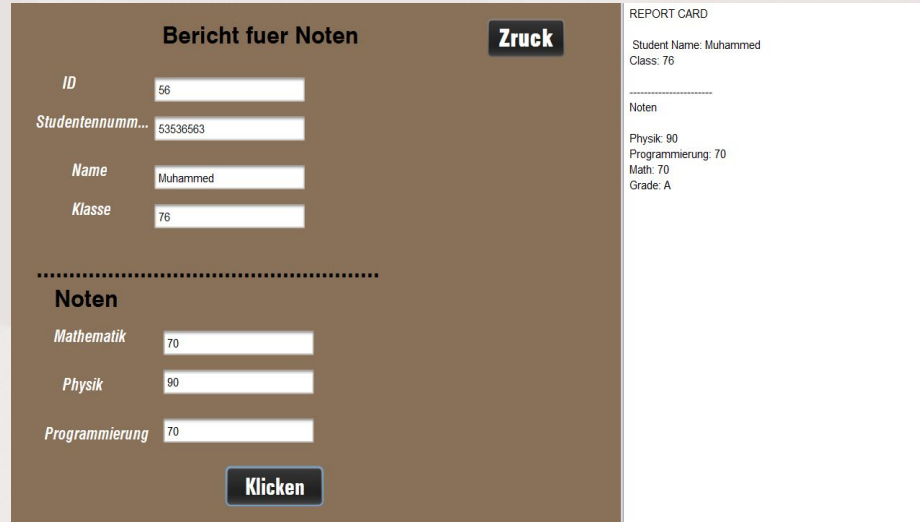
Adresse

Klicken



## ❖ USE CASE #Studentennoten -Module:

- Zusätzlich wird in diesem User Case berechnet, ob der Schüler bestanden hat oder nicht, basierend auf den eingegebenen Noten. Die berechnete Note wird in der Variable "Noten" gespeichert. Der Bericht mit den eingegebenen Daten und der berechneten Note wird im Textfeld "schreiben" angezeigt.
- In diesem Beispiel hat der Student (Muhammed ) die Noten in Math 70 und Physik 90 und Programmierung 90 , deshalb hat er mit Buchstabe A bestanden, was in dieser Abbildung deutlich gezeigt wurde.



**Bericht fuer Noten** **Zruck**

ID: 56

Studentennumm...: 63536563

Name: Muhammed

Klasse: 76

**Noten**

Mathematik: 70

Physik: 90

Programmierung: 70

**Klicken**

**REPORT CARD**

Student Name: Muhammed  
Class: 76

Noten

Physik: 90  
Programmierung: 70  
Math: 70  
Grade: A



# Rückverfolgbarkeit der Anforderungen

- Der Zusammenhang zwischen Anforderung/Anwendungsfällen und Controller-Klasse, Webdiensten zur Datenbank wird ausführlich demonstriert, wenn der Benutzer das Programm startet und die Eingabe eines Passworts erwartet, gefolgt von der Auswahl des Modells. Daher bestimmt der Benutzer seine beabsichtigte Aktion, beispielsweise das Einfügen eines neuen Schülers.
- Die grafische Benutzeroberfläche (GUI) ist mit Kontrollklassen verknüpft, die, wie aus dem Diagramm unten hervorgeht, Verbindungen mit der Datenbank und den Webdiensten herstellen.

