MODUL 326 PROJEKT A

BestellStatus@CoolShoes

1 AUSGANGSLAGE

Die Firma `CoolShoes` bietet in einem Internet-Shop Schuhe an.

Der WEB-Shop ist gut aufgebaut und wird rege benutzt. Die Kunden bemängeln aber, dass sie keinen Bestellstatus haben.

Im Projekt **BestellStatus@CoolShoes** soll ein solches Zusatzsystem erstellt werden.

Hinweis: Die Shop-Funktionalität muss nicht programmiert werden. Diese wird nur zum besseren Verständnis jeweils erwähnt.

2 ANFORDERUNGSSPEZIFIKATION

2.1 Systemübersicht

Der WEB-Shop von `CoolShoes` bietet die übliche Funktionalität an. Ein Benutzer muss sich vor dem Kauf registrieren und seine Kontaktdaten angeben. Danach kann er einkaufen. Bei einem Einkauf kann er mehrere Produkte im Warenkorb ablegen und am Ende den Kauf bestätigen. Die Zahlung erfolgt mittels Kreditkarte oder über Paypal. Die Lieferung durch `CoolShoes` erfolgt mit unterschiedlichen Partner wie z.B. DHL, Post usw.

Jeder Auftrag erhält eine eindeutige Transaktionsnummer / Bestellnummer. Über diese Nummer kann die Bestellung bis hin zur Bezahlung der Rechnung und darüber hinaus auch zu allfälligen Reklamationen verfolgt werden. Diese Nummer ist auch Teil des neuen Teilsystem **BestellStatus@CoolShoes**

2.2 Abläufe und Funktionen

Jede Bestellung wird durch eine Transaktionsnummer / Bestellnummer eindeutig identifiziert. Diese Nummer wird durch das Bestellsystem generiert und dem Auftrag zugewiesen.

2.2.1 Auftrag aufbereiten

Sobald im Vertrieb die Bestellung geprüft wird, muss der Mitarbeiter im System den Bestellstatus auf ‘Auftrag aufbereiten‘ stellen.

Ist erkennbar, dass der Auftrag nicht als Ganzes bearbeitbar ist (z.B. fehlt ein Artikel im Lager), muss der Mitarbeiter einen Vermerk setzen. Somit kann das System der Transaktionsnummer einen Zusatz A..Z beifügen (wir gehen davon aus, dass ein einzelner Auftrag nicht in 24 Teilaufträgen erledigt wird)

Kann ein Auftrag nicht als Ganzes ausgelöst werden, muss der Mitarbeiter für die verspäteten Teilaufträge den Status auf ‘Teilauftrag verspätet‘ setzen und das mutmassliche Datum der Lieferung festhalten.

2.2.2 Auftrag rüsten

Ist die Bestellung aufbereitet, wird der Auftrag gerüstet. Im Lager werden die bestellten Produkte zusammengestellt. Der Lagermitarbeiter erhält immer einen vollständig

rüstbaren Auftrag vom Vertrieb, d.h. dass Teilaufträge in mehreren Sequenzen abgearbeitet werden.

Der Lagermitarbeiter stellt nach dem Rüsten den Status der Bestellung auf ‘Auftrag versandbereit‘.

2.2.3 Auftrag ausliefern

Nach dem Rüsten ist der Auftrag versandbereit und wird einem Versandpartner zur Lieferung übergeben.

Der Versandpartner ist dafür verantwortlich, dass der Status korrekt nachgeführt wird. Konkret muss er zwei Stati eintragen, nämlich ‘Auftrag abgeholt‘ und ‘Auftrag geliefert‘.

Der erste Status meint, dass der Versandpartner die Lieferung bei `CoolShoes` abholt und in sein eigenes Verteilzentrum bringt.

Der zweite Status wird gesetzt, sobald die Lieferung das Verteilzentrum verlässt und sich somit auf dem Weg zum Kunden befindet.

2.2.4 Satus anschauen

Der Kunde kann im WEB-Shop jederzeit den aktuellen Status seines Auftrags bzw. der Teilaufträge nachsehen.

2.2.5 History

Für `CoolShoes` muss es jederzeit möglich sein, die Stati eines Auftrags als History zu betrachten.

Wird ein Auftrag in Teilaufträge unterteilt, müssen alle Teilaufträge mit deren Stati aufgelistet werden.

2.3 Daten

Für jeden Auftrag muss zu jeder Zeit ein Status gesetzt sein; mit Datum der Bearbeitung und Namen des Bearbeiters.

Wird ein Auftrag in mehrere Teilaufträge unterteilt, sind die Daten pro Teilauftrag zu setzen. Zudem muss bei Aufträgen, die sich verspäten, der mutmassliche Liefertermin angegeben werden.

2.4 Benutzerschnittstellen

2.4.1 Bestellung

Bei der Bestellung muss der Kunde keine besonderen Vorkehrungen treffen, um den Status der Bestellung auf ‚‘Auftrag bestellt‘ zu setzen. Dies erfolgt aus der WEB-Anwendung heraus.

2.4.2 Vertrieb

Im Vertrieb wird der Status entsprechend der Beschreibung festgelegt. Der Auftrag wird dabei über die Transaktionsnummer / Bestellnummer identifiziert.

2.4.3 Lager

Im Lager wird der Status entsprechend der Beschreibung festgelegt. Der Auftrag wird dabei über die Transaktionsnummer / Bestellnummer identifiziert.

2.4.4 Lieferant

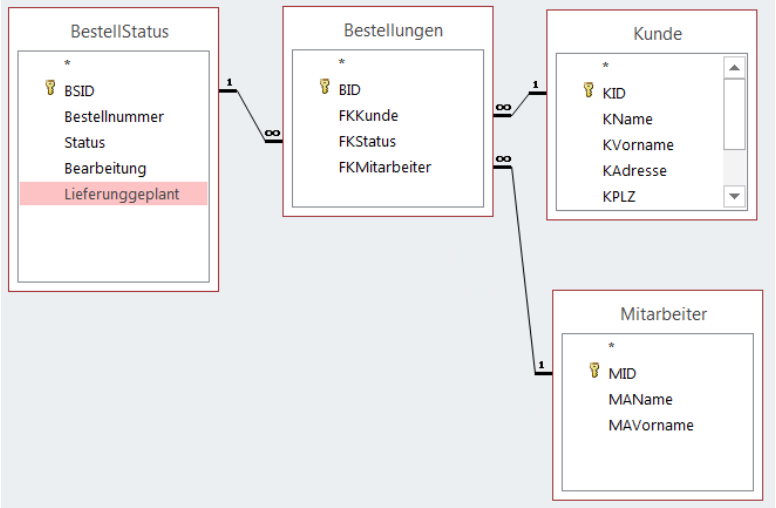
Der Lieferant passt den Status entsprechend der Beschreibung an. Der Auftrag wird dabei über die Transaktionsnummer / Bestellnummer identifiziert.

2.5 Externe Schnittstellen

2.5.1 Datenbank

2.5.1.1 ERD

Die bestehende Datenbank wird um die Tabelle BestellStatus ergänzt. Das ERD zeigt den Teil, der für die Verwaltung des Status relevant ist.



2.5.1.2 View der Teil-DB

KName	KVorname	Bestellnummer	Status	Bearbeitung	Lieferunggeplant	MAName	MAVorname
Trump	Donald	1558972	Auftrag bestellt	03.03.2015		Ringo	Star
Trump	Donald	1558972	Auftrag aufbereiten	04.03.2015		Clinton	Hillary
Trump	Donald	1558972	Auftrag versandbereit	05.03.2015		Max	Muster
Max	Moritz	5268988	Auftrag bestellt	03.03.2015		Ringo	Star
Max	Moritz	5268988A	Auftrag aufbereiten	04.03.2015		Blanco	Robert
Max	Moritz	5268988B	Teilauftrag verspätet	04.03.2015	20.03.2015	Blanco	Robert
Castro	Fidel	4423568	Auftrag bestellt	04.03.2015		Pedro	Enrique
Castro	Fidel	4423568	Auftrag aufbereiten	05.03.2015		Clinton	Hillary
Varoufakis	Yanis	8526322	Auftrag bestellt	05.03.2015		Ringo	Star

3 DB-ANBINDUNG MIT JAVA

Java greift über die sog. JDBC[1] auf externe Datenquellen zu. Je nach Datenbank (Access, mySQL, Oracle ….) müssen entsprechende Treiber verfügbar sein.

Für gewisse Produkte sind die Treiber im Umfang der Java SE enthalten, nicht aber für das Produkt Access. Hier müssen Libraries dem Projekt zugefügt werden.

3.1 JDBC für Access

3.1.1 Treiber

Microsoft stellt für Access keine Java-Unterstützung bereit. Somit ist man auf einen Drittanbieter angewiesen. Unter http://ucanaccess.sourceforge.net/site.html findet sich eine Implementation unter Opensource-Lizenz.

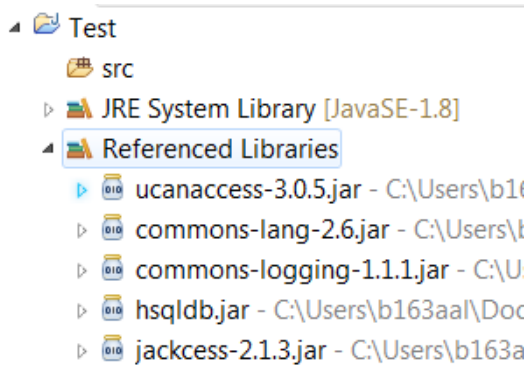
1. Laden Sie die Dateien von UCanAccess auf Ihren Rechner.

   Download UCanAccess 3.0.5

3.1.2 Einbinden in Projekt

2. Entpacken Sie die ZIP-Datei auf Ihrem Rechner. Das Zielverzeichnis kann an beliebiger Stelle liegen.
3. Erstellen Sie in Eclipse ein Java-Projekt.
4. Binden Sie die die folgenden Libraries ins Projekt ein:
   - ucanaccess-3.0.5.jar
   - lib\commons-lang-2.6.jar
   - lib\commons-logging-1.1.1.jar
   - lib\hsqldb.jar
   - lib\jackcess-2.1.3.jar

   Klicken Sie dazu mit der rechten Maustaste auf das erstellte Projekt und wählen Sie das Menü *BuildPath ▸ Add External Archives…*
5. Wählen Sie im Dialog das Verzeichnis aus, in das Sie die Treiber entzippt haben. Fügen Sie die 5 genannten Datei zu.



3.1.3 JDBC-Verbindung codieren

Die JavaSE liefert alle benötigten Funktionalitäten, um Datenbanken mittels SQL-Befehlen zu bearbeiten. Genau Details finden Sie in der API des Pakets java.sql.

3.1.3.1 Benötigte Pakete

```
import java.sql.*;
```

[1] Mehr zu JDBC findet sich unter https://de.wikipedia.org/wiki/Java_Database_Connectivity

3.1.3.2 Benötigte Objekte

Die Verbindung zur Datenbank wird über ein `Connection`-Objekt realisiert

```
private Connection conn;
```

3.1.3.3 Datenbank verbinden

Folgender Beispielcode zeigt, wie eine Datenbank über JDBC angebunden wird.

```
try{
  // Datenbank über Connector verbinden
  Class.forName("net.ucanaccess.jdbc.UcanaccessDriver");
  conn=DriverManager.getConnection("jdbc:ucanaccess://" + path);
}
catch (ClassNotFoundException err){
  // Die Teiber-Kalsse (hier UcanaccessDriver) ist nicht verfügbar
  System.out.println("Treiber kann nicht geladen werden");
}
catch(SQLException err){
  // Die Datenbank kann nicht kontaktiert werden.
  // (falscherName, falscher Pfad, falscher Treiber...)
  System.out.println("Verbindung kann nicht aufgebaut werden");
}
```

Wobei `path` für den (relativen) Verzeichnispfad zur Access-Datenbank des Projekts (CollShoes.accdb) steht.

3.1.3.4 Daten abfragen

Um Daten auszulesen, muss zuerst ein `Statement`-Objekt erstellt und mit dem entsprechenden SQL-Befehl versehen werden. Das Ergebnis erscheint in einem `ResultSet`-Objekt, das dann ausgewertet werden kann.

```
try{
  // Eine Abfrage an die Datenbank richten
  Statement stmt = conn.createStatement();
  ResultSet rs = stmt.executeQuery("SELECT * FROM Kunde;");
  // die Anzahl der Einträge des Datensatzes lesen
  int column = rs.getMetaData().getColumnCount();
  // Datensatz umd Datensatz abarbeiten
  while(rs.next()){
    for (int i=1; i<=column; i++) // Beginnt mit Position 1 !!!
      rs.getString(i); // hier erfolgt die Verarbeitung der Daten
    }
  rs.close();   // nach Gebrauch sind des ResultSet...
  stmt.close(); // ...und das Statement zu schliessen.
}
catch(SQLException err){
  System.out.println("ungültiger SQL-Befehl");
}
```

← SQL-Befehl als String anschreiben

3.1.3.5 Daten schreiben

Daten werden mittels einem `Statement`-Objekt in die Datenbank geschrieben.

```
try{
  // Einen insert an die Datenbank richten
  Statement stmt = conn.createStatement();
  stmt.execute("INSERT INTO Kunde (…, …, …) VALUES ('…', …, '…');");
  stmt.close();
catch(SQLException err){
  System.out.println("ungültiger SQL-Befehl");
}
```

← SQL-Befehl als String anschreiben