Gehen wir von einem Webshop aus, in dem wir Artikel von verschiedenen Lieferanten anbieten.  
  
  
Teil 1: DB  
Als Teil der DB wurden Tabellen angelegt:  
  
CREATE TABLE Lieferanten (  
LieferantId int primary key auto\_increment,  
Name varchar(128) not null unique,  
Email varchar(255) not null,  
Ort varchar(128) not null,  
PLZ varchar(128) not null,  
Strasze varchar(128) not null,  
Hnr varchar(128) not null,  
Land varchar(128) not null  
);  
  
CREATE TABLE Artikel (  
ArtikelId int primary key auto\_increment,  
Bezeichnung varchar(128) not null,  
Beschreibung varchar(20000) not null,  
VerkaufspreisNetto decimal(10,2) not null,  
LieferantId int references Lieferanten(LieferantId)  
);  
…  
  
Die entsprechenden Klassen sind folgendermaßen aufgebaut:  
public class Lieferant {  
private int LieferantId;  
private String Name;  
private String Email;  
private String Ort;  
private String PLZ;  
private String Strasze;  
private String Hnr;  
private String Land;  
  
public Lieferant() { … }  
public Lieferant(  
int LieferantId,  
String Name,  
String Email,  
String Ort,  
String PLZ,  
String Strasze,  
String Hnr,  
String Land) { … }  
// Getter + Setter für jedes Attribut }  
  
  
public class Artikel {  
private int ArtikelId;  
private String Bezeichnung;  
private String Beschreibung;  
private double VerkaufspreisNetto;  
private int LieferantId;  
  
public Artikel() { ... }  
public Artikel (  
int ArtikelId,  
String Bezeichnung,  
String Beschreibung,  
double VerkaufspreisNetto,  
int LieferantId) { … }  
  
// Getter + Setter für jedes Attribut }  
  
  
Ebenso existiert eine Klasse DBManager:  
  
public class DBManager {  
  
…  
public DBManager() { … }  
public Connection getConnection() { … }  
public void releaseConnection() { … }  
…  
}  
  
  
**Aufgabe 1) Datenbanken**Erweitert die Klasse DBManager um die Methoden:  
1a) void speichereArtikel(Connection con, Artikel art)  
1b) Lieferant holeLieferant(Connection con, String name)  
1c) List<String> holeLieferantenNamen(Connection con)  
2d) List<Artikel> holeArtikelVonLieferant(Connection con, Lieferant l)  
  
Diese Routinen laufen innerhalb eines Servers ab.  
Es ist daher wichtig sicherzustellen, dass auch im Fehlerfall keine Ressourcen verschwendet werden.  
=> try - catch - finally  
Nur zur Sicherheit: finally Zweig wird auch nach Aufruf von "return" ausgeführt !  
  
  
**Aufgabe 2) JSP Seiten**a: Artikeluebersicht.jsp

  Lieferanten / Artikel Übersicht  
  
 Text:Lieferant auswählen:     DropDown:LieferantenListe   Button:ArtikelAnzeigen  
------------------------------------------------------------------------------------------  
   Tabelle: Artikel des Lieferanten  
   …  
------------------------------------------------------------------------------------------  
 Button: NeuenArtikelErfassen  
  
================================================================================  
  
und  
b: ArtikelErfassen.jsp  
=================================================================================  
   Artikel erfassen für Lieferanten <Name des Lieferanten>:  
  
   Bezeichnung:    EingabeFeld  
   Beschreibung:   Memo Feld  
   Verkaufspreis  
  
   Button: Speichern  
=================================================================================  
  
(Hoffe es ist verständlich.)  
  
  
Bei der Umsetzung fallen verschiedene Teilaufgaben an. Bitte diese lösen:  
  
**Teilaufgabe 2.1:** Erzeugt die Seite "Artikeluebersicht.jsp":  
Verwendet serverseitig die bereits codierten DB Routinen aus Teil 1.  
Zu codieren sind folgende Teile:  
 - Elemente der Seite anzeigen,  
 - die DropDownListe auf Grund der Werte in der DB  
dynamisch erzeugen.  
  
Gegeben ist:  
Bsp.: DropDown:  
<select>  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="mercedes">Mercedes</option>  
</select>  
  
<%@ page language="java" contentType=" …  
<!DOCTYPE html>

**Teilaufgabe2.2: AJAX**Erweitert die Seite um folgende Funktionalität: Artikel einlesen  
**Schritt 2.2.1: AJAX Request absenden**JavaSkript auf der Seite:  
Absenden der Datenanforderung (client):  
Wenn auf den Button ArtikelAnzeigen geklickt wird,  
wird ein AJAX Request mit dem Namen des ausgewählten  
Lieferanten als Parameter an den Server geschickt.  
  
**Schritt 2.2.2: AJAX Antwort erzeugen (server)**Codiert serverseitig eine Routine(Servlet) "ArtikelVonLieferantenLesen"  
die den Request entgegen nimmt,  
den gewählten Lieferanten (alle Infos zu ihm) in der Session speichert  
und die Artikel des Lieferanten in eine JSON Darstellung als Antwort liefert.  
Verwendet auch hier die zuvor codierten DB Routinen aus Teil 1  
  
**Schritt 2.2.3: AJAX Response**  
Reagieren auf die Antwort (client) => JSON Daten aus Response in die Tabelle einbauen.

**Aufgabe 3: Parameterweitergabe/Form/ Servlets**Abspeichern neuer Artikel  
**Schritt 3.1: Parameter weitergabe**  
Wenn man auf den Button NeuenArtikelErfassen klickt, gelangt man zur Seite ArtikelErfassen.jsp.  
Erweiterung in der Seite Artikeluebersicht.jsp einbauen.  
(ist nicht viel - bitte mit <form> oder <a> lösen -  
Anmerkung: Schaut euch die Möglichkeiten an, Parameter bei <a> mitzugeben)  
  
**Schritt 3.2: Form**  
Erstellt die Seite: ArtikelErfassen.jsp.  
Aufbau der Form mit den Eingabefeldern und dem SubmitButton Speichern.  
Wenn man auf "Speichern" Klickt" wird die Action der Form ausgeführt  
und man gelangt zu Seite "ArtikelSpeichern" (= Wert des Attributs action).  
  
**Schritt 3.3: Servlet**Schreibt die Routine (Servlet in Java) ArtikelSpeichern und öffnet nach erfolgreichem  
Speichern die Seite "Artikeluebersicht.jsp"