



Hochschule Bremerhaven

Software Engineering III

Tassenshop

Autoren:	Tobias Fiedler	Philip Jung	Thorsten Kolling
	MatNr. 31683	MatNr. 31475	MatNr. 31728

Version vom:	April 2, 2017
---------------------	---------------

Betreuer:	Alfred Schmidt
------------------	----------------

Inhalt

1	Projektbeschreibung	3
2	Anforderungsdefinition	3
2.1	Funktionale Anforderungen	3
2.2	Nichtfunktionale Anforderungen	4
3	Klassendiagramm des Modells	5
4	Use-Cases	5
4.1	Registrieren	5
4.2	Kreditkarte ändern	6
4.3	Artikel in den Warenkorb legen	7
5	GUI-Screenshots	8
	Anhang	12

1 Projektbeschreibung

Unser Projekt haben wir stark an den Übungen orientiert und auf dem Ergebnis dieser aufgebaut. Die Datenbank wird durch hibernate in der Option hibernate.hbm2ddl.auto in der Produktion Version auf 'None' konfiguriert. In der Variante 'create' und 'create-drop' werden die Datenbankstabellen bei jedem Start des Programmes sonst neu erstellt. Die Datenbank wird bei erstmaligem Start durch das Laden eines vorher erstellten dump file eingerichtet. Da Tobias viel Erfahrung hat im Bereich Webentwicklung mit Java Enterprise Edition haben wir einige sinnvolle Erweiterungen einbauen können, die so nicht in den Übungen enthalten waren:

- templates für den Seiteninhalt mit festem header und footer
- loginHandler statt sessionHandler
- anderes primefaces theme

Durch die Verwendung von templates muss der Webserver bei Anklicken eines Links weniger Inhalt nachladen und die Geschwindigkeit und Antwortzeit wird durch die Reduzierung des Datenaustausches erhöht. Der sessionHandler kümmert sich neben dem login auch um den Warenkorb des Kunden und die Navigation. Darüber hinaus kann hier auch das dump file für die Datenbank geladen werden. Mit einem geänderten primefaces theme wollen wir uns graphisch von anderen Projekten abheben.

Wir haben uns ebenfalls über die Sicherheit des Programmes Gedanken gemacht. Das Adminpasswort haben wir zu einem 16 Zeichen langen Passwort mit Sonderzeichen, Gross- und Kleinschreibung und Zahlen erweitert. Die Sicherheit der Datenbank vor SQL-Injection ist durch die mit annotations und hibernate erfolgte Einfügung in die Datenbank über prepared statements sichergestellt. Theoretisch dürften die in der Datenbank gespeicherten Passwörter nicht als plain text gespeichert werden, die Sicherheit wäre aber auch nicht viel besser durch das Speichern eines hashcodes (da diese heutzutage auch relativ einfach errechnet werden können), sondern bräuchte ein anderes System. Dies lohnt sich in unserem kleinen Studentenprojekt nicht.

2 Anforderungsdefinition

2.1 Funktionale Anforderungen

Konto erstellen Es muss möglich sein, als Kunde ein Konto zu erstellen. Dieses soll ab Erstellung die Möglichkeit bieten sich einzuloggen. Das soll von überall und immer möglich sein.

Einloggen Wenn ein Konto erstellt wurde, soll es möglich sein, sich in dieses einzuloggen. Dies soll einige Funktionen ermöglichen (siehe weitere Punkte).

Artikel betrachten Es soll für einen Benutzer der Seite möglich sein, die im Shop vorhandenen Artikel anzusehen. Dies soll immer erlaubt sein, unabhängig davon, ob er eingeloggt ist oder nicht.

Artikel bestellen Ein Kunde soll einen Artikel dem Warenkorb hinzufügen und diesen dann einzeln oder mit anderen zusammen bestellen. Dies soll nur möglich sein, wenn der Kunde schon ein Konto hat, in dem auch eine Zahlungsmöglichkeit hinterlegt wurde.

Sprache ändern Es muss möglich sein, die Sprache des Shops zu ändern. Dies soll von jeder Seite des Shops aus und unabhängig davon, ob er eingeloggt ist, machbar sein.

Kreditkarte hinzufügen / ändern Damit ein Benutzer bezahlen kann, muss er eine Bezahlmethode - in diesem Fall eine Kreditkarte - zu seinem Konto hinzufügen können. Außerdem müssen diese Daten geändert werden können, wenn der Benutzer eingeloggt ist.

Kontodaten hinzufügen / ändern Es muss möglich sein, die hinterlegten Daten zu einem Konto zu ändern. Dazu gehören Kreditkarte, Adresse und persönliche Daten. Dies darf nur möglich sein, wenn der Benutzer eingeloggt ist.

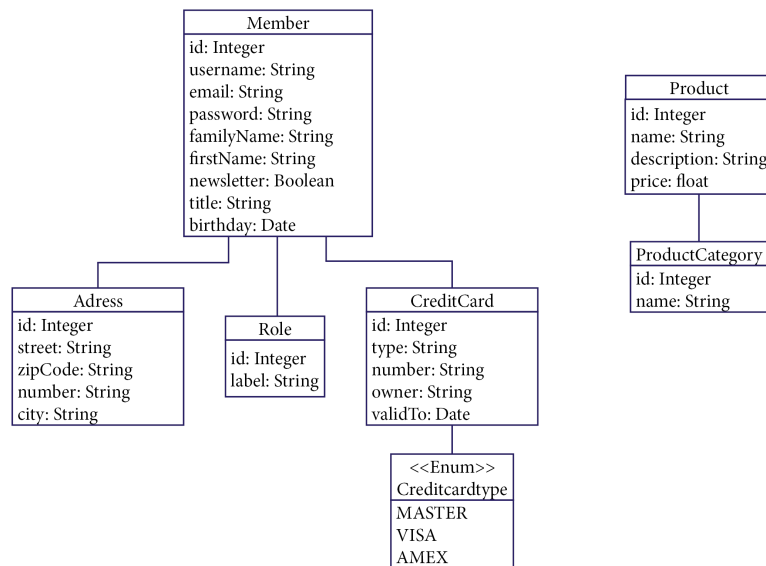
2.2 Nichtfunktionale Anforderungen

Benutzbarkeit Das Programm sollte leicht und intuitiv benutzbar sein. Der Benutzer soll alle Funktionen des Webshops benutzen können, ohne darüber nachdenken zu müssen. Dabei soll es von Seiten des Webshops keine Unklarheiten oder Mehrdeutigkeiten geben. Es muss immer klar sein, wie der Benutzer sein Ziel erreichen kann.

Geschwindigkeit Damit der Webshop angenehm zu bedienen ist, muss sich dementsprechend schnell durch die Seiten bewegt werden können und Aktionen müssen in kürzester Zeit ausgeführt werden. Diese Antwortzeiten dürfen nicht so lange dauern, dass es dem Benutzer negativ auffällt.

Sicherheit Die Sicherheit ist vor allem wichtig um die Daten der Benutzer zu schützen. Dazu gehören die persönlichen Daten wie Name / Geschlecht / Adresse, aber auch die Bezahlmethoden wie hinterlegte Kreditkarten. Es muss also unmöglich sein, von außen an diese Daten zu gelangen. Außerdem darf der Webshop nicht durch Angriffe oder Fehler im System zum Absturz oder Fehlverhalten gebracht werden.

3 Klassendiagramm des Modells



4 Use-Cases

4.1 Registrieren

Name:

Registrieren

Beschreibung:

Ein Kunde möchte einen Benutzeraccount anlegen, um den vollen Umfang der Seite (zum Beispiel Kaufvorgänge) nutzen zu können

Gewünschtes Ergebnis:

Der Kunde besitzt einen Account und wird in diesen eingeloggt

Beteiligte Personen:

Kunde

Bedeutung:

wichtig

Häufigkeit:

mittel

Vorbedingungen:

keine

Ereignisse:

- Kunde klickt auf "Registrieren"
 - ⇒ Die Seite ruft das Formular zum Eintragen der Daten auf
- Kunde trägt die erforderlichen (und evtl. die optionalen) Daten ein und klickt auf "Konto erstellen"
 - ⇒ Die Seite erstellt einen Account mit den eingegebenen Daten und loggt den Kunden ein

4.2 Kreditkarte ändern

Name:

Kreditkarte ändern

Beschreibung:

Ein Kunde will die hinterlegten Daten seiner Kreditkarte ändern

Gewünschtes Ergebnis:

Die alten Daten sind weg, die neuen Daten sind gespeichert

Beteiligte Personen:

Kunde

Bedeutung:

wichtig

Häufigkeit:

selten

Vorbedingungen:

Der Kunde besitzt einen Account und es ist schon mindestens eine Kreditkarte im System hinterlegt

Ereignisse:

- Kunde klickt auf "Login"
 - ⇒ Die Seite zeigt die Login-Abfrage an
- Kunde gibt seinen Benutzernamen und Passwort ein und klickt auf "Login"
 - ⇒ Die Seite loggt den Kunden ein und zeigt die Startseite an
- Kunde klickt auf "Profil bearbeiten"
 - ⇒ Die Seite zeigt die Übersichtsseite "Profil bearbeiten" an
- Der Kunde klickt unter "Zahlungsarten" und "Kreditkarte" auf "Bearbeiten"
 - ⇒ Die Daten der Kreditkarte werden zur Veränderung freigegeben
- Kunde ändert die gewünschten Daten und klickt auf "Speichern"
 - ⇒ Die Seite speichert die Änderungen und zeigt dies an

4.3 Artikel in den Warenkorb legen

Name:

Artikel in den Warenkorb legen

Beschreibung:

Ein Kunde möchte einen Artikel aus dem Shop aussuchen und ihn in den Warenkorb legen, um diesen dann kaufen

Gewünschtes Ergebnis:

Der Artikel soll gekauft werden

Beteiligte Personen:

Kunde

Bedeutung:

wichtig

Häufigkeit:

mittel

Vorbedingungen:

Der Kunde besitzt einen Account und hat eine Zahlungsart und eine Lieferadresse hinterlegt

Ereignisse:

- Kunde klickt auf "Login"
 - ⇒ Die Seite zeigt die Login-Abfrage an

- Kunde gibt seinen Benutzernamen und Passwort ein und klickt auf "Login"
⇒ Die Seite loggt den Kunden ein und zeigt die Startseite an
- Kunde klickt auf "Stöbern"
⇒ Die Seite zeigt alle Produkte in einer mehrseitigen Liste an
- Kunde sucht den gewünschten Artikel aus der Liste aus und drückt dort auf "Zum Warenkorb hinzufügen"
⇒ Die Seite fügt den Artikel zum Warenkorb hinzu und zeigt dies an ("1 Artikel ausgewählt")
- Kunde klickt auf "zum Warenkorb"
⇒ Die Seite zeigt den Warenkorb mit dem Artikel an
- Kunde klickt auf "Kaufabwicklung abschließen"
⇒ Die Seite schließt den Kauf ab und zeigt dies an

5 GUI-Screenshots

Stöbern Alle Kategorien Angebote Hilfe

KONTO ERSTELLEN

Max0815

Herr

Max NullAcht

10.01.1980

Max@web.de

.....

Newsletter abonnieren ☒

Mit Ihrer Anmeldung erklären Sie sich mit
Unseren AGB, unserer
Datenschutzerklärung sowie den
Bestimmungen zu Cookies einverstanden

Konto erstellen

Sie haben schon einen Account? Klicke
Hier zum einloggen

AGB Datenschutzerklärung Impressum 2016-2017, SWE3-Webshop

Figure 1: Konto erstellen

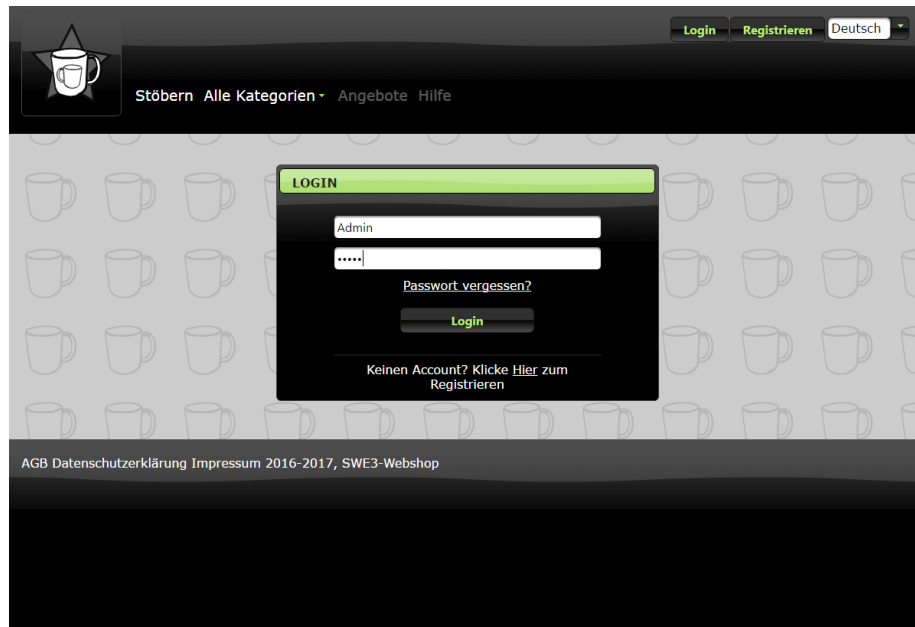


Figure 2: Login

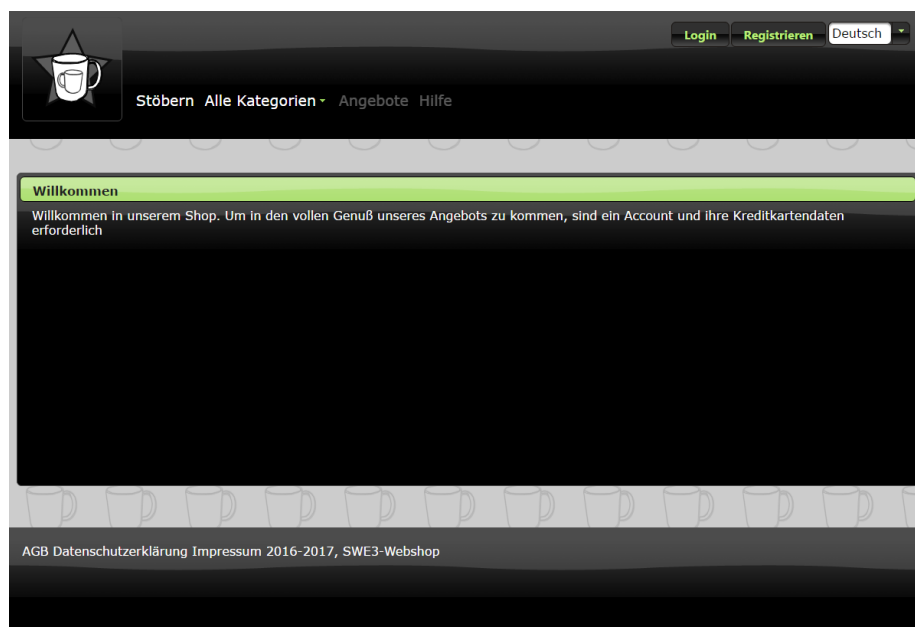


Figure 3: Der Willkommen-Bildschirm

Benutzername:
Admin

Anrede:
Firma

Geburtsdatum:
02.01.1970

Vorname:
John

Nachname:
der Admin

Übernehmen

LIEFERADRESSEN

Straße	Ort	Postleitzahl	
Adminsallee 8e	Adminshaven	937483	Löschen Bearbeiten

Neue Adresse

0 Mar 2017

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Nummer: *
1234567891011213

Inhaber *
John der Admin

Speichern **Abbrechen**

PASSWORT ÄNDERN

Neues Passwort:

Figure 4: Konto bearbeiten

Willkommen, Admin

Kategorie >> Alle

(1 of 2)






	Rote Tasse Formschöner Kaffeebecher in Rot, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maße: Höhe ca 10 cm, Ø ca 8 cm, konische Form 9.99€ / Sofort Lieferbar Zum Warenkorb hinzufügen
	Blaue Tasse Formschöner Kaffeebecher in Blau, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maße: Höhe ca 10 cm, Ø ca 8 cm, konische Form 9.99€ / Sofort Lieferbar Zum Warenkorb hinzufügen
	Gelbe Tasse Formschöner Kaffeebecher in Gelb, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maße: Höhe ca 10 cm, Ø ca 8 cm, konische Form 9.99€ / Sofort Lieferbar Zum Warenkorb hinzufügen
	Punktmuster Tasse Formschöner Kaffeebecher mit Punktmuster, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maße: Höhe ca 10 cm, Ø ca 8 cm, konische Form 11.99€ / Sofort Lieferbar Zum Warenkorb hinzufügen
	Streifenmuster Tasse Formschöner Kaffeebecher mit Streifenmuster, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maße: Höhe ca 10 cm, Ø ca 8 cm, konische Form

Figure 5: Waren auswählen

Willkommen, Admin

Warenkorb und Kaufabwicklung

Artikel			
Produkt	Anzahl	Stückpreis	Gesamtpreis
Rote Tasse	3	EUR 9.99	EUR 29.97

Gezahlt wird mit

Sie brauchen eine gültige Kreditkarte um die Kaufabwicklung abzuschließen

[Profil bearbeiten](#)

Geliefert wird an

Straße:
 Hausnummer:

Straße:
 Postleitzahl:

Bestellungsübersicht

Artikel: **EUR 29.97**

Verpackung und Versand: **EUR 3.99**

Gesamtbetrag: **EUR 33.96**

[Jetzt kaufen](#)
[Warenkorb löschen](#)
[Zurück zur Startseite](#)

Figure 6: Kaufabwicklung

Anhang

Es folgen die .java, .xhtml und .css dateien im Querformat mit Zeilennummerierung in der genannten Reihenfolge.

```

1:  /*
2:  * To change this license header, choose License Headers in Project Properties.
3:  * To change this template file, choose Tools | Templates
4:  * and open the template in the editor.
5:  */
6:  package de.hsb.shop.controller;
7:
8:  import java.io.Serializable;
9:  import java.util.ArrayList;
10: import java.util.GregorianCalendar;
11: import java.util.List;
12:
13: import javax.annotation.PostConstruct;
14: import javax.annotation.Resource;
15: import javax.faces.application.FacesMessage;
16: import javax.faces.bean.ManagedBean;
17: import javax.faces.bean.SessionScoped;
18: import javax.faces.context.FacesContext;
19: import javax.faces.event.ComponentSystemEvent;
20: import javax.persistence.EntityManager;
21: import javax.persistence.PersistenceContext;
22: import javax.persistence.Query;
23: import javax.transaction.UserTransaction;
24:
25: import org.primefaces.model.menu.DefaultMenuItem;
26: import org.primefaces.model.menu.DefaultMenuModel;
27: import org.primefaces.model.menu.DefaultSubMenu;
28: import org.primefaces.model.menu.MenuModel;
29:
30: import de.hsb.shop.model.Adress;
31: import de.hsb.shop.model.Member;
32: import de.hsb.shop.model.Product;
33: import de.hsb.shop.model.ProductCategory;
34: import de.hsb.shop.model.Role;
35: import de.hsb.shop.model.Title;
36:
37: /**
38:  * Verwaltet alle Session-relevante Dinge wie zum Beispiel den Login und den
39:  * Warenkorb
40:  */
41: @ManagedBean(name = "sessionHandler")
42: @SessionScoped
43: public class SessionHandler implements Serializable {
44:     private String username;
45:     private String password;
46:     private Member member;
47:     private String language = "de";
48:     private List<Product> productList;
49:     private List<ProductCategory> categories;
50:     private ArrayList<Product> shoppingCart;
51:     private MenuModel model;
52:     private String productHead = "Alle";
53:
54:     @PersistenceContext
55:     private EntityManager em;
56:     @Resource
57:     private UserTransaction utx;
58:
59:     /**
60:      * Initialisiert den Warenkorb und erstellt das Hauptmenü
61:      */
62:     @PostConstruct
63:     public void init() {
64:         shoppingCart = new ArrayList<>();
65:         // createDump();
66:         createMainMenu();
67:     }

```

```

68:
69:     public boolean isLoggedIn() {
70:         return member != null;
71:     }
72:
73:     /**
74:      * Sucht den Member anhand der Eingaben in der login.xhtml aus der
75:      * Datenbank. Wird er gefunden, wird er eingeloggt.
76:      *
77:      * @return Die Navigation zur Startseite
78:      */
79:     public String login() {
80:         Query query = em.createQuery(
81:             "select m from Member m " + "where LOWER(m.username
e) = :username and m.password = :password");
82:         query.setParameter("username", username.toLowerCase());
83:         query.setParameter("password", password);
84:         List<Member> members = query.getResultList();
85:
86:         if (members.size() == 1) {
87:             member = members.get(0);
88:             return goToStartpage();
89:         } else {
90:             return null;
91:         }
92:     }
93:
94:     /**
95:      * Ã234berprÃ4ft, ob der eingeloggte Member Admin ist.
96:      *
97:      * @return True, wenn er Admin ist, sonst false
98:      */
99:     public boolean isAdmin() {
100:         if (member != null) {
101:             return member.getRole().getLabel().equals("Admin");
102:         }
103:         return false;
104:     }
105:
106:     /**
107:      * Ã234berprÃ4ft, ob der Client eingeloggt ist. Ist er es nicht, wird zur
108:      * Startseite navigiert.
109:      *
110:      * @param ComponentSystemEvent
111:      */
112:     public void checkLoggedIn(ComponentSystemEvent cse) {
113:         FacesContext context = FacesContext.getCurrentInstance();
114:         if (member == null) {
115:             context.getApplication().getNavigationHandler().handleNavigation(context, null, goToStartpage());
116:         }
117:     }
118:
119:     /**
120:      * Terminiert die Session
121:      *
122:      * @return Die Navigation zur Startseite
123:      */
124:     public String logout() {
125:         FacesContext.getCurrentInstance().getExternalContext().invalidateSession();
126:         // warenkorb.clear();
127:         // member = null;
128:         return goToStartpage();
129:     }
130:
131:     public Title[] getTitleValues() {

```

```

132:         return Title.values();
133:     }
134:
135:     /**
136:      * Holt sich alle Produkte aus der Datenbank und navigiert zur Produktseite
e
137:      *
138:      * @return "/products.xhtml?faces-redirect=true"
139:      */
140:     public String goToProductPage() {
141:         productHead = "Alle";
142:         Query query = em.createNamedQuery("SelectProduct", Product.class);
143:         productList = query.getResultList();
144:         return "/products.xhtml?faces-redirect=true";
145:     }
146:
147:     /**
148:      * Sucht sich anhand der Kategorie alle zugehörigen Produkte aus der
149:      * Datenbank und navigiert zur Produktseite
150:      *
151:      * @param category
152:      * @return "/products.xhtml?faces-redirect=true"
153:      */
154:     public String goToProductPage(String category) {
155:         productHead = category;
156:         Query query = em.createNamedQuery("SelectProductByProductCategory"
, Product.class);
157:         query.setParameter("productCategory", category);
158:         productList = query.getResultList();
159:         return "/products.xhtml?faces-redirect=true";
160:     }
161:
162:     /**
163:      * Leert den Warenkorb
164:      *
165:      * @return "/shoppingCart.xhtml?faces-redirect=true"
166:      */
167:     public String emptyShoppingCart() {
168:         shoppingCart.clear();
169:         return "/shoppingCart.xhtml?faces-redirect=true";
170:     }
171:
172:     /**
173:      * Generiert anhand der existenten Produktkategorien das Auswahlmenü für
die
174:      * Navigation zu den Produkten.
175:      */
176:     private void createMainMenu() {
177:         Query query = em.createNamedQuery("SelectProductCategory", Product
Category.class);
178:         categorys = query.getResultList();
179:
180:         model = new DefaultMenuModel();
181:
182:         DefaultMenuItem all = new DefaultMenuItem("Steuern");
183:         all.setCommand("#{sessionHandler.goToProductPage()}");
184:         model.addElement(all);
185:
186:         // First submenu
187:         DefaultSubMenu categorySubMenu = new DefaultSubMenu("Alle Kategori
en");
188:
189:         for (ProductCategory pc : categorys) {
190:             DefaultMenuItem item = new DefaultMenuItem(pc.getName());
191:             item.setCommand("#{sessionHandler.goToProductPage(' " + pc.
getName() + "')}");
192:             categorySubMenu.addElement(item);
193:         }
194:
195:         model.addElement(categorySubMenu);
196:
197:         DefaultMenuItem redu = new DefaultMenuItem("Angebote");
198:         redu.setDisabled(true);
199:         model.addElement(redu);
200:
201:         DefaultMenuItem help = new DefaultMenuItem("Hilfe");
202:         help.setDisabled(true);
203:         model.addElement(help);
204:     }
205:
206:     /**
207:      * Fügt dem Warenkorb ein Produkt hinzu
208:      *
209:      * @param product
210:      */
211:     public void putProductIntoShoppingCart(Product p) {
212:         FacesContext context = FacesContext.getCurrentInstance();
213:         context.addMessage(null, new FacesMessage("", p.getName() + " wurd
e hinzugefügt"));
214:     }
215:
216:     /**
217:      * Navigiert zur Startseite
218:      *
219:      * @return "/startpage.xhtml?faces-redirect=true"
220:      */
221:     public String goToStartpage() {
222:         return "/startpage.xhtml?faces-redirect=true";
223:     }
224:
225:     /**
226:      * Erstellt eine temporäre Datenbank. Ist im Endprodukt irrelevant.
227:      */
228:     public void createDump() {
229:         try {
230:             utx.begin();
231:             ProductCategory mu = new ProductCategory("Muster");
232:             em.persist(mu);
233:             utx.commit();
234:             utx.begin();
235:             ProductCategory ei = new ProductCategory("Einfarbig");
236:             em.persist(ei);
237:             utx.commit();
238:
239:             utx.begin();
240:             ProductCategory me = new ProductCategory("Merchandise");
241:             em.persist(me);
242:             utx.commit();
243:
244:             utx.begin();
245:             Product pro = new Product("Rote Tasse");
246:             pro.setProductCategory(ei);
247:             pro.setDescription(
"Formschöner Kaffeebecher in Rot, Fassung
svernägen: 325 ml, für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß: 10 cm, 230 ca 8 cm, konische Form");
248:             pro.setPrice(9.99f);
249:             em.persist(pro);
250:             utx.commit();
251:
252:             utx.begin();
253:             pro = new Product("Blaue Tasse");
254:             pro.setProductCategory(ei);
255:             pro.setDescription(
"Formschöner Kaffeebecher in Blau, Fassun
256:

```

```

gsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß: H
Äße ca 10 cm, Ä\230 ca 8 cm, konische Form");
257:         pro.setPrice(9.99f);
258:         em.persist(pro);
259:         utx.commit();
260:
261:         utx.begin();
262:         pro = new Product("Grüne Tasse");
263:         pro.setProductCategory(ei);
264:         pro.setDescription(
265:             "Formschöner Kaffeebecher in Grün, Fassu
ngsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß:
HÄße ca 10 cm, Ä\230 ca 8 cm, konische Form");
266:         pro.setPrice(9.99f);
267:         em.persist(pro);
268:         utx.commit();
269:
270:         utx.begin();
271:         pro = new Product("Gelbe Tasse");
272:         pro.setProductCategory(ei);
273:         pro.setDescription(
274:             "Formschöner Kaffeebecher in Gelb, Fassu
ngsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß: H
Äße ca 10 cm, Ä\230 ca 8 cm, konische Form");
275:         pro.setPrice(9.99f);
276:         em.persist(pro);
277:         utx.commit();
278:
279:         utx.begin();
280:         pro = new Product("Schwarze Tasse");
281:         pro.setProductCategory(ei);
282:         pro.setDescription(
283:             "Formschöner Kaffeebecher in Schwarz, Fas
sungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß
: HÄße ca 10 cm, Ä\230 ca 8 cm, konische Form");
284:         pro.setPrice(9.99f);
285:         em.persist(pro);
286:         utx.commit();
287:
288:         utx.begin();
289:         pro = new Product("Punktmuster Tasse");
290:         pro.setProductCategory(mu);
291:         pro.setDescription(
292:             "Formschöner Kaffeebecher mit Punktmuster
, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke, Maß
\237e: HÄße ca 10 cm, Ä\230 ca 8 cm, konische Form");
293:         pro.setPrice(11.99f);
294:         em.persist(pro);
295:         utx.commit();
296:
297:         utx.begin();
298:         pro = new Product("Tassentasse");
299:         pro.setProductCategory(mu);
300:         pro.setDescription(
301:             "Formschöner Premium Kaffeebecher mit ein
er Tassenabbildung, Fassungsvermögen: 700 ml, Für Kaffee, Tee und andere Heiß- und
Kaltgetränke, Maß: HÄße ca 20 cm, Ä\230 ca 8 cm, konische Form");
302:         pro.setPrice(39.99f);
303:         em.persist(pro);
304:         utx.commit();
305:
306:         utx.begin();
307:         pro = new Product("Streifenmuster Tasse");
308:         pro.setProductCategory(mu);
309:         pro.setDescription(
310:             "Formschöner Kaffeebecher mit Streifenmus
ter, Fassungsvermögen: 325 ml, Für Kaffee, Tee und andere Heiß- und Kaltgetränke,
Maß: HÄße ca 10 cm, Ä\230 ca 8 cm, konische Form");
311:         pro.setPrice(12.72f);
312:         em.persist(pro);
313:         utx.commit();
314:
315:         utx.begin();
316:         pro = new Product("Tassenpullover");
317:         pro.setProductCategory(me);
318:         pro.setDescription("Ein Pullover mit einem feschen Aufdruc
k einer Tasse");
319:         pro.setPrice(23.99f);
320:         em.persist(pro);
321:         utx.commit();
322:
323:         utx.begin();
324:         pro = new Product("Tassenhose");
325:         pro.setProductCategory(me);
326:         pro.setDescription("Eine Hose in Form einer Tasse. Jetzt a
uch mit Griff");
327:         pro.setPrice(29.99f);
328:         em.persist(pro);
329:         utx.commit();
330:
331:         utx.begin();
332:         pro = new Product("Tassenschuhe");
333:         pro.setProductCategory(me);
334:         pro.setDescription("Schuhe mit Henkel. Damit fählt sich j
eder Untergrund wie Porzellan an");
335:         pro.setPrice(24.99f);
336:         em.persist(pro);
337:         utx.commit();
338:
339:         utx.begin();
340:         Role rl = new Role("Member");
341:         em.persist(rl);
342:         utx.commit();
343:
344:         utx.begin();
345:         Adress a = new Adress("Adminsallee", "937483", "Adminshave
n", "8e");
346:         em.persist(a);
347:         Role r2 = new Role("Admin");
348:         em.persist(r2);
349:         Member m = new Member("Admin", "John", "der Admin", "admin
", "Admin@webshop.de",
350:             new GregorianCalendar(1970, 0, 2).getTime()
);
351:         m.setTitle(Title.FIRMA.toString());
352:         ArrayList<Adress> al = new ArrayList<>();
353:         al.add(a);
354:         m.setAdressList(al);
355:         m.setRole(r2);
356:         m.setNewsletter(true);
357:         em.persist(m);
358:         utx.commit();
359:     } catch (Exception e) {
360:         e.printStackTrace();
361:     }
362: }
363:
364: public String getUsername() {
365:     return username;
366: }
367:
368: public void setUsername(String username) {
369:     this.username = username;
370: }

```

```
371:
372:     public String getPassword() {
373:         return password;
374:     }
375:
376:     public void setPassword(String password) {
377:         this.password = password;
378:     }
379:
380:     public Member getMember() {
381:         return member;
382:     }
383:
384:     public void setMember(Member member) {
385:         this.member = member;
386:     }
387:
388:     public String getLanguage() {
389:         return language;
390:     }
391:
392:     public void setLanguage(String language) {
393:         this.language = language;
394:     }
395:
396:     public List<Product> getProductList() {
397:         return productList;
398:     }
399:
400:     public void setProductList(List<Product> productList) {
401:         this.productList = productList;
402:     }
403:
404:     public List<ProductCategory> getCategorys() {
405:         return categorys;
406:     }
407:
408:     public void setCategorys(List<ProductCategory> categorys) {
409:         this.categorys = categorys;
410:     }
411:
412:     public ArrayList<Product> getShoppingCart() {
413:         return shoppingCart;
414:     }
415:
416:     public void setShoppingCart(ArrayList<Product> shoppingCart) {
417:         this.shoppingCart = shoppingCart;
418:     }
419:
420:     public MenuModel getModel() {
421:         return model;
422:     }
423:
424:     public void setModel(MenuModel model) {
425:         this.model = model;
426:     }
427:
428:     public String getProductHead() {
429:         return productHead;
430:     }
431:
432:     public void setProductHead(String productHead) {
433:         this.productHead = productHead;
434:     }
435:
436: }
```



```

1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6:  package de.hsb.shop.controller;
7:
8:  import java.io.IOException;
9:  import java.io.Serializable;
10: import java.util.logging.Level;
11: import java.util.logging.Logger;
12:
13: import javax.annotation.PostConstruct;
14: import javax.annotation.Resource;
15: import javax.faces.application.FacesMessage;
16: import javax.faces.bean.ManagedBean;
17: import javax.faces.bean.ManagedProperty;
18: import javax.faces.bean.ViewScoped;
19: import javax.faces.context.FacesContext;
20: import javax.persistence.EntityManager;
21: import javax.persistence.PersistenceContext;
22: import javax.transaction.UserTransaction;
23:
24: import de.hsb.shop.model.Adress;
25: import de.hsb.shop.model.Creditcard;
26: import de.hsb.shop.model.Creditcardtype;
27: import de.hsb.shop.model.Member;
28:
29: /**
30:  * Verwaltet die Profile.xhtml
31:  */
32: @ManagedBean
33: @ViewScoped
34: public class ProfileHandler implements Serializable {
35:
36:     @ManagedProperty(value = "#{sessionHandler}")
37:     private SessionHandler sessionHandler;
38:
39:     @PersistenceContext
40:     private EntityManager em;
41:     @Resource
42:     private UserTransaction utx;
43:
44:     private Member member;
45:
46:     private Boolean addAdress;
47:     private Boolean addCreditCard;
48:
49:     private Adress tmpAdress;
50:     private Creditcard tmpCreditcard;
51:
52:     /**
53:      * Holt den zu bearbeitenden Member aus dem Session-Handler
54:      */
55:     @PostConstruct
56:     public void init() {
57:         if (!sessionHandler.isLogged()) {
58:             try {
59:                 FacesContext.getCurrentInstance().getExternalConte
xt().redirect("startpage.xhtml?faces-redirect=true");
60:             } catch (IOException ex) {
61:
62:             }
63:             member = sessionHandler.getMember();
64:         }
65:
66:         /**
67:          * Speichert die Ã\204nderungen der Daten des Members in die Datenbank und
68:          * navigiert anschlieÃ\237end zur profile.xhtml
69:          */
70:         * @return "profil.xhtml?faces-redirect=true"
71:         */
72:         public String update() {
73:             try {
74:                 utx.begin();
75:                 member = em.merge(member);
76:                 em.persist(member);
77:                 utx.commit();
78:                 FacesContext.getCurrentInstance().addMessage(null,
79:                     new FacesMessage(FacesMessage.SEVERITY_INF
O, "", "Ã\204nderungen gespeichert"));
80:             } catch (Exception ex) {
81:                 Logger.getLogger(ProfileHandler.class.getName()).log(Level
.SEVERE, null, ex);
82:             }
83:             return "profil.xhtml?faces-redirect=true";
84:         }
85:
86:         public void createAdress() {
87:             tmpAdress = new Adress();
88:             addAdress = true;
89:         }
90:
91:         public void editAdress(Adress adress) {
92:             tmpAdress = adress;
93:             addAdress = true;
94:         }
95:
96:         public void createCreditCard() {
97:             tmpCreditcard = new Creditcard();
98:             addCreditCard = true;
99:         }
100:
101:         public void editCreditCard(Creditcard creditCard) {
102:             tmpCreditcard = creditCard;
103:             addCreditCard = true;
104:         }
105:
106:         /**
107:          * Speichert die Adresse in der Datenbank und fÃ\228gt sie dem Member hinzu
108:          */
109:         * @return "profil.xhtml?faces-redirect=true"
110:         */
111:         public String saveAdress() {
112:             try {
113:                 utx.begin();
114:                 if (!member.getAdressList().contains(tmpAdress)) {
115:                     member.getAdressList().add(tmpAdress);
116:                 }
117:                 member = em.merge(member);
118:                 em.persist(member);
119:                 utx.commit();
120:                 addAdress = false;
121:                 FacesContext.getCurrentInstance().addMessage(null,
122:                     new FacesMessage(FacesMessage.SEVERITY_INF
O, "", "Ã\204nderungen gespeichert"));
123:             } catch (Exception ex) {
124:                 Logger.getLogger(ProfileHandler.class.getName()).log(Level
.SEVERE, null, ex);
125:             }
126:             return "profil.xhtml?faces-redirect=true";
127:         }
128:
129:         /**

```

```
130:      * LÃ¶scht die Ã¼bergene Adresse aus der Datenbank
131:      *
132:      * @param address
133:      * @return "profil.xhtml?faces-redirect=true"
134:      */
135:      public String deleteAddress(Adress address) {
136:          try {
137:              utx.begin();
138:              em.remove(em.merge(address));
139:              member.getAddressList().remove(address);
140:              utx.commit();
141:          } catch (Exception ex) {
142:              Logger.getLogger(ProfileHandler.class.getName()).log(Level
143:              .SEVERE, null, ex);
144:          }
145:          return "profil.xhtml?faces-redirect=true";
146:      }
147:      /**
148:      * Speichert die Kreditkarte in der Datenbank und fÃ¼gt sie dem Member hin
149:      zu
150:      *
151:      * @return "profil.xhtml?faces-redirect=true"
152:      */
153:      public String saveCreditcard() {
154:          try {
155:              member.setCreditcard(tmpCreditcard);
156:              utx.begin();
157:              member = em.merge(member);
158:              tmpCreditcard = em.merge(tmpCreditcard);
159:              em.persist(member);
160:              // em.persist(tmpCreditcard);
161:              utx.commit();
162:              addCreditCard = false;
163:          } catch (Exception ex) {
164:              Logger.getLogger(ProfileHandler.class.getName()).log(Level
165:              .SEVERE, null, ex);
166:          }
167:          return "profil.xhtml?faces-redirect=true";
168:      }
169:      public Creditcardtype[] getCreditcardtypeValues() {
170:          return Creditcardtype.values();
171:      }
172:      public SessionHandler getSessionhandler() {
173:          return sessionhandler;
174:      }
175:      public void setSessionhandler(SessionHandler sessionhandler) {
176:          this.sessionhandler = sessionhandler;
177:      }
178:      public Member getMember() {
179:          return member;
180:      }
181:      public void setMember(Member member) {
182:          this.member = member;
183:      }
184:      public Boolean getAddAddress() {
185:          return addAddress;
186:      }
187:      public void setAddAddress(Boolean addAddress) {
188:          this.addAddress = addAddress;
189:      }
```

```
194:      }
195:      public Boolean getAddCreditCard() {
196:          return addCreditCard;
197:      }
198:      public void setAddCreditCard(Boolean addCreditCard) {
199:          this.addCreditCard = addCreditCard;
200:      }
201:      public Address getTmpAddress() {
202:          return tmpAddress;
203:      }
204:      public void setTmpAddress(Adress tmpAddress) {
205:          this.tmpAddress = tmpAddress;
206:      }
207:      public Creditcard getTmpCreditcard() {
208:          return tmpCreditcard;
209:      }
210:      public void setTmpCreditcard(Creditcard tmpCreditcard) {
211:          this.tmpCreditcard = tmpCreditcard;
212:      }
213:      }
214:      }
215:      }
216:      }
217:      }
218:      }
219:      }
220:      }
```

```

1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6:  package de.hsb.shop.controller;
7:
8:  import java.io.Serializable;
9:  import java.util.Collection;
10: import java.util.HashMap;
11:
12: import javax.annotation.PostConstruct;
13: import javax.faces.bean.ManagedBean;
14: import javax.faces.bean.ManagedProperty;
15: import javax.faces.bean.ViewScoped;
16:
17: import de.hsb.shop.model.Product;
18: import de.hsb.shop.utils.Utills;
19: import de.hsb.shop.utils.shoppingCartSummary;
20:
21: /**
22:  * Erwartet die shoppingCart.xhtml
23:  */
24: @ManagedBean
25: @ViewScoped
26: public class shoppingCartController implements Serializable {
27:
28:     @ManagedProperty(value = "#{sessionHandler}")
29:     private SessionHandler sessionHandler;
30:     private Collection<shoppingCartSummary> shList;
31:     private boolean finished;
32:     private double articleSummaryPrice;
33:     private double finalPrice;
34:
35:     /**
36:      *
37:      */
38:     @PostConstruct
39:     public void init() {
40:         finished = false;
41:         HashMap<Integer, shoppingCartSummary> shMap = new HashMap<Integer, shopping
42:         CartSummary>();
43:         for (Product p : sessionHandler.getShoppingCart()) {
44:             if (shMap.containsKey(p.getId())) {
45:                 shMap.get(p.getId()).increment();
46:             } else {
47:                 shMap.put(p.getId(), new shoppingCartSummary(p));
48:             }
49:             double deliveryCosts = 3.99;
50:             shList = shMap.values();
51:             articleSummaryPrice = 0;
52:             for (shoppingCartSummary s : shList) {
53:                 articleSummaryPrice += s.getWholePrice();
54:             }
55:             articleSummaryPrice = Utills.round(articleSummaryPrice,2);
56:             finalPrice = Utills.round(articleSummaryPrice+deliveryCosts,2);
57:         }
58:
59:         public void finishShopping() {
60:             finished = true;
61:             sessionHandler.getShoppingCart().clear();
62:         }
63:
64:         public Collection<shoppingCartSummary> getShList() {
65:             return shList;
66:         }

```

```

67:
68:     public void setShList(Collection<shoppingCartSummary> shList) {
69:         this.shList = shList;
70:     }
71:
72:     public boolean isFinished() {
73:         return finished;
74:     }
75:
76:     public void setFinished(boolean finished) {
77:         this.finished = finished;
78:     }
79:
80:     public double getWholePrice() {
81:         return articleSummaryPrice;
82:     }
83:
84:     public void setWholePrice(double wholePrice) {
85:         this.articleSummaryPrice = wholePrice;
86:     }
87:
88:     public SessionHandler getSessionHandler() {
89:         return sessionHandler;
90:     }
91:
92:     public void setSessionHandler(SessionHandler sessionHandler) {
93:         this.sessionHandler = sessionHandler;
94:     }
95:
96:     public double getFinalPrice() {
97:         return finalPrice;
98:     }
99:
100:    public void setFinalPrice(double finalPrice) {
101:        this.finalPrice = finalPrice;
102:    }
103:
104: }

```

```

1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6:  package de.hsb.shop.controller;
7:
8:  import java.io.IOException;
9:  import java.io.Serializable;
10: import java.util.logging.Level;
11: import java.util.logging.Logger;
12:
13: import javax.annotation.PostConstruct;
14: import javax.annotation.Resource;
15: import javax.faces.bean.ManagedBean;
16: import javax.faces.bean.ManagedProperty;
17: import javax.faces.bean.ViewScoped;
18: import javax.faces.context.FacesContext;
19: import javax.persistence.EntityManager;
20: import javax.persistence.PersistenceContext;
21: import javax.persistence.Query;
22: import javax.transaction.UserTransaction;
23:
24: import de.hsb.shop.model.Member;
25: import de.hsb.shop.model.Role;
26:
27: /**
28:  * Verwaltet die register.xhtml
29:  */
30: @ManagedBean
31: @ViewScoped
32: public class RegisterHandler implements Serializable {
33:
34:     @ManagedProperty(value = "#{sessionHandler}")
35:     private SessionHandler sessionHandler;
36:
37:     @PersistenceContext
38:     private EntityManager em;
39:     @Resource
40:     private UserTransaction utx;
41:
42:     private Member member;
43:
44:     /**
45:      * Bereitet einen neuen Member vor und holt die Rolle Member aus der
46:      * Datenbank und fÃ¼gt sie ihm hinzu
47:      */
48:     @PostConstruct
49:     public void init() {
50:         if (sessionHandler.isLogged()) {
51:             try {
52:                 FacesContext.getCurrentInstance().getExternalConte
xt().redirect("startpage.xhtml?faces-redirect=true");
53:             } catch (IOException ex) {
54:             }
55:         }
56:         member = new Member();
57:         Query query = em.createQuery("select r from Role r where r.label =
'Member'");
58:         Role r = (Role) query.getSingleResult();
59:         member.setRole(r);
60:     }
61:
62:     /**
63:      * Speichert den Member in die Datenbank und setzt diesen als aktiven
64:      * eingeloggten Member in der Session
65:      */
66:     * @return Die Navigation zur Startseite
67:     */
68:     public String saveAndLogin() {
69:         try {
70:             utx.begin();
71:             member = em.merge(member);
72:             em.persist(member);
73:             utx.commit();
74:         } catch (Exception ex) {
75:             Logger.getLogger(RegisterHandler.class.getName()).log(Leve
l.SEVERE, null, ex);
76:         }
77:         sessionHandler.setUsername(member.getUsername());
78:         sessionHandler.setPassword(member.getPassword());
79:         return sessionHandler.login();
80:     }
81:
82:     public Member getMerkeMember() {
83:         return member;
84:     }
85:
86:     public void setMerkeMember(Member merkeMember) {
87:         this.member = merkeMember;
88:     }
89:
90:     public Member getMember() {
91:         return member;
92:     }
93:
94:     public void setMember(Member member) {
95:         this.member = member;
96:     }
97:
98:     public SessionHandler getSessionHandler() {
99:         return sessionHandler;
100:     }
101:
102:     public void setSessionHandler(SessionHandler sessionHandler) {
103:         this.sessionHandler = sessionHandler;
104:     }
105: }

```

```
1: /*
2:  * To change this license header, choose License Headers in Project Properties.
3:  * To change this template file, choose Tools | Templates
4:  * and open the template in the editor.
5:  */
6: package de.hsb.shop.utils;
7:
8: import de.hsb.shop.model.Product;
9:
10: public class shoppingCartSummary {
11:
12:     Product product;
13:
14:     Integer count;
15:
16:     private double wholePrice;
17:
18:     public shoppingCartSummary(Product product) {
19:         this.product = product;
20:         this.count = 1;
21:         if (product.getPrice() != null) {
22:             this.wholePrice = product.getPrice();
23:         } else {
24:             this.wholePrice = 0;
25:         }
26:         wholePrice = Utils.round(wholePrice,2);
27:     }
28:
29:     public void increment() {
30:         ++count;
31:         if (product.getPrice() != null) {
32:             wholePrice += product.getPrice();
33:         }
34:         wholePrice = Utils.round(wholePrice,2);
35:     }
36:
37:     public Product getProduct() {
38:         return product;
39:     }
40:
41:     public void setProduct(Product product) {
42:         this.product = product;
43:     }
44:
45:     public Integer getCount() {
46:         return count;
47:     }
48:
49:     public void setCount(Integer count) {
50:         this.count = count;
51:     }
52:
53:     public double getWholePrice() {
54:         return wholePrice;
55:     }
56:
57:     public void setWholePrice(double wholePrice) {
58:         this.wholePrice = wholePrice;
59:     }
60: }
```

```
1: /*
2:  * To change this license header, choose License Headers in Project Properties.
3:  * To change this template file, choose Tools | Templates
4:  * and open the template in the editor.
5:  */
6: package de.hsb.shop.utils;
7:
8: /**
9:  *
10:  * @author Tobi
11:  */
12: public class Utils {
13:
14:     public static double round(double value, int places) {
15:         if (places < 0) {
16:             throw new IllegalArgumentException();
17:         }
18:
19:         long factor = (long) Math.pow(10, places);
20:         value = value * factor;
21:         long tmp = Math.round(value);
22:         return (double) tmp / factor;
23:     }
24: }
```

```
1: package de.hsb.shop.validator;
2:
3: import java.util.List;
4:
5: import javax.faces.application.FacesMessage;
6: import javax.faces.bean.ManagedBean;
7: import javax.faces.bean.RequestScoped;
8: import javax.faces.component.UIComponent;
9: import javax.faces.context.FacesContext;
10: import javax.faces.validator.Validator;
11: import javax.faces.validator.ValidatorException;
12: import javax.persistence.EntityManager;
13: import javax.persistence.PersistenceContext;
14: import javax.persistence.Query;
15:
16: import de.hsb.shop.model.Member;
17:
18: @ManagedBean
19: @RequestScoped
20: //@FacesValidator("usernameValidator")
21: public class UsernameValidator implements Validator {
22:
23:     @PersistenceContext
24:     private EntityManager em;
25:
26:     @Override
27:     public void validate(FacesContext context, UIComponent component, Object value
) throws ValidatorException {
28:         String username = (String) value;
29:         System.out.println(em);
30:         Query query = em.createNamedQuery("SelectMember", Member.class);
31:         List<Member> memberList = query.getResultList();
32:
33:         boolean invalid = false;
34:         for (Member m : memberList) {
35:             if (m.getUsername().toLowerCase().equals(username.toLowerCase())) {
36:                 invalid = true;
37:             }
38:         }
39:
40:         if (invalid) {
41:             FacesMessage message = new FacesMessage(FacesMessage.SEVERITY_ERROR,
42:                 "", "Der Name ist bereits vergeben");
43:             throw new ValidatorException(message);
44:         } else if (username.length() < 4) {
45:             FacesMessage message = new FacesMessage(FacesMessage.SEVERITY_ERROR,
46:                 "", "Benutzername muss mindestens 4 Zeichen haben");
47:             throw new ValidatorException(message);
48:         } else if (username.length() > 30) {
49:             FacesMessage message = new FacesMessage(FacesMessage.SEVERITY_ERROR,
50:                 "", "Benutzername darf höchstens 30 Zeichen haben");
51:             throw new ValidatorException(message);
52:         }
53:     }
54: }
```

```
1: package de.hsb.shop.validator;
2:
3: import java.util.ArrayList;
4:
5: import javax.faces.application.FacesMessage;
6: import javax.faces.component.UIComponent;
7: import javax.faces.context.FacesContext;
8: import javax.faces.validator.FacesValidator;
9: import javax.faces.validator.Validator;
10: import javax.faces.validator.ValidatorException;
11:
12: @FacesValidator(value = "creditcardValidator")
13: public class CreditcardValidator implements Validator{
14:
15:     private boolean isValid(ArrayList<Integer> digits) {
16:         int sum=0;
17:         int length = digits.size();
18:         for (int i=0; i < length; i++) {
19:             Integer digit = digits.get(length-i-1);
20:             // jede 2. Ziffer multipliziert mit 2
21:             if (i % 2 == 1) digit *= 2;
22:             sum += digit > 9 ? digit - 9 : digit;
23:         }
24:         return sum % 10 == 0;
25:     }
26:
27:     public void validate(FacesContext context, UIComponent component,
28:         Object value) throws ValidatorException {
29:         String kknr = String.valueOf(value);
30:         ArrayList<Integer> kkintarray = new ArrayList<Integer>();
31:         FacesMessage message;
32:         // ArrayList für Luhn initialisieren
33:         for (int i=0; i<kknr.length(); i++)
34:             kkintarray.add(kknr.charAt(i)-48); //ASCII -> Integer
35:         if (! kknr.matches("[0-9]+")) {
36:
37:             message = new FacesMessage(
38:                 FacesMessage.SEVERITY_ERROR,
39:                 "Ungültige Kreditkartennummer",
40:                 "Es dürfen nur Ziffern enthalten sein!");
41:             throw new ValidatorException(message);
42:         } else if (kknr.length() < 12 || kknr.length() > 16) {
43:             message = new FacesMessage(
44:                 FacesMessage.SEVERITY_ERROR,
45:                 "Ungültige Kreditkartennummer",
46:                 "Nummer ist zu lang oder zu kurz!");
47:             throw new ValidatorException(message);
48:         } else if (! isValid(kkintarray)) {
49:             message = new FacesMessage(
50:                 FacesMessage.SEVERITY_ERROR,
51:                 "Ungültige Kreditkartennummer",
52:                 "Die Kreditkartennummer ist ungültig!");
53:             throw new ValidatorException(message);
54:         }
55:     }
56: }
57:
58: }
```



```

1: package de.hsb.shop.converter;
2:
3: import javax.faces.component.UIComponent;
4: import javax.faces.context.FacesContext;
5: import javax.faces.convert.Converter;
6: import javax.faces.convert.FacesConverter;
7:
8: @FacesConverter(value = "creditcardConverter")
9: public class CreditcardConverter implements Converter {
10:
11:     public Object getAsObject(FacesContext context, UIComponent component, String
value) {
12:         if (value == null) {
13:             return null;
14:         }
15:         String s = value;
16:         if (value.contains(" ")) {
17:             s = value.replace(" ", ""); // Leerzeichen entfernen
18:         }
19:         if (value.contains("-")) {
20:             s = value.replace("-", ""); // Trennstrich entfernen
21:         }
22:         return s;
23:     }
24:
25:     public String getAsString(FacesContext context, UIComponent component, Object
value) {
26:         return (String) value;
27:     }
28:
29: }

```

```
1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6:  package de.hsb.shop.model;
7:
8:  import java.io.Serializable;
9:
10: import javax.persistence.Column;
11: import javax.persistence.Entity;
12: import javax.persistence.GeneratedValue;
13: import javax.persistence.GenerationType;
14: import javax.persistence.Id;
15: import javax.persistence.NamedQuery;
16: import javax.validation.constraints.Size;
17:
18: import org.slf4j.Logger;
19: import org.slf4j.LoggerFactory;
20:
21: import de.hsb.shop.controller.SessionHandler;
22:
23: @NamedQuery(name = "SelectAdress", query = "Select m from Adress m")
24: @Entity
25: public class Adress implements Serializable {
26:
27:     private static Logger logger = LoggerFactory.getLogger(SessionHandler.class);
28:     private static final long serialVersionUID = 1L;
29:
30:     @Id
31:     @GeneratedValue(strategy = GenerationType.AUTO)
32:     @Column(name = "id")
33:     private Integer id;
34:
35:     @Size(min = 3, max = 30)
36:     private String street;
37:
38:     private String zipCode;
39:
40:     @Size(max = 10)
41:     private String number;
42:
43:     @Size(min = 3, max = 30)
44:     private String city;
45:
46:     public Adress() {}
47:
48:     public Adress(String street, String zipCode, String city, String number){
49:         this.street = street;
50:         this.zipCode = zipCode;
51:         this.city = city;
52:         this.number = number;
53:     }
54:
55:     public Integer getId() {
56:         return id;
57:     }
58:
59:     public void setId(Integer id) {
60:         this.id = id;
61:     }
62:
63:     public String getStreet() {
64:         return street;
65:     }
66:
67:     public void setStreet(String street) {
```

```
68:         this.street = street;
69:     }
70:
71:     public String getZipCode() {
72:         return zipCode;
73:     }
74:
75:     public void setZipCode(String zipCode) {
76:         this.zipCode = zipCode;
77:     }
78:
79:     public String getCity() {
80:         return city;
81:     }
82:
83:     public void setCity(String city) {
84:         this.city = city;
85:     }
86:
87:     public String getNumber() {
88:         return number;
89:     }
90:
91:     public void setNumber(String number) {
92:         this.number = number;
93:     }
94:
95: }
```

```
1:  /*                                     67:      }
2:  * To change this license header, choose License Headers in Project Properties. 68:  }
3:  * To change this template file, choose Tools | Templates
4:  * and open the template in the editor.
5:  */
6:  package de.hsb.shop.model;
7:
8:  import java.io.Serializable;
9:  import java.util.List;
10:
11:  import javax.persistence.CascadeType;
12:  import javax.persistence.Column;
13:  import javax.persistence.Entity;
14:  import javax.persistence.GeneratedValue;
15:  import javax.persistence.GenerationType;
16:  import javax.persistence.Id;
17:  import javax.persistence.NamedQuery;
18:  import javax.persistence.OneToMany;
19:  import javax.validation.constraints.Size;
20:
21:  @NamedQuery(name = "SelectProductCategory", query = "Select s from ProductCategory
s")
22:  @Entity
23:  public class ProductCategory implements Serializable{
24:
25:      private static final long serialVersionUID = 1L;
26:
27:      @Id
28:      @GeneratedValue(strategy = GenerationType.AUTO)
29:      @Column(name = "id")
30:      private Integer id;
31:
32:      @Size(max = 255)
33:      @Column(name = "name")
34:      private String name;
35:
36:      @OneToMany(mappedBy = "productCategory", cascade = CascadeType.ALL)
37:      private List<Product> productList;
38:
39:      public ProductCategory() {}
40:
41:      public ProductCategory(String name) {
42:          this.name = name;
43:      }
44:
45:      public Integer getId() {
46:          return id;
47:      }
48:
49:      public void setId(Integer id) {
50:          this.id = id;
51:      }
52:
53:      public String getName() {
54:          return name;
55:      }
56:
57:      public void setName(String name) {
58:          this.name = name;
59:      }
60:
61:      public List<Product> getProductList() {
62:          return productList;
63:      }
64:
65:      public void setProductList(List<Product> productList) {
66:          this.productList = productList;
```

```
1: package de.hsb.shop.model;
2:
3: public enum Creditcardtype {
4:
5:     MASTER ("Mastercard"), VISA ("Visa"), AMEX ("American Express");
6:
7:     private final String label;
8:     private Creditcardtype(String label) {
9:         this.label = label;
10:    }
11:    public String getLabel() {
12:        return label;
13:    }
14:
15: }
```

```
1: package de.hsb.shop.model;
2:
3: import java.io.Serializable;
4: import java.util.Date;
5: import java.util.List;
6:
7: import javax.persistence.CascadeType;
8: import javax.persistence.Column;
9: import javax.persistence.Entity;
10: import javax.persistence.FetchType;
11: import javax.persistence.GeneratedValue;
12: import javax.persistence.GenerationType;
13: import javax.persistence.Id;
14: import javax.persistence.JoinColumn;
15: import javax.persistence.ManyToOne;
16: import javax.persistence.NamedQuery;
17: import javax.persistence.OneToMany;
18: import javax.persistence.OneToOne;
19: import javax.persistence.Temporal;
20: import javax.persistence.TemporalType;
21:
22: import org.slf4j.Logger;
23: import org.slf4j.LoggerFactory;
24:
25: import de.hsb.shop.controller.SessionHandler;
26:
27: @NamedQuery(name = "SelectMember", query = "Select m from Member m")
28: @Entity
29: public class Member implements Serializable {
30:
31:     private static final long serialVersionUID = 1L;
32:
33:     @Id
34:     @GeneratedValue(strategy = GenerationType.AUTO)
35:     @Column(name = "id")
36:     private Integer id;
37:
38:     private String username;
39:
40:     private String email;
41:
42:     private String password;
43:
44:     private String familyname;
45:
46:     private String firstname;
47:
48:     private Boolean newsletter;
49:
50:     private String title;
51:
52:     @JoinColumn(name = "role", referencedColumnName = "id")
53:     @ManyToOne
54:     private Role role;
55:
56:     @OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
57:     @JoinColumn(name = "member_id")
58:     private List<Adress> adressList;
59:
60:     @OneToOne(cascade = CascadeType.ALL)
61:     private Creditcard creditcard;
62:
63:     @Temporal(TemporalType.DATE)
64:     private Date birthday;
65:
66:     public Member() {
67:     }
```

```
68:
69:     public Member(String username, String vorname, String nachname, String password, String email, Date geburtsdatum) {
70:         super();
71:         this.username = username;
72:         this.firstname = vorname;
73:         this.familyname = nachname;
74:         this.password = password;
75:         this.email = email;
76:         this.birthday = geburtsdatum;
77:     }
78:
79:     public Integer getId() {
80:         return id;
81:     }
82:
83:     public void setId(Integer id) {
84:         this.id = id;
85:     }
86:
87:     public String getUsername() {
88:         return username;
89:     }
90:
91:     public void setUsername(String username) {
92:         this.username = username;
93:     }
94:
95:     public String getEmail() {
96:         return email;
97:     }
98:
99:     public void setEmail(String email) {
100:         this.email = email;
101:     }
102:
103:     public String getFamilyname() {
104:         return familyname;
105:     }
106:
107:     public void setFamilyname(String familyname) {
108:         this.familyname = familyname;
109:     }
110:
111:     public String getFirstname() {
112:         return firstname;
113:     }
114:
115:     public void setFirstname(String firstname) {
116:         this.firstname = firstname;
117:     }
118:
119:     public Boolean getNewsletter() {
120:         return newsletter;
121:     }
122:
123:     public void setNewsletter(Boolean newsletter) {
124:         this.newsletter = newsletter;
125:     }
126:
127:     public String getTitle() {
128:         return title;
129:     }
130:
131:     public void setTitle(String title) {
132:         this.title = title;
133:     }
```

```
134:
135:     public Role getRole() {
136:         return role;
137:     }
138:
139:     public void setRole(Role role) {
140:         this.role = role;
141:     }
142:
143:     public List<Adress> getAdressList() {
144:         return adressList;
145:     }
146:
147:     public void setAdressList(List<Adress> adressList) {
148:         this.adressList = adressList;
149:     }
150:
151:     public Creditcard getCreditcard() {
152:         return creditcard;
153:     }
154:
155:     public void setCreditcard(Creditcard creditcard) {
156:         this.creditcard = creditcard;
157:     }
158:
159:     public Date getBirthday() {
160:         return birthday;
161:     }
162:
163:     public void setBirthday(Date birthday) {
164:         this.birthday = birthday;
165:     }
166:
167:     public String getPassword() {
168:         return password;
169:     }
170:
171:     public void setPassword(String password) {
172:         this.password = password;
173:     }
174:
175: }
```

```
1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6: package de.hsb.shop.model;
7:
8: import java.io.Serializable;
9: import java.util.List;
10:
11: import javax.persistence.CascadeType;
12: import javax.persistence.Column;
13: import javax.persistence.Entity;
14: import javax.persistence.GeneratedValue;
15: import javax.persistence.GenerationType;
16: import javax.persistence.Id;
17: import javax.persistence.NamedQuery;
18: import javax.persistence.OneToMany;
19: import javax.validation.constraints.Size;
20:
21: import org.slf4j.Logger;
22: import org.slf4j.LoggerFactory;
23:
24: import de.hsb.shop.controller.SessionHandler;
25:
26: @NamedQuery(name = "SelectRole", query = "Select r from Role r")
27: @Entity
28: public class Role implements Serializable{
29:
30:     private static Logger logger = LoggerFactory.getLogger(SessionHandler.class);
31:     private static final long serialVersionUID = 1L;
32:
33:     @Id
34:     @GeneratedValue(strategy = GenerationType.AUTO)
35:     @Column(name = "id")
36:     private Integer id;
37:
38:     @Size(max = 255)
39:     @Column(name = "label")
40:     private String label;
41:
42:     @OneToMany(mappedBy = "role", cascade = CascadeType.ALL)
43:     private List<Member> memberList;
44:
45:     public Role(){}
46:
47:     public Role(String label){
48:         this.label = label;
49:     }
50:
51:     public Integer getId() {
52:         return id;
53:     }
54:
55:     public void setId(Integer id) {
56:         this.id = id;
57:     }
58:
59:     public String getLabel() {
60:         return label;
61:     }
62:
63:     public void setLabel(String label) {
64:         this.label = label;
65:     }
66:
67:     public List<Member> getMemberList() {
```

```
68:         return memberList;
69:     }
70:
71:     public void setMemberList(List<Member> memberList) {
72:         this.memberList = memberList;
73:     }
74: }
```

```
1:  /*
2:   * To change this license header, choose License Headers in Project Properties.
3:   * To change this template file, choose Tools | Templates
4:   * and open the template in the editor.
5:   */
6:  package de.hsb.shop.model;
7:
8:  import java.io.Serializable;
9:
10: import javax.persistence.Column;
11: import javax.persistence.Entity;
12: import javax.persistence.GeneratedValue;
13: import javax.persistence.GenerationType;
14: import javax.persistence.Id;
15: import javax.persistence.JoinColumn;
16: import javax.persistence.ManyToOne;
17: import javax.persistence.NamedQueries;
18: import javax.persistence.NamedQuery;
19: import javax.validation.constraints.Size;
20:
21: @NamedQueries({
22:     @NamedQuery(name = "SelectProduct", query = "Select t from Product t"),
23:     @NamedQuery(name = "SelectProductByProductCategory", query = "Select t from Pr
duct t where t.productCategory.name = :productCategory")
24: })
25: @Entity
26: public class Product implements Serializable {
27:
28:     private static final long serialVersionUID = 1L;
29:
30:     @Id
31:     @GeneratedValue(strategy = GenerationType.AUTO)
32:     @Column(name = "id")
33:     private Integer id;
34:
35:     @Size(max = 255)
36:     @Column(name = "name")
37:     private String name;
38:
39:     @Size(max = 255)
40:     @Column(name = "description")
41:     private String description;
42:
43:     @Column(name = "price")
44:     private Float price;
45:
46:     @JoinColumn(name = "productCategory", referencedColumnName = "id")
47:     @ManyToOne
48:     private ProductCategory productCategory;
49:
50:     public Product() {
51:     }
52:
53:     public Product(String name) {
54:         this.name = name;
55:     }
56:
57:     public Integer getId() {
58:         return id;
59:     }
60:
61:     public void setId(Integer id) {
62:         this.id = id;
63:     }
64:
65:     public String getName() {
66:         return name;
```

```
67:     }
68:
69:     public void setName(String name) {
70:         this.name = name;
71:     }
72:
73:     public String getDescription() {
74:         return description;
75:     }
76:
77:     public void setDescription(String description) {
78:         this.description = description;
79:     }
80:
81:     public ProductCategory getProductCategory() {
82:         return productCategory;
83:     }
84:
85:     public void setProductCategory(ProductCategory productCategory) {
86:         this.productCategory = productCategory;
87:     }
88:
89:     public Float getPrice() {
90:         return price;
91:     }
92:
93:     public void setPrice(Float price) {
94:         this.price = price;
95:     }
96: }
```



```
1: package de.hsb.shop.model;
2:
3: import java.io.Serializable;
4: import java.util.Date;
5:
6: import javax.persistence.Entity;
7: import javax.persistence.GeneratedValue;
8: import javax.persistence.GenerationType;
9: import javax.persistence.Id;
10: import javax.persistence.NamedQuery;
11: import javax.persistence.Temporal;
12: import javax.persistence.TemporalType;
13:
14: @NamedQuery(name = "SelectCreditcard", query = "Select k from Creditcard k")
15: @Entity
16: public class Creditcard implements Serializable {
17:
18:     /**
19:      *
20:      */
21:     private static final long serialVersionUID = 1L;
22:
23:     @Id
24:     @GeneratedValue(strategy = GenerationType.AUTO)
25:     private Integer id;
26:     private String type;
27:     private String number;
28:     private String owner;
29:
30:     @Temporal(TemporalType.DATE)
31:     private Date validTo;
32:
33:     public Creditcard() {
34:     }
35:
36:     public Integer getId() {
37:         return id;
38:     }
39:
40:     public void setId(Integer id) {
41:         this.id = id;
42:     }
43:
44:     public Date getValidTo() {
45:         return validTo;
46:     }
47:
48:     public void setValidTo(Date validTo) {
49:         this.validTo = validTo;
50:     }
51:
52:     public String getType() {
53:         return type;
54:     }
55:
56:     public void setType(String type) {
57:         this.type = type;
58:     }
59:
60:     public String getNumber() {
61:         return number;
62:     }
63:
64:     public void setNumber(String number) {
65:         this.number = number;
66:     }
67: }
```

```
68:     public String getOwner() {
69:         return owner;
70:     }
71:
72:     public void setOwner(String owner) {
73:         this.owner = owner;
74:     }
75:
76: }
```

```
1: /*
2:  * To change this license header, choose License Headers in Project Properties.
3:  * To change this template file, choose Tools | Templates
4:  * and open the template in the editor.
5:  */
6: package de.hsb.shop.model;
7:
8: public enum Title {
9:     HERR("Herr"), FRAU("Frau"), FIRMA("Firma");
10:     private final String label;
11:
12:     private Title(String label) {
13:         this.label = label;
14:     }
15:
16:     public String getLabel() {
17:         return label;
18:     }
19: }
20:
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!DOCTYPE html>
3: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
4:                 xmlns:f="http://java.sun.com/jsf/core"
5:                 xmlns:h="http://java.sun.com/jsf/html"
6:                 xmlns:ui="http://java.sun.com/jsf/facelets"
7:                 xmlns:p="http://primefaces.org/ui">
8:     <h:form id="headform">
9:         <p:panel class="noCorner noBorder">
10:             <div class="inblock">
11:                 <p:menubar class="">
12:                     <p:menuitem action="#{sessionHandler.goToStartpage()}">
13:                         <ui:include src="images/mug.svg"/>
14:                     </p:menuitem>
15:                 </p:menubar>
16:             </div>
17:             <div class="inblock">
18:                 <p:menubar model="#{sessionHandler.model}" class="noCorner noBorde
r largeFont width100 noBackground" />
19:             </div>
20:             <div class="inblock fright">
21:                 <p:menubar class="noCorner noBackground noBorder" rendered="#{ses
sionHandler.logged}">
22:                     <f:facet name="options">
23:                         <ui:fragment rendered="#{sessionHandler.logged}">
24:                             <p:commandButton action="#{sessionHandler.logout()}" u
pdate="@form" immediate="true"
25:                                             value="#{msg.logout}"/>
26:                             <p:spacer></p:spacer>
27:                             <p:selectOneMenu id="language" class="fright" value="#
{sessionHandler.language}">
28:                                 <f:selectItem itemLabel="#{msg.german}" itemValue=
"de" />
29:                                 <f:selectItem itemLabel="#{msg.english}" itemValue
="en" />
30:                                 <p:ajax event="valueChange" update="@all" />
31:                             </p:selectOneMenu>
32:                         </ui:fragment>
33:                     </f:facet>
34:                 </p:menubar>
35:                 <p:menubar class="noCorner noBackground noBorder" rendered="#{!se
ssionHandler.logged}">
36:                     <f:facet name="options">
37:                         <ui:fragment rendered="#{!sessionHandler.logged}">
38:                             <p:commandButton action="login.xhtml?faces-redirect=tr
ue" immediate="true"
39:                                             value="#{msg.login}"/>
40:                             <p:commandButton action="register.xhtml?faces-redirect
=true" immediate="true"
41:                                             value="#{msg.register}"/>
42:                             <p:spacer></p:spacer>
43:                             <p:selectOneMenu id="language2" class="fright" value=
#{sessionHandler.language}">
44:                                 <f:selectItem itemLabel="#{msg.german}" itemValue=
"de" />
45:                                 <f:selectItem itemLabel="#{msg.english}" itemValue
="en" />
46:                                 <p:ajax event="valueChange" update="@all" />
47:                             </p:selectOneMenu>
48:                         </ui:fragment>
49:                     </f:facet>
50:                 </p:menubar>
51:             </div>
52:         </p:panel>
53:         <p:menubar class="noCorner profilePanel" rendered="#{sessionHandler.logge
d}">
54:             <p:menuitem value="#{msg.myOrders}" disabled="true"/>
```

```
55:             <p:menuitem value="#{msg.editProfile}" action="profil.xhtml?faces-redi
rect=true" immediate="true" />
56:             <p:menuitem value="#{msg.adminOptions}" disabled="true" rendered="#{se
ssionHandler.admin}"/>
57:         <f:facet name="options">
58:             <ui:fragment rendered="#{sessionHandler.logged}">
59:                 #{sessionHandler.shoppingCart.size()} #{msg.chosenArticle}
60:                 <p:commandButton update="@form" immediate="true" action="shopp
ingCart.xhtml?faces-redirect=true"
61:                                     id="logout" value="#{msg.toShippingCart}"/>
62:             </ui:fragment>
63:         </f:facet>
64:     </p:menubar>
65: </h:form>
66: </ui:composition>
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:   xmlns:f="http://java.sun.com/jsf/core"
10:   xmlns:h="http://java.sun.com/jsf/html"
11:   xmlns:ui="http://java.sun.com/jsf/facelets"
12:   xmlns:p="http://primefaces.org/ui"
13:   template="index.xhtml">
14:   <ui:define name="content">
15:     <h:form>
16:       <f:event listener="#{profileHandler.sessionhandler.checkLoggedIn}" type="preRenderView" />
17:       <div class="width100">
18:         <div class="MarAuto MaxWidth1400" >
19:           <p:panel id="head" header="#{msg.editProfile}" >
20:             <p:messages autoUpdate="true" showIcon="true" showDetail="true"/>
21:             <div class="EmptyBox20" />
22:             <p:panel id="data" header="#{msg.DATA}">
23:               <p:panelGrid columns="2" styleClass="ui-panelgrid-blank" layout="grid" >
24:                 <p:outputPanel>
25:                   <p:outputLabel value="#{msg.username}: " for="username"/><br/>
26:                   <p:inputText class="width100 border-box" id="username" value="#{profileHandler.member.username}" disabled="true" label="text"/>
27:                 </p:outputPanel>
28:                 <p:separator/></p:separator>
29:                 <p:outputPanel>
30:                   <p:outputLabel value="Anrede: " for="username"/><br/>
31:                   <p:selectOneMenu class="width100 border-box" value="#{profileHandler.member.title}">
32:                     <f:selectItems value="#{sessionHandler.titleValues}" var="a" itemLabel="#{a.label}"></f:selectItems>
33:                   </p:selectOneMenu>
34:                 </p:outputPanel>
35:                 <p:outputPanel>
36:                   <p:outputLabel value="#{msg.birthday}: " for="birthday"/><br/>
37:                   <p:calendar value="#{profileHandler.member.birthday}" class="width100 border-box" placeholder="#{msg.birthday}" id="birthday" mode="popup" navigator="true" showOn="button" mindate="01.01.1900" pagedate="01.01.1980" maxdate="01.01.2003" pattern="dd.MM.yyyy" />
38:                 </p:outputPanel>
39:                 <p:outputPanel>
40:                   <p:outputLabel value="#{msg.firstname}: " for="firstname"/><br/>
41:                   <p:inputText class="width100 border-box" id="firstname" value="#{profileHandler.member.firstname}" label="text" placeholder="#{msg.firstname}" />
42:                 </p:outputPanel>
43:                 <p:outputPanel>
44:                   <p:outputLabel value="#{msg.familyname}: " for="lastname"/><br/>
45:                   <p:inputText class="width100 border-box" id="lastname" value="#{profileHandler.member.familyname}" label="text" placeholder="#{msg.familyname}" />
46:                 </p:outputPanel>
47:                 <p:outputPanel>
48:                   <div class="TextAlignCenter">
49:                     <div class="EmptyBox10"/>
50:                     <div class="MarAuto width50">
51:                       <p:commandButton class="border-box" id="apply" action="#{profileHandler.update()}" update="data" value="#{msg.apply}" partialSubmit="true" process="data" />
52:                     </div>
53:                   </div>
54:                 </div>
55:               </p:panel>
56:               <div class="EmptyBox20" />
57:               <p:panel id="ad" header="#{msg.ADRESS}" >
58:                 <div class="EmptyBox10"/>
59:                 <ui:fragment rendered="#{!profileHandler.addAddress}">
60:                   <p:dataTable value="#{profileHandler.member.addressList}" var="adress" paginator="true" rows="4" paginatorAlwaysVisible="false" paginatorPosition="top" paginatorTempLate="{CurrentPageReport} {FirstPageLink} {PreviousPageLink} {PageLinks} {NextPageLink} {LastPageLink}">
61:                     <f:facet name="header">
62:                       </f:facet>
63:                     </p:column>
64:                     <f:facet name="header">
65:                       </f:facet>
66:                     <f:facet name="header">
67:                       </f:facet>
68:                     <f:facet name="header">
69:                       </f:facet>
70:                     <f:facet name="header">
71:                       </f:facet>
72:                     <f:facet name="header">
73:                       </f:facet>
74:                     <f:facet name="header">
75:                       </f:facet>
76:                     <f:facet name="header">
77:                       </f:facet>
78:                     <f:facet name="header">
79:                       </f:facet>
80:                     <f:facet name="header">
81:                       </f:facet>
82:                     <f:facet name="header">
83:                       </f:facet>
84:                     <f:facet name="header">
85:                       </f:facet>
86:                     <p:commandLink id="trash" value="#{msg.Delete}" action="#{profileHandler.deleteAddress(address)}" immediate="true" />
87:                     <p:spacer/><p:spacer/><p:spacer/>
88:                     <p:commandLink id="edit" value="#{msg.Edit}" action="#{profileHandler.editAddress(address)}" immediate="true"/>
89:                   </p:column>
90:                 </p:dataTable>
91:                 <div class="EmptyBox5" />
92:                 <div class="EmptyBox10"/>
93:                 <div class="TextAlignCenter">
94:                   <p:commandButton class="border-box" id="apply7" value="#{profileHandler.createAddress()}" value="#{msg.newAddress}" partialSubmit="true" process="ad" update="ad"/>
95:                 </div>
96:               </ui:fragment>
97:               <ui:fragment rendered="#{profileHandler.addAddress}">
98:                 <p:panelGrid columns="2" styleClass="ui-panelgrid-blank" layout="grid" >
99:                   <p:outputPanel>
100:                     <p:outputLabel value="#{msg.street}: " for="street"/><br/>
101:                     <p:inputText id="street" required="true" value="#{profileHandler.tmpAddress.street}" class="width100 border-box" label="text" place
```

```

105:         <p:message for="street"/>
106:     </p:outputPanel>
107:     <p:outputPanel>
108:         <p:outputLabel value="#{msg.number}: " for
="num"/><br/>
109:         <p:inputText id="num" value="#{profileHand
ler.tmpAdress.number}" class="width100 border-box" label="text" placeholder="#{msg.number
}"/>
110:         <p:message for="num"/>
111:     </p:outputPanel>
112:     <p:outputPanel>
113:         <p:outputLabel value="#{msg.zipCode}: " fo
r="zip"/><br/>
114:         <p:inputText id="zip" required="true" valu
e="#{profileHandler.tmpAdress.zipCode}" class="width100 border-box" label="text" placehol
der="#{msg.zipCode}" requiredMessage="#{msg.invalidInput}"/>
115:         <p:message for="zip"/>
116:     </p:outputPanel>
117:     <p:outputPanel>
118:         <p:outputLabel value="#{msg.city}: " for="
city"/><br/>
119:         <p:inputText id="city" required="true" val
ue="#{profileHandler.tmpAdress.city}" class="width100 border-box" label="text" placeholde
r="#{msg.city}" requiredMessage="#{msg.invalidInput}"/>
120:         <p:message for="city"/>
121:     </p:outputPanel>
122: </p:panelGrid>
123: <div class="EmptyBox5" />
124: <div class="EmptyBox10"/>
125: <div class="TextAlignCenter">
126:     <p:commandButton id="apply5" class="border-box
" action="#{profileHandler.saveAdress()}" value="#{msg.save}"
127:         partialSubmit="true" process
="ad" update="ad"/>
128:     <p:commandButton class="border-box" id="apply8
" immediate="true" action="#{profileHandler.setAddAdress(false)}" value="#{msg.cancel}"
129:         update="ad"/>
131: </div>
132: </ui:fragment>
133: </p:panel>
134: <div class="EmptyBox20" />
135: <h:panelGroup>
136:     <div class="EmptyBox5" />
137:     <p:panel id="pay" header="#{msg.PAYMENT}">
138:         <div class="EmptyBox10"/>
139:         <ui:fragment rendered="#{!profileHandler.addCredit
Card}">
140:             <ui:fragment rendered="#{profileHandler.member
.creditcard ne null}">
141:                 <p:outputLabel value="#{msg.creditCard}:"/
>
142:                 <p:panelGrid columns="2" styleClass="ui-p
anelgrid-blank" layout="grid" >
143:                     <p:outputPanel>
144:                         <p:outputLabel value="#{msg.ccType}
": " for="typeView"/><br/>
145:                         <p:inputText id="typeView" disable
d="true" value="#{profileHandler.member.creditcard.type}" class="width100 border-box" lab
el="text"/>
146:                     </p:outputPanel>
147:                     <p:outputPanel>
148:                         <p:outputLabel value="#{msg.ccNumb
er}: " for="numberView"/><br/>
149:                         <p:inputText id="numberView" disab
led="true" value="#{profileHandler.member.creditcard.number}" class="width100 border-box"
label="text"/>
150:                     </p:outputPanel>
151:                     <p:outputPanel>
152:                         <p:outputLabel value="#{msg.ccVali
dUntil}: " for="untilView"/><br/>
153:                         <p:calendar id="untilView"
value="#{profileHandle
r.member.creditcard.validTo}" mode="popup"
154:                             navigator="true" showO
n="button" pattern="MM/yy" disabled="true"/>
155:                     </p:outputPanel>
156:                     <p:outputPanel>
157:                         <p:outputLabel value="#{msg.ccOwne
r}: " for="ownerView"/><br/>
158:                         <p:inputText id="ownerView" disabl
ed="true" value="#{profileHandler.member.creditcard.owner}" class="width100 border-box" l
abel="text"/>
159:                     </p:outputPanel>
160:                     </p:panelGrid>
161:                     <div class="EmptyBox5" />
162:                     <div class="EmptyBox10"/>
163:                     <div class="TextAlignCenter">
164:                         <p:commandButton class="border-box" id
="apply99" action="#{profileHandler.editCreditCard(profileHandler.member.creditcard)}"
165:                             value="#{msg.Edit}" p
artialSubmit="true" process="pay" update="pay" />
166:                     </div>
167:                     </ui:fragment>
168:                     <div class="TextAlignCenter">
169:                         <p:commandButton class="border-box" id="ap
ply6" action="#{profileHandler.createCreditCard()}" value="#{msg.newCreditCard}"
170:                             partialSubmit="true" pro
cess="pay" update="pay" rendered="#{profileHandler.member.creditcard eq null}"/>
171:                     </div>
172:                     </ui:fragment>
173:                     <ui:fragment rendered="#{profileHandler.addCreditC
ard}">
174:                         <p:outputLabel value="#{msg.creditCard}:"/>
175:                         <p:panelGrid columns="2" styleClass="ui-panelg
rid-blank" layout="grid" >
176:                             <p:outputPanel>
177:                                 <p:outputLabel value="#{msg.ccType}: "
178:                                     for="typ"/><br/>
179:                                 <p:selectOneMenu id="typ"
value="#{profileHandl
er.tmpCreditcard.type}">
180:                                     <f:selectItems var="form"
value="#{profileHan
dler.creditcardtypeValues}"
181:                                         itemValue="#{form.1
label}" itemLabel="#{form.label}" />
182:                                 </p:selectOneMenu>
183:                                 </p:outputPanel>
184:                                 <p:outputPanel>
185:                                     <p:outputLabel value="#{msg.ccNumber}:
" for="numb"/><br/>
186:                                     <p:inputText id="numb" required="true"
value="#{profileHandler.tmpCreditcard.number}" class="width100 border-box" label="text"
placeholder="#{msg.ccNumber}" requiredMessage="#{msg.invalidInput}" >
187:                                         <f:converter converterId="creditca
rdConverter" />
188:                                     <ui:remove>
189:                                         <f:validator validatorId="cred
itcardValidator" />
190:                                     </ui:remove>
191:                                     </p:inputText>
192:                                     <p:message for="numb"/>
193:                                 </p:outputPanel>
194:                             </p:panelGrid>
195:                         </ui:fragment>
196:                     </div>
197:                 </div>
198:             </ui:fragment>
199:         </p:panel>
200:     </h:panelGroup>
201: </div>
202: </p:panel>
203: </p:panelGrid>
204: </div>
205: </ui:fragment>
206: </p:panel>
207: </p:panelGrid>
208: </div>
209: </ui:fragment>
210: </p:panel>
211: </p:panelGrid>
212: </div>
213: </ui:fragment>
214: </p:panel>
215: </p:panelGrid>
216: </div>
217: </ui:fragment>
218: </p:panel>
219: </p:panelGrid>
220: </div>
221: </ui:fragment>
222: </p:panel>
223: </p:panelGrid>
224: </div>
225: </ui:fragment>
226: </p:panel>
227: </p:panelGrid>
228: </div>
229: </ui:fragment>
230: </p:panel>
231: </p:panelGrid>
232: </div>
233: </ui:fragment>
234: </p:panel>
235: </p:panelGrid>
236: </div>
237: </ui:fragment>
238: </p:panel>
239: </p:panelGrid>
240: </div>
241: </ui:fragment>
242: </p:panel>
243: </p:panelGrid>
244: </div>
245: </ui:fragment>
246: </p:panel>
247: </p:panelGrid>
248: </div>
249: </ui:fragment>
250: </p:panel>
251: </p:panelGrid>
252: </div>
253: </ui:fragment>
254: </p:panel>
255: </p:panelGrid>
256: </div>
257: </ui:fragment>
258: </p:panel>
259: </p:panelGrid>
260: </div>
261: </ui:fragment>
262: </p:panel>
263: </p:panelGrid>
264: </div>
265: </ui:fragment>
266: </p:panel>
267: </p:panelGrid>
268: </div>
269: </ui:fragment>
270: </p:panel>
271: </p:panelGrid>
272: </div>
273: </ui:fragment>
274: </p:panel>
275: </p:panelGrid>
276: </div>
277: </ui:fragment>
278: </p:panel>
279: </p:panelGrid>
280: </div>
281: </ui:fragment>
282: </p:panel>
283: </p:panelGrid>
284: </div>
285: </ui:fragment>
286: </p:panel>
287: </p:panelGrid>
288: </div>
289: </ui:fragment>
290: </p:panel>
291: </p:panelGrid>
292: </div>
293: </ui:fragment>
294: </p:panel>
295: </p:panelGrid>
296: </div>
297: </ui:fragment>
298: </p:panel>
299: </p:panelGrid>
300: </div>
301: </ui:fragment>
302: </p:panel>
303: </p:panelGrid>
304: </div>
305: </ui:fragment>
306: </p:panel>
307: </p:panelGrid>
308: </div>
309: </ui:fragment>
310: </p:panel>
311: </p:panelGrid>
312: </div>
313: </ui:fragment>
314: </p:panel>
315: </p:panelGrid>
316: </div>
317: </ui:fragment>
318: </p:panel>
319: </p:panelGrid>
320: </div>
321: </ui:fragment>
322: </p:panel>
323: </p:panelGrid>
324: </div>
325: </ui:fragment>
326: </p:panel>
327: </p:panelGrid>
328: </div>
329: </ui:fragment>
330: </p:panel>
331: </p:panelGrid>
332: </div>
333: </ui:fragment>
334: </p:panel>
335: </p:panelGrid>
336: </div>
337: </ui:fragment>
338: </p:panel>
339: </p:panelGrid>
340: </div>
341: </ui:fragment>
342: </p:panel>
343: </p:panelGrid>
344: </div>
345: &lt
```

```
195:                </p:outputPanel>
196:                <p:outputPanel>
197:                    <p:outputLabel value="#{msg.ccValidUnt
il}: ({msg.example} 03/17)" for="dateUntil"/><br/>
198:                <p:calendar id="dateUntil"
199:                    value="#{profileHandler.tm
pCreditcard.validTo}" mode="popup"
200:                    navigator="true" showOn="b
utton" pattern="dd/MM/yy" />
201:                <p:message for="dateUntil"/>
202:                </p:outputPanel>
203:                <p:outputPanel>
204:                    <p:outputLabel value="#{msg.ccOwner}"
for="owner"/><br/>
205:                <p:inputText id="owner" required="true
" value="#{profileHandler.tmpCreditcard.owner}" class="width100 border-box" label="text"
placeholder="#{msg.ccOwner}" requiredMessage="#{msg.invalidInput}" />
206:                <p:message for="owner"/>
207:                </p:outputPanel>
208:                <p:panelGrid>
209:                    <div class="EmptyBox5" />
210:                    <div class="EmptyBox10"/>
211:                    <div class="TextAlignCenter">
212:                        <p:commandButton id="apply9" class="border
-box" action="#{profileHandler.saveCreditcard()}" value="#{msg.save}"
213:                        partialSubmit="true" pro
cess="pay" update="pay"/>
214:                        <p:commandButton class="border-box" id="ap
ply10" immediate="true" action="#{profileHandler.setAddCreditCard(false)}" value="#{msg.c
ancel}"
215:                        update="pay"/>
216:                    </div>
217:                </ui:fragment>
218:            </p:panel>
219:        </p:panel>
220:        <div class="EmptyBox20" />
221:        <h:panelGroup>
222:            <div class="EmptyBox5"/>
223:            <p:panel id="pwpanel" header="#{msg.changePassword
}">
224:                <p:panelGrid columns="2" styleClass="ui-panelG
rid-blank" layout="grid" >
225:                    <p:outputPanel>
226:                        <p:outputLabel value="#{msg.newPasswor
t}: " for="pw"/><br/>
227:                    <p:password value="#{profileHandler.me
mber.password}" match="pw2" class="width100 border-box" placeholder="#{msg.password}" id
="pw"
228:                        validatorMessage="#{msg.pa
sswordNeedfourLetters}" requiredMessage="#{msg.passwordNeedfourLetters}">
229:                        <f:validateLength minimum="4" />
230:                    </p:password>
231:                    </p:outputPanel>
232:                    <p:outputPanel>
233:                        <br/><p:password class="width100 borde
r-box" placeholder="#{msg.repeatPassword}" id="pw2" requiredMessage="#{msg.passwordsNotM
atch}" />
234:                    </p:outputPanel>
235:                    <p:panelGrid>
236:                        <div class="TextAlignCenter">
237:                            <div class="EmptyBox10"/>
238:                            <div class="MarAuto width50">
239:                                <p:commandButton class="border-box" id
="apply2" action="#{profileHandler.update()}" value="#{msg.apply}"
240:                                partialSubmit="true"
process="pwpanel" update="pwpanel"/>
241:                            </div>
242:                        </div>
243:                    </p:panel>
244:                </div>
245:            </h:panelGroup>
246:            <div class="EmptyBox20" />
247:            <p:panel id="emailpan" header="#{msg.changeEmail}"
>
248:                <p:panelGrid columns="2" styleClass="ui-panelG
rid-blank" layout="grid" >
249:                    <p:outputPanel>
250:                        <p:outputLabel value="#{msg.newEmail}":
" for="email"/><br/>
251:                    <p:inputText id="email" value="#{profi
leHandler.member.email}" class="width100 border-box" placeholder="#{msg.email}" validato
rMessage="#{msg.emailInvalid}" requiredMessage="#{msg.emailInvalid}">
252:                        <f:validateRegex pattern="^[_A-Za-
z0-9-\\+](\\.[_A-Za-z0-9-\\+])*@[A-Za-z0-9-](\\.[A-Za-z0-9-]{2,})$" />
253:                    </p:inputText>
254:                    </p:outputPanel>
255:                    <p:outputPanel>
256:                        <p:panelGrid>
257:                            <div class="TextAlignCenter">
258:                                <div class="EmptyBox10"/>
259:                                <div class="MarAuto width50">
260:                                    <p:commandButton class="border-box" id
="apply3" action="#{profileHandler.update()}" value="#{msg.apply}"
261:                                    partialSubmit="true"
process="emailpan" />
262:                            </div>
263:                        </div>
264:                    </p:panel>
265:                </h:panelGroup>
266:                <div class="EmptyBox20" />
267:                <p:panel id="sonstiges" header="#{msg.MISC}">
268:                    <div class="TextAlignCenter">
269:                        <div class="EmptyBox10"/>
270:                        <p:outputLabel id="lab1" value="#{msg.newslett
erSubscribe}" />
271:                        <p:selectBooleanCheckbox label="lab1" value="#{
profileHandler.member.newsletter}" />
272:                    </div>
273:                    <div class="EmptyBox10"/>
274:                    <div class="MarAuto width50">
275:                        <p:commandButton class="border-box" id="ap
ply4" action="#{profileHandler.update()}" value="#{msg.apply}"
276:                        partialSubmit="true" pro
cess="sonstiges" />
277:                    </div>
278:                </div>
279:            </h:panelGroup>
280:            <div class="TextAlignCenter">
281:                <div class="EmptyBox30" />
282:                <p:commandButton class="border-box" value="#{msg.BackT
oStartpage}" action="#{sessionHandler.goToStartpage()}" immediate="true"/>
283:            </div>
284:        </p:panel>
285:    </div>
286:    </div>
287:    </div>
288:    </h:form>
289:    </ui:define>
290:    </ui:composition>
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:   xmlns:f="http://java.sun.com/jsf/core"
10:   xmlns:h="http://java.sun.com/jsf/html"
11:   xmlns:ui="http://java.sun.com/jsf/facelets"
12:   xmlns:p="http://primefaces.org/ui"
13:   template="index.xhtml">
14:   <ui:define name="content">
15:     <h:form>
16:       <f:event listener="#{sessionHandler.checkLoggedIn}" type="preRenderView" />
17:       <div class="width100">
18:         <div class="MarAuto MaxWidth1400" >
19:           <p:panel id="data" header="#{msg.shoppingCartTitle}" rendered="#{!shoppingCartController.finished}">
20:             <div class="EmptyBox30" />
21:             <ui:fragment rendered="#{shoppingCartController.shList.size() ne 0}">
22:               <p:panelGrid columns="1" styleClass="ui-panelgrid-blank" layout="grid" >
23:                 <p:dataTable value="#{shoppingCartController.shList}" var="product" paginator="true" rows="500" paginatorAlwaysVisible="false"
24:                   paginatorTemplate="{CurrentPageReport} {FirstPageLink} {PreviousPageLink} {PageLinks} {NextPageLink} {LastPageLink}"
25:                   >
26:                   <f:facet name="header" >
27:                     #{msg.article}
28:                   </f:facet>
29:                   <p:column>
30:                     <f:facet name="header" >
31:                       #{msg.product}
32:                     </f:facet>
33:                     #{product.product.name}
34:                   </p:column>
35:                   <p:column>
36:                     <f:facet name="header" >
37:                       #{msg.Amount}
38:                     </f:facet>
39:                     #{product.count}
40:                   </p:column>
41:                   <p:column>
42:                     <f:facet name="header" >
43:                       #{msg.singleCosts}
44:                     </f:facet>
45:                     <b>EUR<p:spacer width="10"/>#{product.product.price}</b>
46:                   </p:column>
47:                   <p:column>
48:                     <f:facet name="header" >
49:                       #{msg.wholeCosts}
50:                     </f:facet>
51:                     <b>EUR<p:spacer width="10"/>#{product.wholePrice}</b>
52:                   </p:column>
53:                 </p:dataTable>
54:               </p:panelGrid>
55:               <div class="EmptyBox20" />
56:               <p:panelGrid columns="2" styleClass="ui-panelgrid-blank" layout="grid" >
57:                 <p:panel header="#{msg.payingWith}">
58:                   <p:panelGrid columns="2" layout="grid" styleCl
```

```
ass="ui-panelgrid-blank" rendered="#{sessionHandler.member.creditcard ne null}">
59:                 <p:outputPanel>
60:                   <p:outputLabel value="#{msg.ccType}:"
61:                     for="typeView"/><br/>
62:                   <p:inputText id="typeView" disabled="true" value="#{profileHandler.member.creditcard.type}" class="width100 border-box" label="text"/>
63:                 </p:outputPanel>
64:                 <p:outputPanel>
65:                   <p:outputLabel value="#{msg.ccNumber}:"
66:                     for="numberView"/><br/>
67:                   <p:inputText id="numberView" disabled="true" value="#{profileHandler.member.creditcard.number}" class="width100 border-box" label="text"/>
68:                 </p:outputPanel>
69:                 <p:outputPanel>
70:                   <p:outputLabel value="#{msg.ccValidUntil}:"
71:                     for="untilView"/><br/>
72:                   <p:inputText id="untilView" disabled="true" value="#{profileHandler.member.creditcard.validTo}" class="width100 border-box" label="text"/>
73:                 </p:outputPanel>
74:                 <p:outputPanel>
75:                   <p:outputLabel value="#{msg.ccOwner}:"
76:                     for="ownerView"/><br/>
77:                   <p:inputText id="ownerView" disabled="true" value="#{profileHandler.member.creditcard.owner}" class="width100 border-box" label="text"/>
78:                 </p:outputPanel>
79:               </p:panelGrid>
80:             <ui:fragment rendered="#{sessionHandler.member.creditcard eq null}">
81:               <div class="EmptyBox30" />
82:               <div class="TextAlignCenter">
83:                 #{msg.needValidCreditCard}<br/>
84:                 <div class="EmptyBox10"/>
85:                 <p:commandButton class="border-box" value="#{msg.editProfile}" action="profil.xhtml?faces-redirect=true" immediate="true"/>
86:                 <div class="EmptyBox10"/>
87:               </div>
88:             </ui:fragment>
89:           </p:panel>
90:           <p:panel header="#{msg.deliveryTo}">
91:             <p:panelGrid columns="2" layout="grid" styleClass="ui-panelgrid-blank" rendered="#{sessionHandler.member.adressList.size() ne 0}">
92:               <p:outputPanel>
93:                 <p:outputLabel value="#{msg.street}:"
94:                   for="typeView"/><br/>
95:                 <p:inputText disabled="true" value="#{profileHandler.member.adressList.get(0).street}" class="width100 border-box" label="text"/>
96:               </p:outputPanel>
97:               <p:outputPanel>
98:                 <p:outputLabel value="#{msg.number}:"
99:                   for="numberView"/><br/>
100:                 <p:inputText disabled="true" value="#{profileHandler.member.adressList.get(0).city}" class="width100 border-box" label="text"/>
101:               </p:outputPanel>
102:               <p:outputPanel>
103:                 <p:outputLabel value="#{msg.street}:"
104:                   for="untilView"/><br/>
105:                 <p:inputText disabled="true" value="#{profileHandler.member.adressList.get(0).city}" class="width100 border-box" label="text"/>
106:               </p:outputPanel>
107:             </p:panelGrid>
108:           </p:panel>
109:           <p:panel header="#{msg.payingWith}">
110:             <p:panelGrid columns="2" layout="grid" styleCl
```



```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:                 xmlns:f="http://java.sun.com/jsf/core"
10:                 xmlns:h="http://java.sun.com/jsf/html"
11:                 xmlns:ui="http://java.sun.com/jsf/facelets"
12:                 xmlns:p="http://primefaces.org/ui"
13:                 template="index.xhtml">
14:     <ui:define name="content">
15:         <h:form id="productform">
16:             <p:growl id="growl" showDetail="true" sticky="true" />
17:             <div class="width100">
18:                 <div class="MarAuto MaxWidth1400" >
19:                     <p:panel id="data" header="{msg.category}" >> #{sessionHandler
.productHead} " >
20:                         <p:dataTable value="{#{sessionHandler.productList}" var="pr
oduct" paginator="true" rows="6" paginatorAlwaysVisible="false"
21:                                     paginatorTemplate="{CurrentPageReport} {First
PageLink} {PreviousPageLink} {PageLinks} {NextPageLink} {LastPageLink}">
22:                             <p:column class="fixedWidth110">
23:                                 <ui:include src="images/singleMug.svg"/>
24:                             </p:column>
25:                             <p:column>
26:                                 #{product.name}<br/>
27:                                 <p:separator/>
28:                                 #{product.description}
29:                                 <p:separator/>
30:                                 <div class="alignRight">
31:                                     <b>#{product.price}&\202\ / #{msg.immediatlyDe
liveryAble}</b>
32:                                     <p:spacer width="10px" />
33:                                     <p:commandButton class="border-box" action="{#{
sessionHandler.putProductIntoShoppingCart(product)}" value="{msg.addToShoppingCart}"
34:                                                         disabled="{#{!sessionHandler.l
ogged}" update="headform productform:growl"/>
35:                                 </div>
36:                             </p:column>
37:                         </p:dataTable>
38:                     </p:panel>
39:                 </div>
40:             </div>
41:         </h:form>
42:     </ui:define>
43: </ui:composition>
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:                 xmlns:f="http://java.sun.com/jsf/core"
10:                 xmlns:h="http://java.sun.com/jsf/html"
11:                 xmlns:ui="http://java.sun.com/jsf/facelets"
12:                 xmlns:p="http://primefaces.org/ui"
13:                 template="index.xhtml">
14:     <ui:define name="content">
15:         <h:form>
16:             <div class="width100">
17:                 <div class="MarAuto MaxWidth1400" >
18:                     <p:panelGrid columns="1" styleClass="ui-panelgrid-blank" layout
t="grid" columnClasses="ui-grid-col-12">
19:                         <h:panelGroup>
20:                             <p:panel id="data" header="{msg.welcome}" style="heig
ht: 400px">
21:                                 #{msg.welcomeToShop}

22:                             </p:panel>
23:                         </h:panelGroup>
24:                     </p:panelGrid>
25:                 </div>
26:             </div>
27:         </h:form>
28:     </ui:define>
29: </ui:composition>
```

```

1: <?xml version="1.0" encoding="UTF-8"?>
2: <!DOCTYPE html>
3: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
4:                 xmlns:f="http://java.sun.com/jsf/core"
5:                 xmlns:h="http://java.sun.com/jsf/html"
6:                 xmlns:ui="http://java.sun.com/jsf/facelets"
7:                 xmlns:p="http://primefaces.org/ui">
8:     <h:form>
9:         <p:panel class="noCorner noBorder" >
10:             <div class="EmptyBox10" />
11:             AGB Datenschutzerklärung Impressum 2016-2017, SWE3-Webshop
12:             <div class="EmptyBox40" />
13:         </p:panel>
14:     </h:form>
15: </ui:composition>

```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:   xmlns:f="http://java.sun.com/jsf/core"
10:   xmlns:h="http://java.sun.com/jsf/html"
11:   xmlns:ui="http://java.sun.com/jsf/facelets"
12:   xmlns:p="http://primefaces.org/ui"
13:   template="index.xhtml">
14:   <ui:define name="content">
15:     <h:form>
16:       <div class="width100">
17:         <div class="MarAuto fixedWidth500" >
18:           <p:panel id="panel" header="#{msg.registerTitle}">
19:             <div class="MarAuto fixedWidth350">
20:               <div class="TextAlignCenter">
21:
22:                 <div class="EmptyBox20"/>
23:
24:                 <h:panelGroup >
25:                   <p:inputText required="true" value="#{register
Handler.member.username}" class="width100 border-box"
26:                     requiredMessage="#{msg.userNeedFo
urLetters}" id="uname" placeholder="#{msg.username}*">
27:                     <f:validator binding="#{usernameValidator}
" />
28:                   </p:inputText>
29:                   <p:message for="uname"/>
30:                   <div class="EmptyBox10"/>
31:                   <p:outputPanel>
32:                     <p:selectOneMenu class="width100 border-bo
x" value="#{registerHandler.member.title}">
33:                       <f:selectItems value="#{sessionHandler
.titleValues}" var="a" itemLabel="#{a.label}"></f:selectItems>
34:                     </p:selectOneMenu>
35:                   </p:outputPanel>
36:                   <div class="EmptyBox10"/>
37:                   <p:inputText value="#{registerHandler.member.f
irstname}" class="width50 border-box" id="vname" placeholder="#{msg.firstname}">
38:                   </p:inputText>
39:                   <p:inputText value="#{registerHandler.member.f
amilyname}" class="width50 border-box" id="nname" placeholder="#{msg.familyname}">
40:                   </p:inputText>
41:                   <div class="EmptyBox10"/>
42:                   <p:calendar value="#{registerHandler.member.bi
rthday}" class="width100 border-box" placeholder="#{msg.birthday}" id="gb"
43:                     mode="popup" navigator="true" show
On="button" mindate="01.01.1900" pagedate="01.01.1980" maxdate="01.01.2003" pattern="dd.M
M.yyyy" />
44:                   <div class="EmptyBox10"/>
45:                   <p:inputText required="true" id="email" value=
"#{registerHandler.member.email}" class="width100 border-box" placeholder="#{msg.email}*"
validatorMessage="#{msg.emailInvalid}" requiredMessage="#{msg.emailInvalid}">
46:                     <f:validateRegex pattern="#^[_A-Za-z0-9-\+
+(\.[_A-Za-z0-9-]+)*@[A-Za-z0-9-]+(\.[A-Za-z0-9-]+)*\.([A-Za-z]{2,})$" />
47:                   </p:inputText>
48:                   <p:message for="email"/>
49:                   <div class="EmptyBox10"/>
50:                   <p:password required="true" value="#{registerH
andler.member.password}" match="pw2" class="width50 border-box" placeholder="#{msg.passwo
rd}*" id="pw"
51:                     validatorMessage="#{msg.passwordNe
edfourLetters}" requiredMessage="#{msg.passwordNeedfourLetters}">
```

```
52:                     <f:validateLength minimum="4" />
53:                   </p:password>
54:                   <p:password required="true" class="width50 bor
der-box" placeholder="#{msg.repeat}*" id="pw2" requiredMessage="#{msg.passwordsNotMatch}"
/>
55:                   <p:message for="pw"/>
56:                   <p:message for="pw2"/>
57:                 </h:panelGroup>
58:                 <div class="EmptyBox20"/>
59:                 <p:outputLabel id="lab1" value="#{msg.newsletterSu
bscribe}" />
60:                 <p:selectBooleanCheckbox label="lab1" value="#{reg
isterHandler.member.newsletter}"/>
61:                 <div class="EmptyBox20"/>
62:                 <div class="EmptyBox20"/>
63:                 <div class="MarAuto width50">
64:                   <p:commandButton class="width100 border-box" a
ction="#{registerHandler.saveAndLogin()}" update="panel" id="konto" value="#{msg.createAc
count}"/>
65:                 </div>
66:                 <div class="EmptyBox20"/>
67:                 <p:separator/>
68:                 <div class="EmptyBox20"/>
69:                 <p:commandLink value="#{msg.here}" action="login.x
html?faces-redirect=true" immediate="true" id="register"/>
70:                 <div class="EmptyBox20"/>
71:                 <p:commandLink value="#{msg.toLogin}"
72:                 </div>
73:                 </div>
74:                 </p:panel>
75:               </div>
76:             </div>
77:           </h:form>
78:         </ui:define>
79: </ui:composition>
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <html xmlns="http://www.w3.org/1999/xhtml"
9:       xmlns:f="http://java.sun.com/jsf/core"
10:      xmlns:h="http://java.sun.com/jsf/html"
11:      xmlns:ui="http://java.sun.com/jsf/facelets"
12:      xmlns:p="http://primefaces.org/ui">
13:   <f:view locale="#{sessionHandler.language}"/>
14:   <h:head>
15:     <title>#{msg.title}</title>
16:     <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
17:     <link rel="stylesheet" type="text/css" href="styles/shop.css"/>
18:   </h:head>
19:   <h:body class="noMargin outerBackground">
20:     <div class="innerBackground">
21:       <div id="top">
22:         <ui:include src="head.xhtml" />
23:         <div class="EmptyBox20"/>
24:       </div>
25:       <div id="content">
26:         <p:panel class="fright marright20" rendered="#{sessionHandler.logged}">
27:           #{msg.welcome}, #{sessionHandler.username}
28:         </p:panel>
29:         <div class="EmptyBox20"/>
30:         <ui:insert name="content" />
31:         <div class="EmptyBox50"/>
32:       </div>
33:       <div id="footer">
34:         <ui:include src="footer.xhtml" />
35:       </div>
36:     </div>
37:   </h:body>
38: </html>
```

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <!--
3: To change this license header, choose License Headers in Project Properties.
4: To change this template file, choose Tools | Templates
5: and open the template in the editor.
6: -->
7: <!DOCTYPE html>
8: <ui:composition xmlns="http://www.w3.org/1999/xhtml"
9:                 xmlns:f="http://java.sun.com/jsf/core"
10:                 xmlns:h="http://java.sun.com/jsf/html"
11:                 xmlns:ui="http://java.sun.com/jsf/facelets"
12:                 xmlns:p="http://primefaces.org/ui"
13:                 template="index.xhtml">
14:     <ui:define name="content">
15:         <h:form id="Login">
16:             <div class="MarAuto fixedWidth500">
17:                 <p:panel header="#{msg.loginTitle}">
18:                     <div class="MarAuto fixedWidth350">
19:                         <div class="TextAlignCenter">
20:                             <div class="EmptyBox20"/>
21:                             <ui:fragment rendered="#{!sessionHandler.logged}">
22:                                 <h:panelGroup id="panel">
23:                                     <p:inputText id="Name" class="width100 border-
box" required="true" placeholder="#{msg.name}"
24:                                     value="#{sessionHandler.username}"
" requiredMessage="#{msg.usernameInvalid}" />
25:                                     <p:message for="Name"/>
26:                                     <div class="EmptyBox10"/>
27:                                     <p:password id="Password" class="width100 bord
er-box" required="true" placeholder="#{msg.password}"
28:                                     value="#{sessionHandler.password}"
requiredMessage="#{msg.passwordInvalid}" />
29:                                     <p:outputLabel/>
30:                                     <p:message for="Password"/>
31:                                     <div class="EmptyBox10"/>
32:                                     <p:commandLink value="#{msg.forgetPassword}"/>
33:                                 </h:panelGroup>
34:                                 <div class="EmptyBox20"/>
35:                                 <div class="MarAuto width50">
36:                                     <p:commandButton class="width100 border-box" a
ction="#{sessionHandler.login()}" update="panel" id="Login" value="#{msg.login}"/>
37:                                 </div>
38:                                 <div class="EmptyBox20"/>
39:                                 <p:separator/>
40:                                 #{msg.noAccountClick}
41:                                 <p:commandLink value="#{msg.here}" action="registe
r.xhtml?faces-redirect=true" immediate="true"
42:                                 id="register"/>
43:                                 #{msg.toRegister}
44:                             </ui:fragment>
45:                             <ui:fragment rendered="#{sessionHandler.logged}">
46:                                 #{msg.alreadyLoggedIn}
47:                                 <div class="EmptyBox20"/>
48:                                 <div class="MarAuto width50">
49:                                     <p:commandButton class="width100 border-box" v
alue="#{msg.back}" action="#{sessionHandler.goToStartpage()}" update="panel" />
50:                                 </div>
51:                                 <div class="EmptyBox20"/>
52:                             </ui:fragment>
53:                         </div>
54:                     </div>
55:                 </p:panel>
56:             </div>
57:         </h:form>
58:     </ui:define>
59: </ui:composition>
```

```
1: .alignRight{
2:     text-align: right;
3: }
4:
5: .fixedWidth110{
6:     width: 110px;
7: }
8:
9: .fixedWidth500{
10:     width: 500px !important;
11: }
12:
13: .fixedWidth350{
14:     width: 350px !important;
15: }
16:
17: .marright20{
18:     margin-right: 20px;
19: }
20:
21: .fright{
22:     float: right;
23: }
24:
25: .inblock{
26:     display: inline-block;
27: }
28:
29: .largeFont{
30:     font-size: larger !important;
31: }
32:
33: .ui-panelgrid.noPadding .ui-grid-responsive .ui-panelgrid-cell {
34:     border: 0 none;
35:     padding: 0px;
36: }
37:
38: .container{
39:     min-height: 70%;
40:     min-height: -webkit-calc(100% - 186px);
41:     min-height: -moz-calc(100% - 186px);
42:     min-height: calc(100% - 186px);
43: }
44:
45: .noCorner {
46:     -moz-border-radius: 0px;
47:     -webkit-border-radius: 0px;
48:     border-radius: 0px;
49: }
50:
51: .noMargin{
52:     margin: 0;
53: }
54:
55: .Container50{
56:     width:50%;
57:     float:left;
58: }
59:
60: .border-box{
61:     box-sizing: border-box !important;
62: }
63:
64: .width100{
65:     width: 100% !important;
66: }
67:
68: .width90{
69:     width: 90% !important;
70: }
71:
72: .width50{
```

```
73:     width: 50% !important;
74: }
75:
76: .MarAuto{
77:     margin: auto;
78: }
79:
80: .height40{
81:     height: 40px;
82: }
83:
84: .height15{
85:     height: 15px;
86: }
87:
88: .height10{
89:     height: 10px;
90: }
91:
92: .height5{
93:     height: 5px;
94: }
95:
96: .floatRight{
97:     float: right;
98: }
99:
100: .ui-widget {
101:     font-family: Verdana, Arial, sans-serif;
102:     font-size: 1em;
103: }
104:
105: .MaxWidthProfil{
106:     max-width: 400px;
107: }
108:
109: .MaxWidth1400{
110:     max-width: 1200px;
111: }
112:
113: .MaxWidth800{
114:     max-width: 800px;
115: }
116:
117: .Card{
118:     padding: 20px;
119:     display: block;
120:     background-color: #ffffff;
121:     border-radius: 2px;
122:     -webkit-border-radius: 2px;
123:     -moz-border-radius: 2px;
124:     font-family: 'robotoregular';
125:     color: #546e7a;
126: }
127:
128: .EmptyBox5{ display:block; width:100%; height:5px; overflow:hidden;}
129:
130: .EmptyBox10{ display:block; width:100%; height:10px; overflow:hidden;}
131:
132: .EmptyBox20{ display:block; width:100%; height:20px; overflow:hidden;}
133:
134: .EmptyBox30{ display:block; width:100%; height:30px; overflow:hidden;}
135:
136: .EmptyBox40{ display:block; width:100%; height:40px; overflow:hidden;}
137:
138: .EmptyBox50{ display:block; width:100%; height:50px; overflow:hidden;}
139:
140: .EmptyBox60{ display:block; width:100%; height:60px; overflow:hidden;}
141:
142: .EmptyBox70{ display:block; width:100%; height:70px; overflow:hidden;}
143:
144: .EmptyBox80{ display:block; width:100%; height:80px; overflow:hidden;}
145:
146: .EmptyBox90{ display:block; width:100%; height:90px; overflow:hidden;}
147:
148: .EmptyBox100{ display:block; width:100%; height:100px; overflow:hidden;}
149:
150:
151:
152:
153:
154:
155:
156:
157:
158:
159:
160:
161:
162:
163:
164:
165:
166:
167:
168:
169:
170:
171:
172:
173:
174:
175:
176:
177:
178:
179:
180:
181:
182:
183:
184:
185:
186:
187:
188:
189:
190:
191:
192:
193:
194:
195:
196:
197:
198:
199:
200:
201:
202:
203:
204:
205:
206:
207:
208:
209:
210:
211:
212:
213:
214:
215:
216:
217:
218:
219:
220:
221:
222:
223:
224:
225:
226:
227:
228:
229:
230:
231:
232:
233:
234:
235:
236:
237:
238:
239:
240:
241:
242:
243:
244:
245:
246:
247:
248:
249:
250:
251:
252:
253:
254:
255:
256:
257:
258:
259:
260:
261:
262:
263:
264:
265:
266:
267:
268:
269:
270:
271:
272:
273:
274:
275:
276:
277:
278:
279:
280:
281:
282:
283:
284:
285:
286:
287:
288:
289:
290:
291:
292:
293:
294:
295:
296:
297:
298:
299:
300:
301:
302:
303:
304:
305:
306:
307:
308:
309:
310:
311:
312:
313:
314:
315:
316:
317:
318:
319:
320:
321:
322:
323:
324:
325:
326:
327:
328:
329:
330:
331:
332:
333:
334:
335:
336:
337:
338:
339:
340:
341:
342:
343:
344:
345:
346:
347:
348:
349:
350:
351:
352:
353:
354:
355:
356:
357:
358:
359:
360:
361:
362:
363:
364:
365:
366:
367:
368:
369:
370:
371:
372:
373:
374:
375:
376:
377:
378:
379:
380:
381:
382:
383:
384:
385:
386:
387:
388:
389:
390:
391:
392:
393:
394:
395:
396:
397:
398:
399:
400:
401:
402:
403:
404:
405:
406:
407:
408:
409:
410:
411:
412:
413:
414:
415:
416:
417:
418:
419:
420:
421:
422:
423:
424:
425:
426:
427:
428:
429:
430:
431:
432:
433:
434:
435:
436:
437:
438:
439:
440:
441:
442:
443:
444:
445:
446:
447:
448:
449:
450:
451:
452:
453:
454:
455:
456:
457:
458:
459:
460:
461:
462:
463:
464:
465:
466:
467:
468:
469:
470:
471:
472:
473:
474:
475:
476:
477:
478:
479:
480:
481:
482:
483:
484:
485:
486:
487:
488:
489:
490:
491:
492:
493:
494:
495:
496:
497:
498:
499:
500:
501:
502:
503:
504:
505:
506:
507:
508:
509:
510:
511:
512:
513:
514:
515:
516:
517:
518:
519:
520:
521:
522:
523:
524:
525:
526:
527:
528:
529:
530:
531:
532:
533:
534:
535:
536:
537:
538:
539:
540:
541:
542:
543:
544:
545:
546:
547:
548:
549:
550:
551:
552:
553:
554:
555:
556:
557:
558:
559:
560:
561:
562:
563:
564:
565:
566:
567:
568:
569:
570:
571:
572:
573:
574:
575:
576:
577:
578:
579:
580:
581:
582:
583:
584:
585:
586:
587:
588:
589:
590:
591:
592:
593:
594:
595:
596:
597:
598:
599:
600:
601:
602:
603:
604:
605:
606:
607:
608:
609:
610:
611:
612:
613:
614:
615:
616:
617:
618:
619:
620:
621:
622:
623:
624:
625:
626:
627:
628:
629:
630:
631:
632:
633:
634:
635:
636:
637:
638:
639:
640:
641:
642:
643:
644:
645:
646:
647:
648:
649:
650:
651:
652:
653:
654:
655:
656:
657:
658:
659:
660:
661:
662:
663:
664:
665:
666:
667:
668:
669:
670:
671:
672:
673:
674:
675:
676:
677:
678:
679:
680:
681:
682:
683:
684:
685:
686:
687:
688:
689:
690:
691:
692:
693:
694:
695:
696:
697:
698:
699:
700:
701:
702:
703:
704:
705:
706:
707:
708:
709:
710:
711:
712:
713:
714:
715:
716:
717:
718:
719:
720:
721:
722:
723:
724:
725:
726:
727:
728:
729:
730:
731:
732:
733:
734:
735:
736:
737:
738:
739:
740:
741:
742:
743:
744:
745:
746:
747:
748:
749:
750:
751:
752:
753:
754:
755:
756:
757:
758:
759:
760:
761:
762:
763:
764:
765:
766:
767:
768:
769:
770:
771:
772:
773:
774:
775:
776:
777:
778:
779:
780:
781:
782:
783:
784:
785:
786:
787:
788:
789:
790:
791:
792:
793:
794:
795:
796:
797:
798:
799:
800:
801:
802:
803:
804:
805:
806:
807:
808:
809:
810:
811:
812:
813:
814:
815:
816:
817:
818:
819:
820:
821:
822:
823:
824:
825:
826:
827:
828:
829:
830:
831:
832:
833:
834:
835:
836:
837:
838:
839:
840:
841:
842:
843:
844:
845:
846:
847:
848:
849:
850:
851:
852:
853:
854:
855:
856:
857:
858:
859:
860:
861:
862:
863:
864:
865:
866:
867:
868:
869:
870:
871:
872:
873:
874:
875:
876:
877:
878:
879:
880:
881:
882:
883:
884:
885:
886:
887:
888:
889:
890:
891:
892:
893:
894:
895:
896:
897:
898:
899:
900:
901:
902:
903:
904:
905:
906:
907:
908:
909:
910:
911:
912:
913:
914:
915:
916:
917:
918:
919:
920:
921:
922:
923:
924:
925:
926:
927:
928:
929:
930:
931:
932:
933:
934:
935:
936:
937:
938:
939:
940:
941:
942:
943:
944:
945:
946:
947:
948:
949:
950:
951:
952:
953:
954:
955:
956:
957:
958:
959:
960:
961:
962:
963:
964:
965:
966:
967:
968:
969:
970:
971:
972:
973:
974:
975:
976:
977:
978:
979:
980:
981:
982:
983:
984:
985:
986:
987:
988:
989:
990:
991:
992:
993:
994:
995:
996:
997:
998:
999:
1000:
1001:
1002:
1003:
1004:
1005:
1006:
1007:
1008:
1009:
1010:
1011:
1012:
1013:
1014:
1015:
1016:
1017:
1018:
1019:
1020:
1021:
1022:
1023:
1024:
1025:
1026:
1027:
1028:
1029:
1030:
1031:
1032:
1033:
1034:
1035:
1036:
1037:
1038:
1039:
1040:
1041:
1042:
1043:
1044:
1045:
1046:
1047:
1048:
1049:
1050:
1051:
1052:
1053:
1054:
1055:
1056:
1057:
1058:
1059:
1060:
1061:
1062:
1063:
1064:
1065:
1066:
1067:
1068:
1069:
1070:
1071:
1072:
1073:
1074:
1075:
1076:
1077:
1078:
1079:
1080:
1081:
1082:
1083:
1084:
1085:
1086:
1087:
1088:
1089:
1090:
1091:
1092:
1093:
1094:
1095:
1096:
1097:
1098:
1099:
1100:
1101:
1102:
1103:
1104:
1105:
1106:
1107:
1108:
1109:
1110:
1111:
1112:
1113:
1114:
1115:
1116:
1117:
1118:
1119:
1120:
1121:
1122:
1123:
1124:
1125:
1126:
1127:
1128:
1129:
1130:
1131:
1132:
1133:
1134:
1135:
1136:
1137:
1138:
1139:
1140:
1141:
1142:
1143:
1144:
1145:
1146:
1147:
1148:
1149:
1150:
1151:
1152:
1153:
1154:
1155:
1156:
1157:
1158:
1159:
1160:
1161:
1162:
1163:
1164:
1165:
1166:
1167:
1168:
1169:
1170:
1171:
1172:
1173:
1174:
1175:
1176:
1177:
1178:
1179:
1180:
1181:
1182:
1183:
1184:
1185:
1186:
1187:
1188:
1189:
1190:
1191:
1192:
1193:
1194:
1195:
1196:
1197:
1198:
1199:
1200:
1201:
1202:
1203:
1204:
1205:
1206:
1207:
1208:
1209:
1210:
1211:
1212:
1213:
1214:
1215:
1216:
1217:
1218:
1219:
1220:
1221:
1222:
1223:
1224:
1225:
1226:
1227:
1228:
1229:
1230:
1231:
1232:
1233:
1234:
1235:
1236:
1237:
1238:
1239:
1240:
1241:
1242:
1243:
1244:
1245:
1246:
1247:
1248:
1249:
1250:
1251:
1252:
1253:
1254:
1255:
1256:
1257:
1258:
1259:
1260:
1261:
1262:
1263:
1264:
1265:
1266:
1267:
1268:
1269:
1270:
1271:
1272:
1273:
1274:
1275:
1276:
1277:
1278:
1279:
1280:
1281:
1282:
1283:
1284:
1285:
1286:
1287:
1288:
1289:
1290:
1291:
1292:
1293:
1294:
1295:
1296:
1297:
1298:
1299:
1300:
1301:
1302:
1303:
1304:
1305:
1306:
1307:
1308:
1309:
1310:
1311:
1312:
1313:
1314:
1315:
1316:
1317:
1318:
1319:
1320:
1321:
1322:
1323:
1324:
1325:
1326:
1327:
1328:
1329:
1330:
1331:
1332:
1333:
1334:
1335:
1336:
1337:
1338:
1339:
1340:
1341:
1342:
1343:
1344:
1345:
1346:
1347:
1348:
1349:
1350:
1351:
1352:
1353:
1354:
1355:
1356:
1357:
1358:
1359:
1360:
1361:
1362:
1363:
1364:
1365:
1366:
1367:
1368:
1369:
1370:
1371:
1372:
1373:
1374:
1375:
1376:
1377:
1378:
1379:
1380:
1381:
1382:
1383:
1384:
1385:
1386:
1387:
1388:
1389:
1390:
1391:
1392:
1393:
1394:
1395:
1396:
1397:
1398:
1399:
1400:
1401:
1402:
1403:
1404:
1405:
1406:
1407:
1408:
1409:
1410:
1411:
1412:
1413:
1414:
1415:
1416:
1417:
1418:
1419:
1420:
1421:
1422:
1423:
1424:
1425:
1426:
1427:
1428:
1429:
1430:
1431:
1432:
1433:
1434:
1435:
1436:
1437:
1438:
1439:
1440:
1441:
1442:
1443:
1444:
1445:
1446:
1447:
1448:
1449:
1450:
1451:
1452:
1453:
1454:
1455:
1456:
1457:
1458:
1459:
1460:
1461:
1462:
1463:
1464:
1465:
1466:
1467:
1468:
1469:
1470:
1471:
1472:
1473:
1474:
1475:
1476:
1477:
1478:
1479:
1480:
1481:
1482:
1483:
1484:
1485:
1486:
1487:
1488:
1489:
1490:
1491:
1492:
1493:
1494:
1495:
1496:
1497:
1498:
1499:
1500:
1501:
1502:
1503:
1504:
1505:
1506:
1507:
1508:
1509:
1510:
1511:
1512:
1513:
1514:
1515:
1516:
1517:
1518:
1519:
1520:
1521:
1522:
1523:
1524:
1525:
1526:
1527:
1528:
1529:
1530:
1531:
1532:
1533:
1534:
1535:
1536:
1537:
1538:
1539:
1540:
1541:
1542:
1543:
1544:
1545:
1546:
1547:
1548:
1549:
1550:
1551:
1552:
1553:
1554:
1555:
1556:
1557:
1558:
1559:
1560:
1561:
1562:
1563:
1564:
1565:
1566:
1567:
1568:
1569:
1570:
1571:
1572:
1573:
1574:
1575:
1576:
1577:
1578:
1579:
1580:
1581:
1582:
1583:
1584:
1585:
1586:
1587:
1588:
1589:
1590:
1591:
1592:
1593:
1594:
1595:
1596:
1597:
1598:
1599:
1600:
1601:
1602:
1603:
1604:
1605:
1606:
1607:
1608:
1609:
1610:
1611:
1612:
1613:
1614:
1615:
1616:
1617:
1618:
1619:
1620:
1621:
1622:
1623:
1624:
1625:
1626:
1627:
1628:
1629:
1630:
1631:
1632:
1633:
1634:
1635:
1636:
1637:
1638:
1639:
1640:
1641:
1642:
1643:
1644:
1645:
1646:
1647:
1648:
1649:
1650:
1651:
1652:
1653:
1654:
1655:
1656:
1657:
1658:
1659:
1660:
1661:
1662:
1663:
1664:
1665:
1666:
1667:
1668:
1669:
1670:
1671:
1672:
1673:
1674:
1675:
1676:
1677:
1678:
1679:
1680:
1681:
1682:
1683:
1684:
1685:
1686:
1687:
1688:
1689:
1690:
1691:
1692:
1693:
1694:
1695:
1696:
1697:
1698:
1699:
1700:
1701:
1702:
1703:
1704:
1705:
1706:
1707:
1708:
1709:
1710:
1711:
1712:
1713:
1714:
1715:
1716:
1717:
1718:
1719:
1720:
1721:
1722:
1723:
1724:
1725:
1726:
1727:
1728:
1729:
1730:
1731:
1732:
1733:
1734:
1735:
1736:
1737:
1738:
1739:
1740:
1741:
1742:
1743:
1744:
1745:
1746:
1747:
1748:
1749:
1750:
1751:
1752:
1753:
1754:
1755:
1756:
1757:
1758:
1759:
1760:
1761:
1762:
1763:
1764:
1765:
1766:
1767:
1768:
1769:
1770:
1771:
1772:
1773:
1774:
1775:
1776:
1777:
1778:
1779:
1780:
1781:
1782:
1783:
1784:
1785:
1786:
1787:
1788:
1789:
1790:
1791:
1792:
1793:
1794:
1795:
1796:
1797:
1798:
1799:
1800:
1801:
1802:
1803:
1804:
1805:
1806:
1807:
1808:
1809:
1810:
1811:
1812:
1813:
1814:
1815:
1816:
1817:
1818:
1819:
1820:
1821:
1822:
1823:
1824:
1825:
1826:
1827:
1828:
1829:
1830:
1831:
1832:
1833:
1834:
1835:
1836:
1837:
1838:
1839:
1840:
1841:
1842:
1843:
1844:
1845:
1846:
1847:
1848:
1849:
1850:
1851:
1852:
1853:
1854:
1855:
1856:
1857:
1858:
1859:
1860:
1861:
1862:
1863:
1864:
1865:
1866:
1867:
1868:
1869:
1870:
1871:
1872:
1873:
1874:
1875:
1876:
1877:
1878:
1879:
1880:
1881:
1882:
1883:
1884:
1885:
1886:
1887:
1888:
1889:
1890:
1891:
1892:
1893:
1894:
1895:
1896:
1897:
1898:
1899:
1900:
1901:
1902:
1903:
1904:
1905:
1906:
1907:
1908:
1909:
1910:
1911:
1912:
1913:
1914:
1915:
1916:
1917:
1918:
1919:
1920:
1921:
1922:
1923:
1924:
1925:
1926:
1927:
1928:
1929:
1930:
1931:
1932:
1933:
1934:
1935:
1936:
1937:
1938:
1939:
1940:
1941:
1942:
1943:
1944:
1945:
1946:
1947:
1948:
1949:
1950:
1951:
1952:
1953:
1954:
1955:
1956:
1957:
1958:
1959:
1960:
1961:
1962:
1963:
1964:
1965:
1966:
1967:
1968:
1969:
1970:
1971:
1972:
1973:
1974:
1975:
1976:
1977:
1978:
1979:
1980:
1981:
1982:
1983:
1984:
1985:
1986:
1987:
1988:
1989:
1990:
1991:
1992:
1993:
1994:
1995:
1996:
1997:
1998:
1999:
2000:
2001:
2002:
2003:
2004:
2005:
2006:
2007:
2008:
2009:
2010:
2011:
2012:
2013:
2014:
2015:
2016:
2017:
2018:
2019:
2020:
2021:
2022:
2023:
2024:
2025:
2026:
2027:
2028:
2029:
2030:
2031:
2032:
2033:
2034:
2035:
2036:
2037:
2038:
2039:
2040:
2041:
2042:
2043:
2044:
2045:
2046:
2047:
2048:
2049:
2050:
2051:
2052:
2053:
2054:
2055:
2056:
2057:
2058:
2059:
2060:
2061:
2062:
2063:
2064:
2065:
2066:
2067:
2068:
2069:
2070:
2071:
2072:
2073:
2074:
2075:
2076:
2077:
2078:
2079:
2080:
2081:
2082:
```

```
135: }
136:
137: .outerBackground{
138:     background-color: #000000;
139: }
140:
141: .profilePanel{
142:     background-image: none;
143:     background-color: black;
144: }
145:
146: .noBackground{
147:     background-image: none;
148:     background-color: transparent;
149: }
150:
151: body{
152:     font-family: 'Montserrat', sans-serif !important;
153:     font-size: 1em;
154:     color: white;
155: }
156:
157: .padding03em{
158:     padding: 0.5em;
159: }
160:
161: .noBorder{
162:     border: 0px;
163: }
```