

```

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 fr.insalyon.dasi.gustatif.metier.service;
7
8  import com.google.maps.model.LatLng;
9  import fr.insalyon.dasi.gustatif.dao.ClientDao;
10 import fr.insalyon.dasi.gustatif.dao.CommandeDao;
11 import fr.insalyon.dasi.gustatif.dao.JpaUtil;
12 import fr.insalyon.dasi.gustatif.dao.LivreurDao;
13 import fr.insalyon.dasi.gustatif.dao.LivreurDroneDao;
14 import fr.insalyon.dasi.gustatif.dao.LivreurVeloDao;
15 import fr.insalyon.dasi.gustatif.dao.ProduitDao;
16 import fr.insalyon.dasi.gustatif.dao.RestaurantDao;
17 import fr.insalyon.dasi.gustatif.metier.modele.Client;
18 import fr.insalyon.dasi.gustatif.metier.modele.Commande;
19 import fr.insalyon.dasi.gustatif.metier.modele.LigneDeCommande;
20 import fr.insalyon.dasi.gustatif.metier.modele.Produit;
21 import fr.insalyon.dasi.gustatif.metier.modele.Restaurant;
22 import fr.insalyon.dasi.gustatif.metier.modele.Livreur;
23 import fr.insalyon.dasi.gustatif.metier.modele.LivreurVelo;
24 import fr.insalyon.dasi.gustatif.metier.modele.LivreurDrone;
25 import fr.insalyon.dasi.gustatif.util.GeoTest;
26 import java.util.Date;
27 import java.util.List;
28 import java.util.Scanner;
29 import java.util.logging.Level;
30 import java.util.logging.Logger;
31 import javax.persistence.RollbackException;
32
33 /**
34 *
35 * @author Nico
36 */
37 public class ServiceMetier {
38
39     private final ClientDao clientDao = new ClientDao();
40     private final RestaurantDao restaurantDao = new RestaurantDao();
41     private final ProduitDao produitDao = new ProduitDao();
42     private final CommandeDao commandeDao = new CommandeDao();
43     private final LivreurDao livreurDao = new LivreurDao();
44     private final LivreurVeloDao livreurVeloDao = new LivreurVeloDao();
45     private final LivreurDroneDao livreurDroneDao = new
46     LivreurDroneDao();
47     private final ServiceTechnique serviceTechnique = new
48     ServiceTechnique();
49
50     public ServiceMetier() {
51
52     }
53
54     public boolean createClient(Client client)

```

```

52     {
53         boolean result = false;
54
55         LatLng latLng = GeoTest.getLatLng(client.getAdresse());
56
57         //adresse invalide car ne permet pas de calculer une LatLng
58         if(latLng != null) {
59             client.setCoordonnees(latLng);
60             JpaUtil.creerEntityManager();
61             try {
62
63                 JpaUtil.ouvrirTransaction();
64                 clientDao.create(client);
65                 JpaUtil.validerTransaction();
66
67                 result = true;
68             } catch (RollbackException e) {
69
70             } catch (Exception e) {
71                 //System.out.println(e);
72             } catch (Throwable ex) {
73
74                 //Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
75             }
76             JpaUtil.fermerEntityManager();
77         }
78
79         // envoi du mail
80         String recipient = client.getMail();
81         String subject = null;
82         String body = null;
83         if(result)
84         {
85             //envoyer mail succes
86             subject = "Bienvenue chez Gustat'IF";
87             body = "Bonjour " + client.getPrenom() + ",\n"
88                 + "\n"
89                 + "          Nous vous confirmons votre inscription au
90                 service "
91                 + "Gustat'IF. Votre numéro de client est : " +
92                 client.getId() + ".";
93         } else {
94             //envoyer mail echec
95             subject = "Bienvenue chez Gustat'IF";
96             body = "Bonjour " + client.getPrenom() + ",\n"
97                 + "\n"
98                 + "          Votre inscription au service Gustat'IF a
99                 malencontreusement"
100                 + " échoué... Merci de recommencer ultérieurement.";
101         }
102         serviceTechnique.sendFakeMail(recipient, subject, body);
103         //serviceTechnique.sendRealMail(recipient, subject, body);

```

```

101         return result;
102     }
103
104     public boolean clientExists(String mail, String motDePasse) {
105         boolean result = false;
106
107         //aucun argument ne doit être égale à null
108         if(mail != null && motDePasse != null)
109         {
110             //tous les champs doivent être renseignés
111             if(!mail.isEmpty() && !motDePasse.isEmpty())
112             {
113                 JpaUtil.creerEntityManager();
114                 try {
115
116                     //si le client existe en BD
117                     if(clientDao.findByMailAndPassword(mail,
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151

```

```

152     }
153
154     public Client findClientById(Long id) {
155         JpaUtil.creerEntityManager();
156
157         Client client = null;
158
159         try {
160             client = clientDao.findById(id);
161         } catch (Exception e) {
162             System.out.println(e);
163         } catch (Throwable ex) {
164
165             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
166         }
167
168         JpaUtil.fermerEntityManager();
169
170         return client;
171     }
172
173     public Client findClientByMailAndPassword(String mail, String password) {
174         JpaUtil.creerEntityManager();
175
176         Client client = null;
177
178         try {
179             client = clientDao.findByMailAndPassword(mail, password);
180         } catch (Exception e) {
181             System.out.println(e);
182         } catch (Throwable ex) {
183
184             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
185         }
186
187         JpaUtil.fermerEntityManager();
188
189         return client;
190     }
191
192     public Client findClientByMail(String mail) {
193         JpaUtil.creerEntityManager();
194
195         Client client = null;
196
197         try {
198             client = clientDao.findByMail(mail);
199         } catch (Exception e) {
200             System.out.println(e);
201         } catch (Throwable ex) {

```

```
201         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
202     }
203
204     JpaUtil.fermerEntityManager();
205
206     return client;
207 }
208
209 public List<Client> findAllClients() {
210     JpaUtil.creerEntityManager();
211
212     List<Client> clients = null;
213
214     try {
215         clients = clientDao.findAll();
216     } catch (Exception e) {
217         System.out.println(e);
218     } catch (Throwable ex) {
219         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
220     }
221
222     JpaUtil.fermerEntityManager();
223
224     return clients;
225 }
226
227 public Restaurant findRestaurantById(Long id) {
228     JpaUtil.creerEntityManager();
229
230     Restaurant restaurant = null;
231
232     try {
233         restaurant = restaurantDao.findById(id);
234     } catch (Exception e) {
235         System.out.println(e);
236     } catch (Throwable ex) {
237         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
238     }
239
240     JpaUtil.fermerEntityManager();
241
242     return restaurant;
243 }
244
245 public List<Restaurant> findAllRestaurants() {
246     JpaUtil.creerEntityManager();
247
248     List<Restaurant> restaurants = null;
```

```
249
250     try {
251         restaurants = restaurantDao.findAll();
252     } catch (Exception e) {
253         System.out.println(e);
254     } catch (Throwable ex) {
255
256         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
257     }
258     JpaUtil.fermerEntityManager();
259
260     return restaurants;
261 }
262
263 public Produit findProduitById(Long id) {
264     JpaUtil.creerEntityManager();
265
266     Produit produit = null;
267
268     try {
269         produit = produitDao.findById(id);
270     } catch (Exception e) {
271         System.out.println(e);
272     } catch (Throwable ex) {
273
274         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
275     }
276     JpaUtil.fermerEntityManager();
277
278     return produit;
279 }
280
281 public List<Produit> findAllProduits() {
282     JpaUtil.creerEntityManager();
283
284     List<Produit> produits = null;
285
286     try {
287         produits = produitDao.findAll();
288     } catch (Exception e) {
289         System.out.println(e);
290     } catch (Throwable ex) {
291
292         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
293     }
294     JpaUtil.fermerEntityManager();
295
296     return produits;
```

```

297     }
298
299     public boolean createCommande(Commande commande) {
300         JpaUtil.creerEntityManager();
301         boolean result = false;
302         Livreur livreur = null;
303         int nbTentative = 0;
304         while(result == false && nbTentative < 10) {
305             try {
306
307                 JpaUtil.ouvrirTransaction();
308                 List<Livreur> livreurs =
309                     livreurDao.findAllDisponible(commande.getPoidsEnGrammes(
310                         ));
311                 livreur =
312                     serviceTechnique.chooseBestLivreur(commande, livreurs);
313                 if(livreur != null){
314                     commande.setDateDebut(new Date());
315                     commande.setLivreur(livreur);
316                     commandeDao.create(commande);
317                     livreur.addCommande(commande);
318                     livreur.setDisponible(false);
319                     livreurDao.update(livreur);
320
321                     //a commenté pour pouvoir l'utiliser en ihm non
322                     console (sert juste au test)
323                     Scanner sc = new Scanner(System.in);
324                     System.out.println("Press enter...");
325                     sc.nextLine();
326                     //_____
327
328                     JpaUtil.validerTransaction();
329                     result = true;
330                 } else {
331                     JpaUtil.annulerTransaction();
332                 }
333             } catch (Exception e) {
334                 System.out.println(e);
335                 JpaUtil.annulerTransaction();
336             } catch (Throwable ex) {
337
338                 Logger.getLogger(ServiceMetier.class.getName()).log(Leve
339                     l.SEVERE, null, ex);
340                 JpaUtil.annulerTransaction();
341             }
342             nbTentative++;
343         }
344         JpaUtil.fermerEntityManager();
345
346         // envoi du mail
347         String recipient = null;
348         String subject = null;
349         String body = null;
350         if(result && livreur instanceof LivreurVelo)

```

```

345     {
346         //envoyer mail succes
347         recipient = livreur.getMail();
348         subject = "Livraison n°" + commande.getId() + " à effectuer";
349         body = "Bonjour " + ((LivreurVelo)livreur).getPrenom() + ",\n"
350             + "\n      Merci d'effectuer cette livraison dès
351             maintenant, "
352             + "\n      tout en respectant le code de la route ;-)"
353             + "\n\nLe Chef"
354             + "\n\nDétails de la Livraison"
355             + "\n      - Date/heure : " +
356             commande.getDateDebut().toString()
357             + "\n      - Livreur : " +
358             ((LivreurVelo)livreur).getPrenom() + " "
359             + ((LivreurVelo)livreur).getNom().toUpperCase() +
360             " (n°"
361             + livreur.getId() + ")"
362             + "\n      - Restaurant : " +
363             commande.getRestaurant().getDenomination()
364             + "\n      - Client : "
365             + "\n      " + commande.getClient().getPrenom() + " "
366             + commande.getClient().getNom().toUpperCase()
367             + "\n      " + commande.getClient().getAdresse()
368             + "\n      " + commande.getClient().getTelephone()
369             + "\n\n Commande :";
370
371         for (LigneDeCommande ligne :
372             commande.getLignesDeCommande()) {
373             body += "\n      - " + ligne.getQuantite() + " "
374                 + ligne.getProduit().getDenomination() + " "
375                 + ligne.getQuantite() + " x " +
376                 ligne.getPrixUnitaire()
377                 + "€";
378         }
379         body += "\n\nTOTAL : " + commande.getPrix() + "€";
380         serviceTechnique.sendFakeMail(recipient, subject, body);
381         //serviceTechnique.sendRealMail(recipient, subject, body);
382     }
383     recipient = commande.getClient().getMail();
384     if(result){
385         //envoyer mail succes
386         subject = "Commande n°"+commande.getId();
387         body = "Bonjour " + commande.getClient().getPrenom() + ",\n"
388             + "\n"
389             + "      Nous vous confirmons votre commande n°" +
390             commande.getId()
391             + " d'un montant de " + commande.getPrix() + "€.";
392     } else {
393         subject = "Echec commande";
394         body = "Bonjour " + commande.getClient().getPrenom() + ",\n"
395             + "\n"
396             + "      La validation de votre commande a échoué,
397             veuillez "
398             + "réessayer ultérieurement.";

```



```
390     }
391     }
392     serviceTechnique.sendFakeMail(recipient, subject, body);
393     //serviceTechnique.sendRealMail(recipient, subject, body);
394
395     return result;
396 }
397
398 public Commande validateCommande(Commande commande) {
399     JpaUtil.creerEntityManager();
400
401     try {
402         if(commande.getDateFin() != null) {
403             JpaUtil.ouvrirTransaction();
404             commandeDao.update(commande);
405             commande.getLivreur().setDisponible(Boolean.TRUE);
406             livreurDao.update(commande.getLivreur());
407             JpaUtil.validerTransaction();
408         } else {
409             commande = null;
410         }
411     } catch (Exception e) {
412         System.out.println(e);
413         commande = null;
414         JpaUtil.annulerTransaction();
415     } catch (Throwable ex) {
416
417         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
418             "VERE, null, ex");
419         commande = null;
420         JpaUtil.annulerTransaction();
421     }
422
423     JpaUtil.fermerEntityManager();
424
425     return commande;
426 }
427
428 public List<Commande> findCommandesByLivreurId(Long id) {
429     JpaUtil.creerEntityManager();
430
431     List<Commande> commandes = null;
432
433     try {
434         commandes = commandeDao.findByLivreurID(id);
435     } catch (Exception e) {
436         System.out.println(e);
437     } catch (Throwable ex) {
438
439         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
440             "VERE, null, ex");
441     }
442
443     JpaUtil.fermerEntityManager();
```

```
440         return commandes;
441     }
442
443     public Commande findNotEndedByLivreur(Livreur livreur) {
444         JpaUtil.creerEntityManager();
445
446         Commande commande = null;
447
448         try {
449             commande = commandeDao.findNotEndedByLivreur(livreur);
450         } catch (Exception e) {
451             System.out.println(e);
452         } catch (Throwable ex) {
453
454             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
455         }
456
457         JpaUtil.fermerEntityManager();
458
459         return commande;
460     }
461     public List<Commande> findAllCommandes() {
462         JpaUtil.creerEntityManager();
463
464         List<Commande> commandes = null;
465
466         try {
467             commandes = commandeDao.findAll();
468         } catch (Exception e) {
469             System.out.println(e);
470         } catch (Throwable ex) {
471
472             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
473         }
474
475         JpaUtil.fermerEntityManager();
476
477         return commandes;
478     }
479     public Commande findCommandeById(Long id) {
480         JpaUtil.creerEntityManager();
481
482         Commande commande = null;
483
484         try {
485             commande = commandeDao.findById(id);
486         } catch (Exception e) {
487             System.out.println(e);
488         } catch (Throwable ex) {
489
490             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
491         }
492     }
```

```
VERE, null, ex);
489     }
490
491     JpaUtil.fermerEntityManager();
492
493     return commande;
494 }
495
496 public List<Commande> findAllCommandesNotEnded() {
497     JpaUtil.creerEntityManager();
498
499     List<Commande> commandes = null;
500
501     try {
502         commandes = commandeDao.findAllNotEnded();
503     } catch (Exception e) {
504         System.out.println(e);
505     } catch (Throwable ex) {
506
507         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
508             VERE, null, ex);
509
510     }
511
512     JpaUtil.fermerEntityManager();
513
514     return commandes;
515 }
516
517 public Livreur updateLivreur(Livreur livreur) {
518     JpaUtil.creerEntityManager();
519
520     try {
521         JpaUtil.ouvrirTransaction();
522         livreur = livreurDao.update(livreur);
523         JpaUtil.validerTransaction();
524     } catch (Exception e) {
525         System.out.println(e);
526         livreur = null;
527         JpaUtil.annulerTransaction();
528     } catch (Throwable ex) {
529
530         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
531             VERE, null, ex);
532         livreur = null;
533         JpaUtil.annulerTransaction();
534
535     }
536
537     JpaUtil.fermerEntityManager();
538
539     return livreur;
540 }
```

```

538
539     public LivreurVelo findLivreurVeloByMailAndPassword(String mail,
String password) {
540         JpaUtil.creerEntityManager();
541
542         LivreurVelo livreur = null;
543
544         try {
545             livreur =
                livreurVeloDao.findLivreurVeloByMailAndPassword(mail,
                password);
546         } catch (Exception e) {
547             System.out.println(e);
548         } catch (Throwable ex) {
549             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
                null, ex);
550         }
551
552         JpaUtil.fermerEntityManager();
553
554         return livreur;
555     }
556
557     public List<LivreurDrone>
findLivreursDronesByMailAndPassword(String mail, String password) {
558         JpaUtil.creerEntityManager();
559
560         List<LivreurDrone> livreurs = null;
561
562         try {
563             livreurs =
                livreurDroneDao.findLivreursDronesByMailAndPassword(mail,
                password);
564         } catch (Exception e) {
565             System.out.println(e);
566         } catch (Throwable ex) {
567             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
                null, ex);
568         }
569
570         JpaUtil.fermerEntityManager();
571
572         return livreurs;
573     }
574
575     public Livreur findLivreurById(Long id) {
576         JpaUtil.creerEntityManager();
577
578         Livreur livreur = null;
579
580         try {
581             livreur = livreurDao.findById(id);

```

```
582         } catch (Exception e) {
583             System.out.println(e);
584         } catch (Throwable ex) {
585             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
586         }
587
588         JpaUtil.fermerEntityManager();
589
590         return livreur;
591     }
592
593     public List<Livreur> findAllLivres() {
594         JpaUtil.creerEntityManager();
595
596         List<Livreur> livres = null;
597
598         try {
599             livres = livreurDao.findAll();
600         } catch (Exception e) {
601             System.out.println(e);
602         } catch (Throwable ex) {
603             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
604         }
605
606         JpaUtil.fermerEntityManager();
607
608         return livres;
609     }
610
611     public List<LivreurDrone> findAllLivresDrones() {
612         JpaUtil.creerEntityManager();
613
614         List<LivreurDrone> livres = null;
615
616         try {
617             livres = livreurDroneDao.findAllLivresDrones();
618         } catch (Exception e) {
619             System.out.println(e);
620         } catch (Throwable ex) {
621             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
622         }
623
624         JpaUtil.fermerEntityManager();
625
626         return livres;
627     }
628
629     public List<LivreurVelo> findAllLivresVelos() {
```

```
630         JpaUtil.creerEntityManager();
631
632         List<LivreurVelo> livreurs = null;
633
634         try {
635             livreurs = livreurVeloDao.findAllLivresVelo();
636         } catch (Exception e) {
637             System.out.println(e);
638         } catch (Throwable ex) {
639
640             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
641
642         }
643
644         JpaUtil.fermerEntityManager();
645
646         return livreurs;
647     }
648
649     public List<Livreur> findAllLivresDisponibles() {
650         JpaUtil.creerEntityManager();
651
652         List<Livreur> livreurs = null;
653
654         try {
655             livreurs = livreurDao.findAllDisponible();
656         } catch (Exception e) {
657             System.out.println(e);
658         } catch (Throwable ex) {
659
660             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
661
662         }
663
664         JpaUtil.fermerEntityManager();
665
666         return livreurs;
667     }
668 }
```