

Metier.Service

```
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                 livreur =
311                     serviceTechnique.chooseBestLivreur(commande, livreurs);
312                 if(livreur != null){
313                     commande.setDateDebut(new Date());
314                     commande.setLivreur(livreur);
315                     commandeDao.create(commande);
316                     livreur.addCommande(commande);
317                     livreur.setDisponible(false);
318                     livreurDao.update(livreur);
319
320                     //a commenté pour pouvoir l'utiliser en ihm non
321                     console (sert juste au test)
322                     Scanner sc = new Scanner(System.in);
323                     System.out.println("Press enter...");
324                     sc.nextLine();
325                     //_____
326
327                     JpaUtil.validerTransaction();
328                     result = true;
329                 } else {
330                     JpaUtil.annulerTransaction();
331                 }
332             } catch (Exception e) {
333                 System.out.println(e);
334                 JpaUtil.annulerTransaction();
335             } catch (Throwable ex) {
336
337                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
338                 JpaUtil.annulerTransaction();
339             }
340             nbTentative++;
341         }
342         JpaUtil.fermerEntityManager();
343
344         // envoi du mail
345         String recipient = null;
346         String subject = null;
347         String body = null;
348         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     serviceTechnique.sendFakeMail(recipient, subject, body);
392     //serviceTechnique.sendRealMail(recipient, subject, body);
393
394     return result;
395 }
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, null, ex);
418         commande = null;
419         JpaUtil.annulerTransaction();
420     }
421
422     JpaUtil.fermerEntityManager();
423
424     return commande;
425 }
426
427 public List<Commande> findCommandesByLivreurId(Long id) {
428     JpaUtil.creerEntityManager();
429
430     List<Commande> commandes = null;
431
432     try {
433         commandes = commandeDao.findByLivreurID(id);
434     } catch (Exception e) {
435         System.out.println(e);
436     } catch (Throwable ex) {
437
438         Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
439     }
440
441     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,
540 String password) {
541         JpaUtil.creerEntityManager();
542         LivreurVelo livreur = null;
543
544         try {
545             livreur =
546                 livreurVeloDao.findLivreurVeloByMailAndPassword(mail,
547                 password);
548         } catch (Exception e) {
549             System.out.println(e);
550         } catch (Throwable ex) {
551             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
552                 null, ex);
553         }
554         JpaUtil.fermerEntityManager();
555         return livreur;
556     }
557     public List<LivreurDrone>
558     findLivreaursDronesByMailAndPassword(String mail, String password) {
559         JpaUtil.creerEntityManager();
560         List<LivreurDrone> livreaurs = null;
561
562         try {
563             livreaurs =
564                 livreurDroneDao.findLivreaursDronesByMailAndPassword(mail,
565                 password);
566         } catch (Exception e) {
567             System.out.println(e);
568         } catch (Throwable ex) {
569             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE,
570                 null, ex);
571         }
572         JpaUtil.fermerEntityManager();
573         return livreaurs;
574     }
575     public Livreur findLivreurById(Long id) {
576         JpaUtil.creerEntityManager();
577         Livreur livreur = null;
578
579         try {
580             livreur = livreurDao.findById(id);
581
```

```

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         JpaUtil.fermerEntityManager();
644
645         return livreurs;
646     }
647     public List<Livreur> findAllLivresDisponibles() {
648         JpaUtil.creerEntityManager();
649
650         List<Livreur> livreurs = null;
651
652         try {
653             livreurs = livreurDao.findAllDisponible();
654         } catch (Exception e) {
655             System.out.println(e);
656         } catch (Throwable ex) {
657
658             Logger.getLogger(ServiceMetier.class.getName()).log(Level.SEVERE, null, ex);
659         }
660
661         JpaUtil.fermerEntityManager();
662
663         return livreurs;
664     }
665 }
666
667
```

```

1  /*
2  * To change this license header, choose License Headers in Project
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 fr.insalyon.dasi.gustatif.metier.modele.Commande;
9  import fr.insalyon.dasi.gustatif.metier.modele.Livreur;
10 import fr.insalyon.dasi.gustatif.metier.modele.LivreurDrone;
11 import fr.insalyon.dasi.gustatif.metier.modele.LivreurVelo;
12 import fr.insalyon.dasi.gustatif.util.GeoTest;
13 import fr.insalyon.dasi.gustatif.util.MailSender;
14 import java.util.List;
15
16 /**
17  *
18  * @author Nico
19  */
20 public class ServiceTechnique {
21
22     public boolean sendRealMail(String recipientMail, String subject,
23 String body) {
24         MailSender ms = new
25         MailSender("gustatif.b3125@gmail.com","B3125-2016");
26         return ms.sendMail(recipientMail,subject,body);
27     }
28
29     public void sendFakeMail(String recipientMail, String subject,
30 String body) {
31         System.out.println("");
32         System.out.println("***** Mail envoyé *****");
33         System.out.println("From : gustatif.b3125@gmail.com");
34         System.out.println("To : " + recipientMail);
35         System.out.println("Subject : " + subject);
36         System.out.println(body);
37         System.out.println("*****");
38         System.out.println("");
39     }
40
41     public Livreur chooseBestLivreur(Commande commande,List<Livreur>
42     livreurs )
43         throws Throwable {
44         Livreur livreurChoisi = null;
45         double temps;
46         double tempsFinal;
47
48         if(!livreurs.isEmpty()) {
49             livreurChoisi = livreurs.get(0);
50             if(livreurChoisi instanceof LivreurVelo){
51                 tempsFinal = GeoTest.getTripDurationByBicycleInMinute(

```



```
        livreurChoisi.getLatLng(), commande.getClient().getLatLng(),
        commande.getRestaurant().getLatLng());
50     } else if (livreurChoisi instanceof LivreurDrone){
51         tempsFinal = (
52             (GeoTest.getFlightDistanceInKm(livreurChoisi.getLatLng(),
53             commande.getRestaurant().getLatLng())
54             +
55             GeoTest.getFlightDistanceInKm(commande.getRestaurant().getLatLng(),
56             commande.getClient().getLatLng()))
57             /
58             ((LivreurDrone)livreurChoisi).getVitesseMoyenneDeVolEnKmH()) * 60;
59     } else {
60         return null;
61     }
62     for (int i=1; i < livreurs.size(); i++) {
63         if(livreurs.get(i) instanceof LivreurVelo){
64             temps = GeoTest.getTripDurationByBicycleInMinute(
65                 livreurs.get(i).getLatLng(),
66                 commande.getClient().getLatLng(),
67                 commande.getRestaurant().getLatLng());
68         } else if (livreurs.get(i) instanceof LivreurDrone){
69             temps = (
70                 (GeoTest.getFlightDistanceInKm(livreurs.get(i).getLatLng(),
71                 commande.getRestaurant().getLatLng())
72                 +
73                 GeoTest.getFlightDistanceInKm(commande.getRestaurant().getLatLng(),
74                 commande.getClient().getLatLng()) )
75                 /
76                 ((LivreurDrone)livreurs.get(i)).getVitesseMoyenneDeVolEnKmH()) * 60;
77         } else {
78             return null;
79         }
80         if(temps < tempsFinal) {
81             tempsFinal = temps;
82             livreurChoisi = livreurs.get(i);
83         }
84     }
85     return livreurChoisi;
86 }
```