Metier.Service

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package fr.insalyon.dasi.gustatif.metier.service;

import com.google.maps.model.LatLng;
import fr.insalyon.dasi.gustatif.dao.ClientDao;
import fr.insalyon.dasi.gustatif.dao.CommandeDao;
import fr.insalyon.dasi.gustatif.dao.JpaUtil;
import fr.insalyon.dasi.gustatif.dao.LivreurDao;
import fr.insalyon.dasi.gustatif.dao.LivreurDroneDao;
import fr.insalyon.dasi.gustatif.dao.LivreurVeloDao;
import fr.insalyon.dasi.gustatif.dao.ProduitDao;
import fr.insalyon.dasi.gustatif.dao.RestaurantDao;
import fr.insalyon.dasi.gustatif.metier.modele.Client;
import fr.insalyon.dasi.gustatif.metier.modele.Commande;
import fr.insalyon.dasi.gustatif.metier.modele.LigneDeCommande;
import fr.insalyon.dasi.gustatif.metier.modele.Produit;
import fr.insalyon.dasi.gustatif.metier.modele.Restaurant;
import fr.insalyon.dasi.gustatif.metier.modele.Livreur;
import fr.insalyon.dasi.gustatif.metier.modele.LivreurVelo;
import fr.insalyon.dasi.gustatif.metier.modele.LivreurDrone;
import fr.insalyon.dasi.gustatif.util.GeoTest;
import java.util.Date;
import java.util.List;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.RollbackException;

/**
 *
 * @author Nico
 */
public class ServiceMetier {

    private final ClientDao clientDao = new ClientDao();
    private final RestaurantDao restaurantDao = new RestaurantDao();
    private final ProduitDao produitDao = new ProduitDao();
    private final CommandeDao commandeDao = new CommandeDao();
    private final LivreurDao livreurDao = new LivreurDao();
    private final LivreurVeloDao livreurVeloDao = new LivreurVeloDao();
    private final LivreurDroneDao livreurDroneDao = new LivreurDroneDao();
    private final ServiceTechnique serviceTechnique = new ServiceTechnique();

    public ServiceMetier() {
    }

    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                  //Logger.getLogger(ServiceMetier.class.getName()).log(Le
                    vel.SEVERE, null, ex);
74              }
75              JpaUtil.fermerEntityManager();
76          }
77
78          // envoi du mail
79          String recipient = client.getMail();
80          String subject = null;
81          String body = null;
82          if(result)
83          {
84              //envoyer mail succes
85              subject = "Bienvenue chez Gustat'IF";
86              body = "Bonjour "+ client.getPrenom() +",\n"
87                  + "\n"
88                  + "       Nous vous confirmons votre inscription au
                    service "
89                  + "Gustat'IF. Votre numéro de client est : " +
                    client.getId() + ".";
90          } else {
91              //envoyer mail echec
92              subject = "Bienvenue chez Gustat'IF";
93              body = "Bonjour "+ client.getPrenom() +",\n"
94                  + "\n"
95                  + "       Votre inscription au service Gustat'IF a
                    malencontreusement"
96                  + " échoué... Merci de recommencer ultérieurement.";
97          }
98          serviceTechnique.sendFakeMail(recipient, subject, body);
99          //serviceTechnique.sendRealMail(recipient, subject, body);
100
```

```
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,
                            motDePasse) != null)
118                         {
119                             result = true;
120                         }
121                     } catch(Throwable e) {
122 
123                     }
124                     JpaUtil.fermerEntityManager();
125                 }
126             }
127 
128             return result;
129         }
130 
131 
132          public Client updateClient(Client client) {
133             JpaUtil.creerEntityManager();
134 
135             try {
136                 JpaUtil.ouvrirTransaction();
137                 client = clientDao.update(client);
138                 JpaUtil.validerTransaction();
139             } catch (Exception e) {
140                 System.out.println(e);
141                 client = null;
142                 JpaUtil.annulerTransaction();
143             } catch (Throwable ex) {
144 
                    Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
145                 client =  null;
146                 JpaUtil.annulerTransaction();
147             }
148 
149             JpaUtil.fermerEntityManager();
150 
151            return client;
```

```
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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
165             }
166
167             JpaUtil.fermerEntityManager();
168
169             return client;
170         }
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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
184             }
185
186             JpaUtil.fermerEntityManager();
187
188             return client;
189         }
190
191         public Client findClientByMail(String mail) {
192             JpaUtil.creerEntityManager();
193
194             Client client = null;
195
196             try {
197                 client = clientDao.findByMail(mail);
198             } catch (Exception e) {
199                 System.out.println(e);
200             } catch (Throwable ex) {
```

```
201                                                                    ⤶
                Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE ⤶
                VERE, 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.SE ⤶
                VERE, 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.SE ⤶
                VERE, 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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
256             }
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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
274             }
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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
292             }
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 =
                        livreurDao.findAllDisponible(commande.getPoidsEnGrammes(
                        ));
309                     livreur =
                        serviceTechnique.chooseBestLivreur(commande,livreurs);
310                     if(livreur != null){
311                         commande.setDateDebut(new Date());
312                         commande.setLivreur(livreur);
313                         commandeDao.create(commande);
314                         livreur.addCommande(commande);
315                         livreur.setDisponible(false);
316                         livreurDao.update(livreur);
317 
318                         //a commenté pour pouvoir l'utiliser en ihm non
                            console (sert juste au test)
319                         Scanner sc = new Scanner(System.in);
320                         System.out.println("Press enter...");
321                         sc.nextLine();
322                         //_____________________________________
323 
324                         JpaUtil.validerTransaction();
325                         result = true;
326                     } else {
327                         JpaUtil.annulerTransaction();
328                     }
329                 } catch (Exception e) {
330                     System.out.println(e);
331                     JpaUtil.annulerTransaction();
332                 } catch (Throwable ex) {
333 
                        Logger.getLogger(ServiceMetier.class.getName()).log(Leve
                        l.SEVERE, null, ex);
334                     JpaUtil.annulerTransaction();
335                 }
336                 nbTentative++;
337             }
338             JpaUtil.fermerEntityManager();
339 
340             // envoi du mail
341             String recipient = null;
342             String subject = null;
343             String body = null;
344             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 "+ ((LlivreurVelo)livreur).getPrenom()+",\n"
350                     + "\n       Merci d'effectuer cette livraison dès
                    maintenant, "
351                         + "tout en respectant le code de la route ;-)"
352                     + "\n\nLe Chef"
353                     + "\n\nDétails de la Livraison"
354                     + "\n   - Date/heure : " +
                    commande.getDateDebut().toString()
355                     + "\n   - Livreur : " +
                    ((LivreurVelo)livreur).getPrenom() + " "
356                         + ((LivreurVelo)livreur).getNom().toUpperCase() +
                        " (n°"
357                         + livreur.getId() + ")"
358                     + "\n   - Restaurant : " +
                    commande.getRestaurant().getDenomination()
359                     + "\n   - Client :"
360                     + "\n       " + commande.getClient().getPrenom() + " "
361                         + commande.getClient().getNom().toUpperCase()
362                     + "\n       " + commande.getClient().getAdresse()
363                     + "\n       " + commande.getClient().getTelephone()
364                     + "\n\n Commande :";
365
366                 for (LigneDeCommande ligne :
                    commande.getLignesDeCommande()) {
367                     body += "\n    - " + ligne.getQuantite() + " "
368                             + ligne.getProduit().getDenomination() + " "
369                             + ligne.getQuantite() + " x " +
                            ligne.getPrixUnitaire()
370                             + "€";
371                 }
372                 body += "\n\nTOTAL : " + commande.getPrix() + "€";
373                 serviceTechnique.sendFakeMail(recipient, subject, body);
374                 //serviceTechnique.sendRealMail(recipient, subject, body);
375             }
376             recipient = commande.getClient().getMail();
377             if(result){
378                 //envoyer mail succes
379                 subject = "Commande n°"+commande.getId();
380                 body = "Bonjour " + commande.getClient().getPrenom() + ",\n"
381                     + "\n"
382                     + "       Nous vous confirmons votre commande n°" +
                    commande.getId()
383                         + " d'un montant de " + commande.getPrix() + "€.";
384             } else {
385                 subject = "Echec commande";
386                 body = "Bonjour " + commande.getClient().getPrenom() + ",\n"
387                     + "\n"
388                     + "       La validation de votre commade a échouée,
                    veuillez "
389                         + "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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
417                 commande = null;
418                 JpaUtil.annulerTransaction();
419             }
420
421             JpaUtil.fermerEntityManager();
422
423             return commande;
424         }
425
426         public List<Commande> findCommandesByLivreurId(Long id) {
427             JpaUtil.creerEntityManager();
428
429             List<Commande> commandes = null;
430
431             try {
432                 commandes = commandeDao.findByLivreurID(id);
433             } catch (Exception e) {
434                 System.out.println(e);
435             } catch (Throwable ex) {
436                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
437             }
438
439             JpaUtil.fermerEntityManager();
```

```
440
441             return commandes;
442         }
443
444         public Commande findNotEndedByLivreur(Livreur livreur) {
445             JpaUtil.creerEntityManager();
446
447             Commande commande = null;
448
449             try {
450                 commande = commandeDao.findNotEndedByLivreur(livreur);
451             } catch (Exception e) {
452                 System.out.println(e);
453             } catch (Throwable ex) {
454                                                                                 ⏎
                    Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE ⏎
                    VERE, 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                                                                                 ⏎
                    Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE ⏎
                    VERE, null, ex);
472             }
473
474             JpaUtil.fermerEntityManager();
475
476             return commandes;
477         }
478         public Commande findCommandeById(Long id) {
479             JpaUtil.creerEntityManager();
480
481             Commande commande = null;
482
483             try {
484                 commande = commandeDao.findById(id);
485             } catch (Exception e) {
486                 System.out.println(e);
487             } catch (Throwable ex) {
488                                                                                 ⏎
                    Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE ⏎
```

```
                VERE, null, ex);
            }

            JpaUtil.fermerEntityManager();

            return commande;
        }

        public List<Commande> findAllCommandesNotEnded() {
            JpaUtil.creerEntityManager();

            List<Commande> commandes = null;

            try {
                commandes = commandeDao.findAllNotEnded();
            } catch (Exception e) {
                System.out.println(e);
            } catch (Throwable ex) {
                Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                VERE, null, ex);
            }

            JpaUtil.fermerEntityManager();

            return commandes;
        }


        public Livreur updateLivreur(Livreur livreur) {
            JpaUtil.creerEntityManager();

            try {
                JpaUtil.ouvrirTransaction();
                livreur = livreurDao.update(livreur);
                JpaUtil.validerTransaction();
            } catch (Exception e) {
                System.out.println(e);
                livreur = null;
                JpaUtil.annulerTransaction();
            } catch (Throwable ex) {
                Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                VERE, null, ex);
                livreur = null;
                JpaUtil.annulerTransaction();
            }

            JpaUtil.fermerEntityManager();

            return livreur;
        }


```

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

```
630             JpaUtil.creerEntityManager();
631
632             List<LivreurVelo> livreurs = null;
633
634             try {
635                 livreurs = livreurVeloDao.findAllLivreursVelos();
636             } catch (Exception e) {
637                 System.out.println(e);
638             } catch (Throwable ex) {
639                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
640             }
641
642             JpaUtil.fermerEntityManager();
643
644             return livreurs;
645         }
646
647         public List<Livreur> findAllLivreursDisponibles() {
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                 Logger.getLogger(ServiceMetier.class.getName()).log(Level.SE
                    VERE, null, ex);
658             }
659
660             JpaUtil.fermerEntityManager();
661
662             return livreurs;
663         }
664
665
666     }
667
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package fr.insalyon.dasi.gustatif.metier.service;

import fr.insalyon.dasi.gustatif.metier.modele.Commande;
import fr.insalyon.dasi.gustatif.metier.modele.Livreur;
import fr.insalyon.dasi.gustatif.metier.modele.LivreurDrone;
import fr.insalyon.dasi.gustatif.metier.modele.LivreurVelo;
import fr.insalyon.dasi.gustatif.util.GeoTest;
import fr.insalyon.dasi.gustatif.util.MailSender;
import java.util.List;

/**
 *
 * @author Nico
 */
public class ServiceTechnique {

    public boolean sendRealMail(String recipientMail, String subject, String body) {
        MailSender ms = new MailSender("gustatif.b3125@gmail.com","B3125-2016");
        return ms.sendMail(recipientMail,subject,body);
    }

    public void sendFakeMail(String recipientMail, String subject, String body) {
        System.out.println("");
        System.out.println("********* Mail envoyé **********");
        System.out.println("From : gustatif.b3125@gmail.com");
        System.out.println("To : " + recipientMail);
        System.out.println("Subject : " + subject);
        System.out.println(body);
        System.out.println("********************************");
        System.out.println("");
    }


    public Livreur chooseBestLivreur(Commande commande,List<Livreur> livreurs )
            throws Throwable {
        Livreur livreurChoisi = null;
        double temps;
        double tempsFinal;

        if(!livreurs.isEmpty()) {
            livreurChoisi = livreurs.get(0);
            if(livreurChoisi instanceof LivreurVelo){
                tempsFinal = GeoTest.getTripDurationByBicycleInMinute(
```

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