

dao.PersonneDAO

```

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 dao;
7
8  import java.util.List;
9  import javax.persistence.EntityManager;
10 import javax.persistence.Query;
11 import metier.modele.Personne;
12
13 /**
14  *
15  * @author ffonteneau
16  */
17 public class PersonneDAO {
18     public static Personne authentication(String mail, String mdp) {
19         String jpql = "select p from Personne p where p.mail=:email and p.mdp=:motdepasse";
20         EntityManager em = JpaUtil.obtenirEntityManager();
21         Query q = em.createQuery(jpql);
22         q.setParameter("email", mail);
23         q.setParameter("motdepasse", mdp);
24         List<Personne> res = (List<Personne>) q.getResultList();
25         return res.get(0);
26     }
27 }

```

dao.InterventionDAO

```

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 dao;
7
8  import java.util.Date;
9  import java.util.List;
10 import javax.persistence.EntityManager;
11 import javax.persistence.Query;
12 import metier.modele.Employe;
13 import metier.modele.Intervention;
14
15 /**
16 *
17 * @author ffonteneau
18 */
19 public class InterventionDAO {
20
21     public static void demandeInterv(Intervention i) {
22         EntityManager em = JpaUtil.obtenirEntityManager();
23         em.persist(i);
24     }
25
26     public static void cloturerIntervention(Intervention intervention){
27         EntityManager em = JpaUtil.obtenirEntityManager();
28         em.merge(intervention);
29     }
30
31     public static Intervention interventionCourante(Employe emp){
32         String jpql = "select i from Intervention i where i.employe=:emp and i.statut=:encours";
33         EntityManager em = JpaUtil.obtenirEntityManager();
34         Query q = em.createQuery(jpql);
35         q.setParameter("emp", emp);
36         q.setParameter("encours", "ENCOURS");
37         List<Intervention> res = (List<Intervention>) q.getResultList();
38         return res.get(0);
39     }
40
41     public static List<Intervention> historiqueJournee(){
42         String jpql = "select i from Intervention i where i.statut=:encours or i.dateFin>=:date";
43         EntityManager em = JpaUtil.obtenirEntityManager();
44         Query q = em.createQuery(jpql);
45         Date d = new Date();
46         d.setMinutes(0);
47         d.setHours(0);
48         q.setParameter("date", d);
49         q.setParameter("encours", "ENCOURS");
50         List<Intervention> res = (List<Intervention>) q.getResultList();
51         return res;
52     }
53 }

```

dao.ClientDAO

```

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 dao;
7
8  import java.util.List;
9  import javax.persistence.EntityManager;
10 import javax.persistence.Query;
11 import metier.modele.Client;
12 import metier.modele.Intervention;
13
14 /**
15  *
16  * @author ffonteneau
17  */
18 public class ClientDAO {
19
20     public static void inscrire(Client c) {
21         EntityManager em = JpaUtil.obtenirEntityManager();
22         em.persist(c);
23     }
24
25     public static List<Intervention> historique(Client c){
26         String jpql = "select i from Intervention i where i.client.id=:id";
27         EntityManager em = JpaUtil.obtenirEntityManager();
28         Query q = em.createQuery(jpql);
29         q.setParameter("id", c.getId());
30         List<Intervention> res = (List<Intervention>) q.getResultList();
31         return res;
32     }
33
34 }

```

dao.EmployeeDAO

```

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 dao;
7
8  import java.util.Date;
9  import java.util.List;
10 import javax.persistence.EntityManager;
11 import javax.persistence.Query;
12 import metier.modele.Employee;
13 import metier.modele.Intervention;
14
15 /**
16 *
17 * @author ffonteneau
18 */
19 public class EmployeeDAO {
20
21     public static void initBase() {
22         Date d = new Date(1998, 9, 2);
23         Employee employee = new Employee("Quentin", "Ferro", d, "homme", "0668629985", "vladimir.
24             poutine@insa-lyon.fr", "20 Avenue Albert Einstein, Lyon", true, 9, 22, "mdp");
25         Employee employee2 = new Employee("Zinedine", "Zidane", d, "homme", "0606363682", "zinedine.
26             superstar@insa-lyon.fr", "Avenue de Valescure, Saint-Raphaël", true, 9, 22, "mdp");
27         JpaUtil.creerEntityManager();
28         JpaUtil.ouvrirTransaction();
29         try{
30             EntityManager em = JpaUtil.obtenirEntityManager();
31             em.persist(employee);
32             em.persist(employee2);
33             JpaUtil.validerTransaction();
34         } catch (Exception ex){
35             JpaUtil.annulerTransaction();
36             ex.printStackTrace(); // ajouter l'envoi du mail
37         }
38         JpaUtil.fermerEntityManager();
39     }
40
41     public static List<Employee> employeeDispo(){
42         String jpql = "select e from Employee e where e.statut=1";
43         EntityManager em = JpaUtil.obtenirEntityManager();
44         Query q = em.createQuery(jpql);
45         List<Employee> res = (List<Employee>) q.getResultList();
46         return res;
47     }
48
49     public static void miseAJourEmployee(Employee emp){
50         EntityManager em = JpaUtil.obtenirEntityManager();
51         em.merge(emp);
52     }
53 }

```

dao.JpaUtil

```

1 package dao;
2
3 import javax.persistence.EntityManager;
4 import javax.persistence.EntityManagerFactory;
5 import javax.persistence.Persistence;
6 import javax.persistence.RollbackException;
7
8 /**
9  * Cette classe fournit des méthodes statiques utiles pour accéder aux
10  * fonctionnalités de JPA (Entity Manager, Entity Transaction). Le nom de
11  * l'unité de persistance (PERSISTENCE_UNIT_NAME) doit être conforme à la
12  * configuration indiquée dans le fichier persistence.xml du projet.
13  *
14  * @author DASI Team
15  */
16 public class JpaUtil {
17
18     // *****
19     // * TODO: IMPORTANT -- Adapter le nom de l'Unité de Persistance (cf. persistence.xml) *
20     // *****
21     /**
22      * Nom de l'unité de persistance utilisée par la Factory de Entity Manager.
23      * <br><strong>Vérifier le nom de l'unité de persistance
24      * (cf.&nbsp;persistence.xml)</strong>
25      */
26     public static final String PERSISTENCE_UNIT_NAME = "UP";
27     /**
28      * Factory de Entity Manager liée à l'unité de persistance.
29      * <br><strong>Vérifier le nom de l'unité de persistance indiquée dans
30      * l'attribut statique PERSISTENCE_UNIT_NAME
31      * (cf.&nbsp;persistence.xml)</strong>
32      */
33     private static EntityManagerFactory entityManagerFactory = null;
34     /**
35      * Gère les instances courantes de Entity Manager liées aux Threads.
36      * L'utilisation de ThreadLocal garantie une unique instance courante par
37      * Thread.
38      */
39     private static final ThreadLocal<EntityManager> threadLocalEntityManager = new ThreadLocal<
40         EntityManager>() {
41
42         @Override
43         protected EntityManager initialValue() {
44             return null;
45         }
46     };
47
48     // Méthode pour avoir des messages de Log dans le bon ordre (pause)
49     private static void pause(long milliseconds) {
50         try {
51             Thread.sleep(milliseconds);
52         } catch (InterruptedException ex) {
53             ex.hashCode();
54         }
55     }
56
57     // Méthode pour avoir des messages de Log dans le bon ordre (log)
58     private static void log(String message) {
59         System.out.flush();
60         pause(5);
61         System.err.println("[JpaUtil:Log]" + message);
62         System.err.flush();
63         pause(5);
64     }
65
66     /**
67      * Initialise la Factory de Entity Manager.
68      * <br><strong>À utiliser uniquement au début de la méthode main() [projet
69      * Java Application] ou dans la méthode init() de la Servlet Contrôleur
70      * (ActionServlet) [projet Web Application].</strong>
71      */
72     public static synchronized void init() {
73         log("Initialisation de la factory de contexte de persistance");
74         if (entityManagerFactory != null) {
75             entityManagerFactory.close();
76         }
77     }
78 }

```

```

75     }
76     entityManagerFactory = Persistence.createEntityManagerFactory(PERSISTENCE_UNIT_NAME);
77 }
78
79 /**
80  * Libère la Factory de Entity Manager.
81  * <br><strong>À utiliser uniquement à la fin de la méthode main() [projet
82  * Java Application] ou dans la méthode destroy() de la Servlet Contrôleur
83  * (ActionServlet) [projet Web Application].</strong>
84  */
85 public synchronized void destroy() {
86     log("Libération de la factory de contexte de persistance");
87     if (entityManagerFactory != null) {
88         entityManagerFactory.close();
89         entityManagerFactory = null;
90     }
91 }
92
93 /**
94  * Crée l'instance courante de Entity Manager (liée à ce Thread).
95  * <br><strong>À utiliser uniquement au niveau Service.</strong>
96  */
97 public static void creerEntityManager() {
98     log("Création du contexte de persistance");
99     threadLocalEntityManager.set(entityManagerFactory.createEntityManager());
100 }
101
102 /**
103  * Ferme l'instance courante de Entity Manager (liée à ce Thread).
104  * <br><strong>À utiliser uniquement au niveau Service.</strong>
105  */
106 public static void fermerEntityManager() {
107     log("Fermeture du contexte de persistance");
108     EntityManager em = threadLocalEntityManager.get();
109     em.close();
110     threadLocalEntityManager.set(null);
111 }
112
113 /**
114  * Démarre une transaction sur l'instance courante de Entity Manager.
115  * <br><strong>À utiliser uniquement au niveau Service.</strong>
116  */
117 public static void ouvrirTransaction() {
118     log("Ouverture de la transaction (begin)");
119     try {
120         EntityManager em = threadLocalEntityManager.get();
121         em.getTransaction().begin();
122     } catch (Exception ex) {
123         log("Erreur lors de l'ouverture de la transaction");
124         throw ex;
125     }
126 }
127
128 /**
129  * Valide la transaction courante sur l'instance courante de Entity Manager.
130  * <br><strong>À utiliser uniquement au niveau Service.</strong>
131  *
132  * @exception RollbackException lorsque le <em>commit</em> n'a pas réussi.
133  */
134 public static void validerTransaction() throws RollbackException {
135     log("Validation de la transaction (commit)");
136     try {
137         EntityManager em = threadLocalEntityManager.get();
138         em.getTransaction().commit();
139     } catch (Exception ex) {
140         log("Erreur lors de la validation (commit) de la transaction");
141         throw ex;
142     }
143 }
144
145 /**
146  * Annule la transaction courante sur l'instance courante de Entity Manager.
147  * Si la transaction courante n'est pas démarrée, cette méthode n'effectue
148  * aucune opération.
149  * <br><strong>À utiliser uniquement au niveau Service.</strong>
150  */
151 public static void annulerTransaction() {

```

```

152     try {
153         log("Annulation de la transaction (rollback)");
154
155         EntityManager em = threadLocalEntityManager.get();
156         if (em.getTransaction().isActive()) {
157             log("Annulation effective de la transaction (rollback d'une transaction active)");
158             em.getTransaction().rollback();
159         }
160
161     } catch (Exception ex) {
162         log("Erreur lors de l'annulation (rollback) de la transaction");
163         throw ex;
164     }
165 }
166
167 /**
168  * Retourne l'instance courante de Entity Manager.
169  * <br><strong>À utiliser uniquement au niveau DAO.</strong>
170  *
171  * @return instance de Entity Manager
172  */
173 protected static EntityManager obtenirEntityManager() {
174     log("Obtention du contexte de persistance");
175     return threadLocalEntityManager.get();
176 }
177 }

```

util.Saisie

```

1 package util;
2
3 import java.io.BufferedReader;
4 import java.io.IOException;
5 import java.io.InputStreamReader;
6 import java.util.Arrays;
7 import java.util.List;
8
9 /**
10  *
11  * @author DASI Team
12  */
13 public class Saisie {
14
15     public static String lireChaine(String invite) {
16         String chaineLue = null;
17         System.out.print(invite);
18         try {
19             InputStreamReader isr = new InputStreamReader(System.in);
20             BufferedReader br = new BufferedReader(isr);
21             chaineLue = br.readLine();
22         } catch (IOException ex) {
23             ex.printStackTrace(System.err);
24         }
25         return chaineLue;
26     }
27
28     public static Integer lireInteger(String invite) {
29         Integer valeurLue = null;
30         while (valeurLue == null) {
31             try {
32                 valeurLue = Integer.parseInt(lireChaine(invite));
33             } catch (NumberFormatException ex) {
34                 System.out.println("/!\ Erreur de saisie - Nombre entier attendu /!\");
35             }
36         }
37         return valeurLue;
38     }
39
40     public static Integer lireInteger(String invite, List<Integer> valeursPossibles) {
41         Integer valeurLue = null;
42         while (valeurLue == null) {
43             try {
44                 valeurLue = Integer.parseInt(lireChaine(invite));
45             } catch (NumberFormatException ex) {
46                 System.out.println("/!\ Erreur de saisie - Nombre entier attendu /!\");
47             }
48             if (!(valeursPossibles.contains(valeurLue))) {
49                 System.out.println("/!\ Erreur de saisie - Valeur non-autorisée /!\");
50                 valeurLue = null;
51             }
52         }
53         return valeurLue;
54     }
55
56     public static void pause() {
57         lireChaine("--PAUSE--");
58     }
59
60     public static void main(String[] args) {
61
62         System.out.println("Bonjour!");
63
64         String nom = Saisie.lireChaine("Entrez votre nom:");
65         System.out.println("Bonjour, " + nom + "!");
66
67         Integer age = Saisie.lireInteger("Entrez votre âge:");
68         System.out.println("Vous avez " + age + " ans.");
69
70         Integer annee = Saisie.lireInteger("Entrez votre année au Département IF(3,4,5):", Arrays.
71             asList(3,4,5));
72         System.out.println("Vous êtes en " + annee + " IF.");
73
74         Saisie.pause();
75     }
76 }

```



```
75  
76     System.out.println("Au revoir!");  
77 }  
78 }
```

util.Message

```

1 package util;
2
3 import java.io.PrintStream;
4 import java.io.PrintWriter;
5 import java.io.StringWriter;
6 import java.text.SimpleDateFormat;
7 import java.util.Date;
8
9 /**
10  *
11  * @author DASI Team
12  */
13 public class Message {
14
15     private final static PrintStream OUT = System.out;
16     private final static SimpleDateFormat TIMESTAMP_FORMAT = new SimpleDateFormat("yyyy-MM-dd~HH:mm:ss");
17     private final static SimpleDateFormat HORODATE_FORMAT = new SimpleDateFormat("dd/MM/yyyy~HH:mm:ss");
18
19     private static void debut() {
20         Date maintenant = new Date();
21         OUT.println();
22         OUT.println();
23         OUT.println("---<([MESSAGE@ + TIMESTAMP_FORMAT.format(maintenant) + "])>---");
24         OUT.println();
25     }
26
27     private static void fin() {
28         OUT.println();
29         OUT.println("---<([FINDUMESSAGE])>---");
30         OUT.println();
31         OUT.println();
32     }
33
34     public static void envoyerMail(String mailExpediteur, String mailDestinataire, String objet, String
        corps) {
35
36         Date maintenant = new Date();
37         Message.debut();
38         OUT.println("~~~E-mail~envoyé~" + HORODATE_FORMAT.format(maintenant) + "~~~");
39         OUT.println("De: " + mailExpediteur);
40         OUT.println("À: " + mailDestinataire);
41         OUT.println("Obj: " + objet);
42         OUT.println();
43         OUT.println(corps);
44         Message.fin();
45     }
46
47     public static void envoyerNotification(String telephoneDestinataire, String message) {
48
49         Date maintenant = new Date();
50         Message.debut();
51         OUT.println("~~~Notification~envoyée~" + HORODATE_FORMAT.format(maintenant) + "~~~");
52         OUT.println("À: " + telephoneDestinataire);
53         OUT.println();
54         OUT.println(message);
55         Message.fin();
56     }
57
58     public static void main(String[] args) {
59
60         //DebugLogger.log("Début des Tests...");
61
62         StringWriter corps = new StringWriter();
63         PrintWriter mailWriter = new PrintWriter(corps);
64
65         mailWriter.println("Bonjour,");
66         mailWriter.println();
67         mailWriter.println("Ceci est un mail destiné à tester l'envoi simulé par affichage sur la
            console.");
68         mailWriter.println();
69         mailWriter.println("Cordialement,");
70         mailWriter.println();
71         mailWriter.println("Yann Gripay");
72

```

```

73     Message.envoyerMail(
74         "yann.gripay@insa-lyon.fr",
75         "etudiants.3IF@insa-lyon.fr",
76         "[DASI]_Test_d'envoi_de_e-mail",
77         corps.toString()
78     );
79
80
81     StringWriter message = new StringWriter();
82     PrintWriter notificationWriter = new PrintWriter(message);
83
84     notificationWriter.println("Ceci_est_une_notification_pour_prévenir_de_2_choses:");
85     notificationWriter.println("1)_NE_PAS_oublier_le_poly");
86     notificationWriter.println("2)_TESTER_au_fur_et_à_mesure_du_développement");
87
88     Message.envoyerNotification(
89         "0988776655",
90         message.toString()
91     );
92
93     //DebugLogger.log("Fin des Tests...");
94
95 }
96 }

```

util.DebugLogger

```

1 package util;
2
3 /**
4  *
5  * @author DASI Team
6  */
7 public class DebugLogger {
8
9     // Méthode pour avoir des messages de Log dans le bon ordre (pause)
10    public static void pause(long milliseconds) {
11        try {
12            Thread.sleep(milliseconds);
13        } catch (InterruptedException ex) {
14            ex.hashCode();
15        }
16    }
17
18    // Méthode pour avoir des messages de Log dans le bon ordre (log)
19    public static void log(String message) {
20        System.out.flush();
21        pause(5);
22        System.err.println("[DebugLogger]_" + message);
23        System.err.flush();
24        pause(5);
25    }
26
27    // Méthode pour avoir des messages de Log dans le bon ordre (log avec exception)
28    public static void log(String message, Exception ex) {
29        System.out.flush();
30        pause(5);
31        System.err.println("[DebugLogger]_" + message);
32        System.err.println("**EXCEPTION**_" + ex.getMessage());
33        System.err.print("**EXCEPTION**_>>>_");
34        ex.printStackTrace(System.err);
35        System.err.flush();
36        pause(5);
37    }
38
39    public static void main(String[] args) {
40
41        DebugLogger.log("**_Début_du_Test_**");
42
43        DebugLogger.log("Message_de_DEBUG_pour_tester...");
44
45        Integer a = 0;
46        Integer b = null;
47
48        if (a < 0) {
49            DebugLogger.log("ERREUR_sur_la_valeur_de_A");
50        }
51
52        try {
53            if (b > 0) {
54                DebugLogger.log("Test_OK_pour_la_valeur_de_B");
55            }
56        } catch (Exception ex) {
57            DebugLogger.log("Problème_avec_B", ex);
58        }
59
60        DebugLogger.log("**_Fin_du_Test_**");
61    }
62 }

```

util.GeoTest

```

1 package util;
2
3 import com.google.maps.DirectionsApi;
4 import com.google.maps.DirectionsApiRequest;
5 import com.google.maps.GeoApiContext;
6 import com.google.maps.GeocodingApi;
7 import com.google.maps.model.DirectionsResult;
8 import com.google.maps.model.DirectionsRoute;
9 import com.google.maps.model.GeocodingResult;
10 import com.google.maps.model.LatLng;
11 import com.google.maps.model.TravelMode;
12
13 /**
14  *
15  * @author DASI Team
16  */
17 /* DÉPENDANCES Maven:
18 <dependency>
19   <groupId>com.google.maps</groupId>
20   <artifactId>google-maps-services</artifactId>
21   <version>0.2.11</version>
22 </dependency>
23 <dependency>
24   <groupId>org.slf4j</groupId>
25   <artifactId>slf4j-nop</artifactId>
26   <version>1.7.26</version>
27 </dependency>
28 */
29 public class GeoTest {
30
31     final static String MA_CLE_GOOGLE_API = "AIzaSyAhf3JleYpal9S-xouJYH8lf7Dvz5Y2Nko";
32
33     final static GeoApiContext MON_CONTEXTE_GEOAPI = new GeoApiContext.Builder().apiKey(MA_CLE_GOOGLE_API)
34         .build();
35
36     public static LatLng getLatLng(String adresse) {
37         try {
38             GeocodingResult[] results = GeocodingApi.geocode(MON_CONTEXTE_GEOAPI, adresse).await();
39
40             return results[0].geometry.location;
41
42         } catch (Exception ex) {
43             return null;
44         }
45     }
46
47     public static double toRad(double angleInDegree) {
48         return angleInDegree * Math.PI / 180.0;
49     }
50
51     public static double getFlightDistanceInKm(LatLng origin, LatLng destination) {
52         // From: http://www.movable-type.co.uk/scripts/latlong.html
53         double R = 6371.0; // Average radius of Earth (km)
54         double dLat = toRad(destination.lat - origin.lat);
55         double dLon = toRad(destination.lng - origin.lng);
56         double lat1 = toRad(origin.lat);
57         double lat2 = toRad(destination.lat);
58
59         double a = Math.sin(dLat / 2.0) * Math.sin(dLat / 2.0)
60             + Math.sin(dLon / 2.0) * Math.sin(dLon / 2.0) * Math.cos(lat1) * Math.cos(lat2);
61         double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1.0 - a));
62         double d = R * c;
63
64         return Math.round(d * 1000.0) / 1000.0;
65     }
66
67     public static Double getTripDurationByBicycleInMinute(LatLng origin, LatLng destination, LatLng...
68         steps) {
69         return getTripDurationOrDistance(TravelMode.BICYCLING, true, origin, destination, steps);
70     }
71
72     public static Double getTripDistanceByCarInKm(LatLng origin, LatLng destination, LatLng... steps) {
73         return getTripDurationOrDistance(TravelMode.DRIVING, false, origin, destination, steps);
74     }
75 }

```

```

74
75 public static Double getTripDurationOrDistance(TravelMode mode, boolean duration, LatLng origin,
76     LatLng destination, LatLng... steps) {
77     DirectionsApiRequest request = DirectionsApi.getDirections(MON_CONTEXTE_GEOAPI, origin.toString()
78         , destination.toString());
79     request.mode(mode);
80     request.region("fr");
81     if (steps.length > 0) {
82
83         String[] stringSteps = new String[steps.length];
84         for (int i = 0; i < steps.length; i++) {
85             stringSteps[i] = steps[i].toString();
86         }
87
88         request.waypoints(stringSteps);
89     }
90
91     double cumulDistance = 0.0;
92     double cumulDuration = 0.0;
93
94     try {
95         DirectionsResult result = request.await();
96         DirectionsRoute[] directions = result.routes;
97
98         for (int legIndex = 0; legIndex < directions[0].legs.length; legIndex++) {
99
100             cumulDistance += directions[0].legs[legIndex].distance.inMeters / 1000.0;
101             cumulDuration += Math.ceil(directions[0].legs[legIndex].duration.inSeconds / 60.0);
102         }
103
104     } catch (Exception ex) {
105         return null;
106     }
107
108     if (duration) {
109         return cumulDuration;
110     } else {
111         return cumulDistance;
112     }
113 }
114
115 public static void main(String[] args) {
116
117     if (MA_CLE_GOOGLE_API.equals("XXXXXXX-Moodle-Clé")) {
118         for (int i=0; i<100; i++) {
119             System.err.println("[ERREUR] VOUS AVEZ OUBLIÉ DE CHANGER LA CLÉ DE L'API!!!!");
120         }
121         System.exit(-1);
122     }
123
124     String adresse1 = "7 Avenue Jean Capelle Ouest, Villeurbanne";
125     LatLng coords1 = getLatLng(adresse1);
126     System.out.println("Lat/Lng de Adresse #1: " + coords1);
127
128     String adresse2 = "37 Avenue Jean Capelle Est, Villeurbanne";
129     LatLng coords2 = getLatLng(adresse2);
130     System.out.println("Lat/Lng de Adresse #2: " + coords2);
131
132     String adresse3 = "61 Avenue Roger Salengro, Villeurbanne";
133     LatLng coords3 = getLatLng(adresse3);
134     System.out.println("Lat/Lng de Adresse #3: " + coords3);
135
136     // Coordonnées directes: Rond-Point du Totem, Cours Tolstoï, Villeurbanne
137     LatLng coords4 = new LatLng(45.763781, 4.8735128);
138     System.out.println("Lat/Lng de Coords #4: " + coords4.lat + "; " + coords4.lng + "");
139
140
141     Double duree = getTripDurationByBicycleInMinute(coords1, coords3);
142     System.out.println("Durée de Trajet à Vélo de Adresse #1 à Adresse #3 (trajet direct): " + duree
143         + " min");
144
145     Double distance = getTripDistanceByCarInKm(coords1, coords3, coords2);
146     System.out.println("Distance en Voiture de Adresse #1 à Adresse #3 en passant par Adresse #2 (
147         distance par la route): " + distance + " km");

```

```
147     Double distanceVolDOiseau = getFlightDistanceInKm(coords1, coords3);
148     System.out.println("Distance à Vol d'Oiseau de Adresse #1 à Adresse #3 (distance géographique): "
149         + distanceVolDOiseau + " km");
149
150     Double autreDistanceVolDOiseau = getFlightDistanceInKm(coords1, coords4);
151     System.out.println("Distance à Vol d'Oiseau de Adresse #1 à Coords #4 (distance géographique): "
152         + autreDistanceVolDOiseau + " km");
152 }
153 }
```

metier.modele.Personne

```

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 metier.modele;
7
8  import java.io.Serializable;
9  import java.util.Date;
10 import javax.persistence.Entity;
11 import javax.persistence.GeneratedValue;
12 import javax.persistence.GenerationType;
13 import javax.persistence.Id;
14 import javax.persistence.Inheritance;
15 import javax.persistence.InheritanceType;
16 import javax.persistence.Temporal;
17 import javax.persistence.TemporalType;
18
19 /**
20 *
21 * @author ffonteneau
22 */
23 @Entity
24 @Inheritance(strategy = InheritanceType.JOINED)
25 public abstract class Personne implements Serializable {
26     @Id
27     @GeneratedValue(strategy = GenerationType.AUTO)
28     protected Integer id;
29     protected String nom;
30     protected String prenom;
31     @Temporal(TemporalType.DATE) protected Date dateNaissance;
32     protected String civilite;
33     protected String numTel;
34     protected String mail;
35     protected String adresse;
36     protected String mdp;
37
38     public Personne(){}
39
40     public Personne(String nom, String prenom, Date date, String civilite,
41                     String numTel, String mail, String adresse, String mdp) {
42         this.nom = nom;
43         this.prenom = prenom;
44         this.dateNaissance = date;
45         this.civilite = civilite;
46         this.numTel = numTel;
47         this.mail = mail;
48         this.adresse = adresse;
49         this.mdp = mdp;
50     }
51
52     public String getNom() {
53         return nom;
54     }
55
56     public String getPrenom() {
57         return prenom;
58     }
59
60     public Date getDateNaissance() {
61         return dateNaissance;
62     }
63
64     public String getCivilite() {
65         return civilite;
66     }
67
68     public String getNumTel() {
69         return numTel;
70     }
71
72     public String getMail() {
73         return mail;
74     }
75

```



```
76     public String getAdresse() {
77         return adresse;
78     }
79
80     public Integer getId() {
81         return id;
82     }
83
84     public String getMdp() {
85         return mdp;
86     }
87
88     public void setDateNaissance(Date dateNaissance) {
89         this.dateNaissance = dateNaissance;
90     }
91
92     public void setCivilite(String civilite) {
93         this.civilite = civilite;
94     }
95
96     public void setNom(String nom) {
97         this.nom = nom;
98     }
99
100    public void setPrenom(String prenom) {
101        this.prenom = prenom;
102    }
103 }
```

metier.modele.Intervention

```

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 metier.modele;
7
8  import java.io.Serializable;
9  import java.util.Date;
10 import javax.persistence.Entity;
11 import javax.persistence.GeneratedValue;
12 import javax.persistence.GenerationType;
13 import javax.persistence.Id;
14 import javax.persistence.Inheritance;
15 import javax.persistence.InheritanceType;
16 import javax.persistence.Temporal;
17 import javax.persistence.TemporalType;
18
19 /**
20 *
21 * @author ffonteneau
22 */
23
24 @Entity
25 @Inheritance(strategy = InheritanceType.JOINED)
26 public abstract class Intervention implements Serializable {
27     @Id
28     @GeneratedValue(strategy = GenerationType.AUTO)
29     protected Integer id;
30     protected Client client;
31     protected Employe employe;
32     @Temporal(TemporalType.DATE) protected Date dateInter;
33     @Temporal(TemporalType.DATE) protected Date dateFin;
34     protected String description;
35     protected String statut;
36     protected String commentaire;
37     public Intervention(Client client, Date dateInter,
38         String description, String statut){
39         this.client = client;
40         this.employe = employe;
41         this.dateInter = dateInter;
42         this.description = description;
43         this.statut = statut;
44     }
45
46     public Intervention() {
47     }
48
49     public Date getDateFin() {
50         return dateFin;
51     }
52
53     public void setCommentaire(String commentaire) {
54         this.commentaire = commentaire;
55     }
56
57     public String getCommentaire() {
58         return commentaire;
59     }
60
61     public String getStatut() {
62         return statut;
63     }
64
65     public void setDateFin(Date dateFin) {
66         this.dateFin = dateFin;
67     }
68
69     public void setStatut(String statut) {
70         this.statut = statut;
71     }
72
73
74     public Client getClient() {
75         return client;

```

```
76     }
77
78     public Employee getEmploye() {
79         return employe;
80     }
81
82     public Date getDateInter() {
83         return dateInter;
84     }
85
86     public String getDescription() {
87         return description;
88     }
89
90     public Integer getId() {
91         return id;
92     }
93
94     public void setClient(Client client) {
95         this.client = client;
96     }
97
98     public void setEmploye(Employee employe) {
99         this.employe = employe;
100     }
101
102     public void setDateInter(Date dateNaissance) {
103         this.dateInter = dateNaissance;
104     }
105
106     public void setDescription(String description) {
107         this.description = description;
108     }
109
110 }
```

metier.modele.Employee

```

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 metier.modele;
7
8  import java.util.Date;
9  import javax.persistence.Entity;
10 import javax.persistence.Transient;
11
12
13 /**
14 *
15 * @author ffonteneau
16 */
17
18 @Entity
19 //public class Employee extends Personne {
20 public class Employee extends Personne {
21     protected Integer numEmploye;
22     protected boolean statut;
23     protected Integer heureDebut;
24     protected Integer heureFin;
25
26     public Employee (){}
27
28     public Employee(String nom, String prenom, Date date, String civilite,
29                     String numTel, String mail, String adresse,
30                     boolean statut, Integer heureDebut, Integer heureFin, String mdp){
31         super(nom, prenom, date, civilite, numTel, mail, adresse, mdp);
32         this.statut = statut;
33         this.heureDebut = heureDebut;
34         this.heureFin = heureFin;
35         this.numEmploye = id;
36     }
37
38     public Integer getNumEmploye() {
39         return numEmploye;
40     }
41
42     public boolean isStatut() {
43         return statut;
44     }
45
46     public Integer getHeureDebut() {
47         return heureDebut;
48     }
49
50     public Integer getHeureFin() {
51         return heureFin;
52     }
53
54     public void setNumEmploye(Integer numEmploye) {
55         this.numEmploye = numEmploye;
56     }
57
58     public void setStatut(boolean statut) {
59         this.statut = statut;
60     }
61
62     public void setHeureDebut(Integer heureDebut) {
63         this.heureDebut = heureDebut;
64     }
65
66     public void setHeureFin(Integer heureFin) {
67         this.heureFin = heureFin;
68     }
69
70
71 }

```

metier.modele.InterventionAnimale

```
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 metier.modele;
7
8  import java.util.Date;
9  import javax.persistence.Entity;
10
11  /**
12   *
13   * @author ffonteneau
14   */
15  @Entity
16  public class InterventionAnimale extends Intervention {
17      protected String typeAnimal;
18
19      public InterventionAnimale(Client client, Date dateInter,
20          String description, String typeAnimal, String statut){
21          super(client, dateInter, description, statut);
22          this.typeAnimal = typeAnimal;
23      }
24
25      public InterventionAnimale(){}
26
27      public String getTypeAnimal() {
28          return typeAnimal;
29      }
30
31      public void setTypeAnimal(String typeAnimal) {
32          this.typeAnimal = typeAnimal;
33      }
34  }
```

metier.modele.InterventionIncident

```
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 metier.modele;
7
8  import java.util.Date;
9  import javax.persistence.Entity;
10
11  /**
12   *
13   * @author ffonteneau
14   */
15  @Entity
16  public class InterventionIncident extends Intervention {
17
18      public InterventionIncident(Client client, Date dateInter,
19          String description, String statut){
20          super(client, dateInter, description, statut);
21      }
22
23      public InterventionIncident() {
24      }
25
26
27  }
```

metier.modele.Client

```
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 metier.modele;
7
8  import java.util.Date;
9  import javax.persistence.Entity;
10
11  /**
12   *
13   * @author ffonteneau
14   */
15
16  @Entity
17  public class Client extends Personne {
18
19      public Client (){}
20
21      public Client(String nom, String prenom, Date date, String civilite, String numTel, String mail,
22                     String adresse, String mdp){
23          super(nom, prenom, date, civilite, numTel, mail, adresse, mdp);
24      }
```

metier.modele.InterventionLivraison

```
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 metier.modele;
7
8  import java.util.Date;
9  import javax.persistence.Entity;
10
11  /**
12   *
13   * @author ffonteneau
14   */
15  @Entity
16  public class InterventionLivraison extends Intervention {
17      protected String objet;
18      protected String entreprise;
19
20      public InterventionLivraison(String objet, String entreprise, Client client,
21          Date dateInter, String description, String statut){
22          super(client, dateInter, description, statut);
23          this.objet = objet;
24          this.entreprise = entreprise;
25      }
26
27      public InterventionLivraison() {
28      }
29
30      public String getObjet() {
31          return objet;
32      }
33
34      public String getEntreprise() {
35          return entreprise;
36      }
37
38      public void setObjet(String objet) {
39          this.objet = objet;
40      }
41
42      public void setEntreprise(String entreprise) {
43          this.entreprise = entreprise;
44      }
45  }
```


metier.service.Service

```

1 package metier.service;
2
3 import com.google.maps.model.LatLng;
4 import dao.ClientDAO;
5 import dao.EmloyeeDAO;
6 import dao.InterventionDAO;
7 import dao.JpaUtil;
8 import dao.PersonneDAO;
9 import java.io.PrintWriter;
10 import java.io.StringWriter;
11 import java.util.Date;
12 import java.util.List;
13 import metier.modele.Client;
14 import metier.modele.Employe;
15 import metier.modele.Intervention;
16 import metier.modele.Personne;
17 import util.DebugLogger;
18 import static util.GeoTest.getLatLng;
19 import static util.GeoTest.getTripDurationByBicycleInMinute;
20 import util.Message;
21
22 /**
23  *
24  * @author ffonteneau
25  */
26 public class Service {
27
28
29     public static boolean inscription(Client c) {
30         //System.out.println("[Service:Log] inscription de "+c.getNom());
31         boolean inscrit = true;
32         JpaUtil.creerEntityManager();
33         JpaUtil.ouvrirTransaction();
34         try{
35             ClientDAO.inscrire(c);
36             JpaUtil.validerTransaction();
37         } catch(Exception e){
38             inscrit = false;
39             DebugLogger.log("[Service:Log]_echec_d'inscription_de_"+c.getNom(), e);
40             JpaUtil.annulerTransaction();
41         }
42         if (inscrit){
43             //envoi du mail
44             StringWriter corps = new StringWriter();
45             PrintWriter mailWriter = new PrintWriter(corps);
46
47             mailWriter.println("Bonjour_"+c.getPrenom()+",");
48             mailWriter.println();
49             mailWriter.println("_Ceci_est_un_mail_vous_confirant_votre_inscription_à_PROACT' IF!");
50             mailWriter.println();
51             mailWriter.println("_Cordialement,");
52             mailWriter.println();
53             mailWriter.println("_L'équipe_PROACT' IF");
54
55             Message.envoyerMail(
56                 "pro@actif.fr",
57                 c.getMail(),
58                 "[PROACT' IF]_Confirmation_inscription",
59                 corps.toString()
60             );
61             //System.out.println("[Service:Log] inscription de "+c.getNom()+" terminée");
62         }
63         JpaUtil.fermerEntityManager();
64         return inscrit;
65     }
66
67     //méthode servant à rechercher un employe pour une inscription
68     //la recherche s'effectue sur
69     private static Employe rechercherEmploye(Intervention i){
70         List<Employe> listeEmploye;
71         Employe emp = null;
72         JpaUtil.creerEntityManager();
73         listeEmploye = EmloyeeDAO.employeDispo();
74         if(listeEmploye != null){
75             //dans tous les employes disponibles rechrcher le plus proche

```

```

76         emp = listeEmploye.get(0);
77         for(Employe e : listeEmploye){
78             LatLng coordsClient = getLatLng(i.getClient().getAdresse());
79             LatLng coordsCurrent = getLatLng(emp.getAdresse());
80             LatLng coordsComp = getLatLng(e.getAdresse());
81             if(getTripDurationByBicycleInMinute(coordsClient, coordsComp) <
                getTripDurationByBicycleInMinute(coordsClient, coordsCurrent)) {
82                 emp = e;
83             }
84         }
85     }
86     JpaUtil.fermerEntityManager();
87     return emp;
88 }
89
90 //prend une intervention sans employe mais avec un client,
91 //trouve l'employe correspondant
92 public static boolean demandeInter(Intervention inter) {
93     //System.out.println("[Service:Log] "+inter.getClient().getNom()+" demande intervention");
94     Employe emp = rechercherEmploye(inter);
95     boolean interventionAcceptee = true;
96     if (emp != null){
97         inter.setEmploye(emp);
98         inter.setStatut("EN_COURS");
99         emp.setStatut(false);
100
101         JpaUtil.creerEntityManager();
102         JpaUtil.ouvrirTransaction();
103
104         try{
105             InterventionDAO.demandeInterv(inter);
106             EmployeDAO.miseAJourEmploye(emp);
107             JpaUtil.validerTransaction();
108
109             //notification de l'intervention de l'employe par message
110             StringWriter message = new StringWriter();
111             PrintWriter notificationWriter = new PrintWriter(message);
112             notificationWriter.println("Bonjour,"+emp.getPrenom());
113             notificationWriter.println("Vous avez une nouvelle intervention!");
114             String typeInter = inter.getClass().getName();
115             typeInter = typeInter.substring(26);
116             notificationWriter.println("C'est une intervention de type:"+typeInter);
117             notificationWriter.println("Adresse:"+inter.getClient().getAdresse());
118             notificationWriter.println("Descriptif:"+inter.getDescription());
119             Message.envoyerNotification(
120                 emp.getNumTel(),
121                 message.toString()
122             );
123         } catch (Exception e){
124             JpaUtil.annulerTransaction();
125             interventionAcceptee = false;
126             DebugLogger.log("[Service:Log] erreur d'ajout d'intervention:"+e);
127         }
128         JpaUtil.fermerEntityManager();
129
130     } else {
131         interventionAcceptee = false;
132         DebugLogger.log("[Service:Log] Intervention refusée, aucun employe disponible");
133     }
134     return interventionAcceptee;
135 }
136
137 //Envoie pointernull si Personne non trouvée
138 public static Personne authentication(String mail, String mdp) {
139     Personne personne = null;
140     JpaUtil.creerEntityManager();
141     try {
142         personne = PersonneDAO.authentication(mail, mdp);
143         //System.out.println("[Service:Log] "+personne.getNom()+" connecté");
144     } catch (Exception e) {
145         DebugLogger.log("[Service:Log] echec de connexion de "+mail);
146         //authentication non complete
147     }
148     JpaUtil.fermerEntityManager();
149     return personne;
150 }
151 }

```

```

152
153
154 //récupère l'historique des intervention d'un client, renvoie une
155 //une liste vide ou pointeur null si aucune intervention
156 public static List<Intervention> historiqueClient(Client c) {
157     //System.out.println("[Service:Log] demande historique de "+c.getNom());
158     List<Intervention> historique = null;
159     JpaUtil.creerEntityManager();
160     try{
161         historique = ClientDAO.historique(c);
162     }catch(Exception e){
163         DebugLogger.log("[Service:Log] Erreur en cherchant l'historique du client: ",e);
164     }
165     JpaUtil.fermerEntityManager();
166     return historique;
167 }
168
169 //Renvoie l'intervention que l'employe a à effectuer,
170 //Renvoie null si pas d'intervention à faire
171 public static Intervention interventionActuelle(Employe emp){
172     Intervention intervention = null;
173     JpaUtil.creerEntityManager();
174
175     //récupération de l'intervention
176     try {
177         intervention = InterventionDAO.interventionCourante(emp);
178         //System.out.println("[Service:Log] Cloture de l'intervention");
179     } catch(Exception e) {
180         DebugLogger.log("[Service:Log] Erreur en trouvant l'intervention de l'employe: ",e);
181     }
182     JpaUtil.fermerEntityManager();
183     return intervention;
184 }
185
186 //cloture l'intervention actuelle d'un employe
187 public static boolean cloturerIntervention(Employe emp, String commentaire, String statut){
188     boolean cloturee = true;
189     Intervention intervention = interventionActuelle(emp);
190
191     //mise à jour de l'intervention et du statut de l'employe
192     if(intervention != null){
193         intervention.setStatut(statut);
194         intervention.setCommentaire(commentaire);
195         intervention.setDateFin(new Date());
196         emp.setStatut(true);
197
198         JpaUtil.creerEntityManager();
199         JpaUtil.ouvrirTransaction();
200         try{
201             InterventionDAO.cloturerIntervention(intervention);
202             EmployeDAO.miseAJourEmploye(emp);
203             //System.out.println("[Service:Log] Intervention cloturée");
204             JpaUtil.validerTransaction();
205
206             //notification du client
207             StringWriter message = new StringWriter();
208             PrintWriter notificationWriter = new PrintWriter(message);
209             notificationWriter.println("Bonjour, "+intervention.getClient().getPrenom());
210             notificationWriter.println("Votre intervention a bien été traitée,");
211             notificationWriter.println("Commentaire de l'employé: "+intervention.getCommentaire());
212             Message.envoyerNotification(
213                 intervention.getClient().getNumTel(),
214                 message.toString()
215             );
216         }catch(Exception e){
217             cloturee = false;
218             JpaUtil.annulerTransaction();
219             DebugLogger.log("[Service:Log] Erreur de cloture d'intervention",e);
220         }
221         JpaUtil.fermerEntityManager();
222     }else{
223         cloturee = false;
224         DebugLogger.log("[Service:Log] Employe n'a pas d'intervention courante");
225     }
226     return cloturee;
227 }
228

```

```
229 //envoie une liste de toutes les interventions de la journée
230 public static List<Intervention> interventionsJournee(){
231     List<Intervention> historique = null;
232     JpaUtil.creerEntityManager();
233     try{
234         historique = InterventionDAO.historiqueJournee();
235     }catch(Exception e){
236         DebugLogger.log("[Service:Log]_Erreur_en_cherchant_l'historique_du_client:",e);
237     }
238     JpaUtil.fermerEntityManager();
239     return historique;
240 }
241 }
```

vue.Main

```

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 vue;
7
8  import com.google.maps.model.LatLng;
9  import util.GeoTest;
10 import dao.EmployeDAO;
11 import dao.JpaUtil;
12 import java.io.PrintWriter;
13 import java.io.StringWriter;
14 import java.util.Date;
15 import java.util.List;
16 import javax.persistence.EntityManager;
17 import metier.modele.*;
18 import metier.service.*;
19 import util.DebugLogger;
20 import static util.GeoTest.getFlightDistanceInKm;
21 import static util.GeoTest.getLatLng;
22 import static util.GeoTest.getTripDistanceByCarInKm;
23 import static util.GeoTest.getTripDurationByBicycleInMinute;
24 import util.Message;
25
26 /**
27 *
28 * @author Quentin Ferro
29 */
30 public class Main {
31
32     public static void main(String[] args) {
33
34         //exemple d'utilisation des méthodes de la couche service
35         JpaUtil.init();
36
37         //initialisation des employes dans la base
38         EmployeDAO.initBase();
39
40         //ajout de quelques clients dans la base
41         Date d = new Date(1998, 9, 2);
42         Client salut = new Client("Ferro", "BG", d, "CHOSE", "0668629985", "mao.zedong@insa-lyon.fr", "
43             221_Boulevard_Adrian,_Saint-Raphael", "mdp");
44         Client salut2 = new Client("Lechat", "Felix", d, "CHOSE", "0609837263", "lenine@insa-lyon.fr", "
45             Rue_de_la_République,_Lyon", "mdp");
46         Service.inscription(salut);
47         Service.inscription(salut2);
48
49         //authentification client
50         Service.authentification("mao.zedong@insa-lyon.fr", "mdp");
51         Personne p = Service.authentification("mao.zedong@insa-lyon.fr", "hmmm");
52         Service.authentification("bah", "mdp");
53
54         //authentification employe
55         Employe emp = (Employe) Service.authentification("vladimir.poutine@insa-lyon.fr", "mdp");
56
57         //ajout d'intervention à la base
58         InterventionAnimale interMilan = new InterventionAnimale(salut, new Date(), "blabla", "Chat", "
59             test");
60         InterventionAnimale inter2 = new InterventionAnimale(salut2, new Date(), "miaou", "Chat", "test");
61
62         System.out.println("===="+interMilan.getClass().getName());
63         Service.demandeInter(interMilan);
64         Service.demandeInter(inter2);
65
66         //liste de l'historique du client
67         List<Intervention> historique = Service.historiqueClient(salut);
68         for(Intervention i : historique){
69             System.out.println("___"+i.getDescription());
70         }
71
72         //cloture d'une intervention
73         Service.cloturerIntervention(emp, "c'était très bien", "Terminée");
74

```

```
72         //historique des intervention de la journées
73         List<Intervention> historique2 = Service.interventionsJournee();
74         for(Intervention i : historique2){
75             System.out.println("    "+i.getDescription());
76         }
77
78
79         JpaUtil.destroy();
80     }
81 }
```