```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Web;
5   using System.Data.Linq;
6
7   public class BD_CoEco
8   {
9       /// <summary>
10      /// Méthode obtenant tous les employés
11      /// </summary>
12      /// <param name="p_Actifs">Si ce flag est actif, retourne seulement les employés
        actifs</param>
13      /// <returns></returns>
14      public static List<T_Employe> GetListeEmploye(bool p_Actifs = false)
15      {
16          CoEco_BDDataContext BD = new CoEco_BDDataContext();
17          Table<T_Employe> tableEmp = BD.T_Employe;
18          List<T_Employe> listeEmp = tableEmp.ToList();
19          List<T_Employe> rtnList = listeEmp;
20          //Enlever les admins
21          List<T_Employe> listeFiltre = new List<T_Employe>();
22          for (int i = 0; i < listeEmp.Count; i++)
23          {
24              if (listeEmp[i].idFonction != 3)
25              {
26                  listeFiltre.Add(listeEmp[i]);
27              }
28          }
29          listeEmp = listeFiltre;
30
31          if (p_Actifs)
32          {
33              rtnList = new List<T_Employe>();
34              foreach (T_Employe emp in listeEmp)
35              {
36                  if (emp.idStatus == 1)
37                  {
38                      rtnList.Add(emp);
39                  }
40              }
41          }
42          BD.Dispose();
43
44          return rtnList;
45      }
46
47
48
49      /// <summary>
50      /// Méthode permettant obtenir la liste des projets. Retourne tous les projets par
        défaut
51      /// </summary>
52      /// <param name="p_Inactif">Si ce flag est mit à True, retourne seulement les
        projets actifs</param>
53      /// <returns></returns>
54      public static List<T_Projet> GetListeProjet(bool p_Actifs = false)
55      {
56          CoEco_BDDataContext BD = new CoEco_BDDataContext();
57          Table<T_Projet> tableProjet = BD.T_Projet;
58          List<T_Projet> listeProjet = tableProjet.ToList();
59          List<T_Projet> rtnList = listeProjet;
60          if (p_Actifs == true)
61          {
62              rtnList = new List<T_Projet>();
63              foreach (T_Projet pro in listeProjet)
64              {
65                  if (pro.idStatus == 1)
66                  {
```

```
67                      rtnList.Add(pro);
68                  }
69              }
70          }
71          BD.Dispose();
72          return rtnList;
73      }
74
75      /// <summary>
76      /// Méthode permettant obtenir la liste des employés au projet. Retourne tous les
        emp au projet par défaut
77      /// </summary>
78      /// <param name="p_Inactif">Si ce flag est mit à True, retourne seulement les
        employé au projet actifs</param>
79      /// <returns></returns>
80      public static List<T_EmployeProjet> GetListeEmpPro(bool p_Actifs = false)
81      {
82          CoEco_BDDataContext BD = new CoEco_BDDataContext();
83          Table<T_EmployeProjet> tableEmpPro = BD.T_EmployeProjet;
84          List<T_EmployeProjet> listeEmpPro = tableEmpPro.ToList();
85          List<T_EmployeProjet> rtnList = listeEmpPro;
86
87          if (p_Actifs == true)
88          {
89              rtnList = new List<T_EmployeProjet>();
90              foreach (T_EmployeProjet emppro in listeEmpPro)
91              {
92                      rtnList.Add(emppro);
93              }
94          }
95          BD.Dispose();
96          return rtnList;
97      }
98
99      /// <summary>
100     /// Méthode permettant obtenir la liste des nouveaux projets.
101     /// </summary>
102     /// <param name="p_Inactif">Si ce flag est mit à True, retourne seulement les
        projets actifs</param>
103     /// <returns></returns>
104     public static List<T_Projet> GetListeNewProjet()
105     {
106         DateTime today = DateTime.Now;
107
108         CoEco_BDDataContext BD = new CoEco_BDDataContext();
109         Table<T_Projet> tableProjet = BD.T_Projet;
110
111         List<T_Projet> listeProjet = tableProjet.ToList();
112         List<T_Projet> rtnList = listeProjet;
113
114         rtnList = new List<T_Projet>();
115
116         foreach (T_Projet pro in listeProjet)
117         {
118             if (pro.idStatus == 1)
119             {
120                 if (pro.dateDebut >= today.AddDays(-30))
121                 {
122                     rtnList.Add(pro);
123                 }
124             }
125         }
126         BD.Dispose();
127         return rtnList;
128     }
129
130
131     /// <summary>
132     /// Méthode permettant obtenir la liste des dépenses. Retourne tous les dépenses
```

```
        par défaut
133     /// </summary>
134     /// <returns></returns>
135     public static List<T_Depense> GetListeDepenseEmp(int idEmploye)
136     {
137         CoEco_BDDataContext BD = new CoEco_BDDataContext();
138         Table<T_Depense> tableDepense = BD.T_Depense;
139         List<T_Depense> listeDepense = tableDepense.ToList();
140         List<T_Depense> rtnList = listeDepense;
141 
142             rtnList = new List<T_Depense>();
143             foreach (T_Depense dep in listeDepense)
144             {
145                 if (dep.idEmp == idEmploye)
146                 {
147                     rtnList.Add(dep);
148                 }
149 
150         }
151 
152         BD.Dispose();
153         return rtnList;
154     }
155 
156     /// <summary>
157     /// Méthode permettant d'obtenir les catégories pour un projet en particulié
158     /// </summary>
159     /// <param name="p_projet">Projet dont on veut savoir les catégories</param>
160     /// <returns>La liste des catégories associés au projet</returns>
161     public static List<T_CategoriePro> GetListeCategorie(T_Projet p_projet = null)
162     {
163         CoEco_BDDataContext BD = new CoEco_BDDataContext();
164         Table<T_CategoriePro> tableCategorie = BD.T_CategoriePro;
165         List<T_CategoriePro> listeCategorie = tableCategorie.ToList();
166         List<T_CategoriePro> rtnLst = new List<T_CategoriePro>();
167         if (p_projet != null)
168         {
169             foreach (T_CategoriePro cat in listeCategorie)
170             {
171                 if (cat.idProjet == p_projet.idProjet)
172                 {
173                     rtnLst.Add(cat);
174                 }
175             }
176         }
177         else
178         {
179             rtnLst = listeCategorie;
180         }
181 
182         BD.Dispose();
183         return rtnLst;
184     }
185 
186     /// <summary>
187     /// Méthode permettant d'obtenir les status des projets
188     /// </summary>
189     /// <returns>Liste des status possibles des projets</returns>
190     public static List<T_StatusProjet> GetListeStatusProjet()
191     {
192         CoEco_BDDataContext BD = new CoEco_BDDataContext();
193         Table<T_StatusProjet> tableStatus = BD.T_StatusProjet;
194         List<T_StatusProjet> listeStatus = tableStatus.ToList();
195         BD.Dispose();
196         return listeStatus;
197     }
198 
199     /// <summary>
200     /// Méthode permettant d'obtenir les status des catégories
```

```
201         /// </summary>
202         /// <returns>Liste des status possibles des catégories</returns>
203         public static List<T_StatusCategorie> GetListeStatusCategorie()
204         {
205             CoEco_BDDataContext BD = new CoEco_BDDataContext();
206             Table<T_StatusCategorie> tableStatus = BD.T_StatusCategorie;
207             List<T_StatusCategorie> listeStatus = tableStatus.ToList();
208             BD.Dispose();
209             return listeStatus;
210         }
211 
212         /// <summary>
213         /// Méthode permettant d'obtenir les status des employés
214         /// </summary>
215         /// <returns>Liste des status possibles des employés</returns>
216         public static List<T_StatusEmploye> GetListeStatusEmploye()
217         {
218             CoEco_BDDataContext BD = new CoEco_BDDataContext();
219             Table<T_StatusEmploye> tableStatus = BD.T_StatusEmploye;
220             List<T_StatusEmploye> listeStatus = tableStatus.ToList();
221             BD.Dispose();
222             return listeStatus;
223         }
224 
225         /// <summary>
226         /// Méthode permettant d'obtenir les fonctions des employés
227         /// </summary>
228         /// <returns>Liste des fonctions possibles des employés</returns>
229         public static List<T_FonctionEmploye> GetListeFontionsEmploye()
230         {
231             CoEco_BDDataContext BD = new CoEco_BDDataContext();
232             Table<T_FonctionEmploye> tableFonction = BD.T_FonctionEmploye;
233             List<T_FonctionEmploye> listeFonctions = tableFonction.ToList();
234             BD.Dispose();
235             return listeFonctions;
236         }
237 
238         /// <summary>
239         /// Méthode permettant d'obtenir un employé selon l'ID fourni. Throw une exception
        si pas trouvé
240         /// </summary>
241         /// <param name="id">L'id de l'employé</param>
242         /// <returns></returns>
243         public static T_Employe GetEmpByID(int id)
244         {
245             List<T_Employe> listeEmp = GetListeEmploye();
246 
247             bool trouve = false;
248             int i = -1;
249             while (i < listeEmp.Count -1 && !trouve)
250             {
251                 i++;
252                 if (listeEmp[i].idEmploye == id)
253                 {
254                     trouve = true;
255                 }
256             }
257             if (trouve)
258             {
259                 return listeEmp[i];
260             }
261             //else
262             //{
263             //throw new Exception("Id correspondant à aucun employé existant");
264             //}
265             return null;
266 
267         }
268 
```

```
269         /// <summary>
270         /// Méthode permettant d'obtenir une catégoeie selon l'ID fourni. Throw une
            exception si pas trouvé
271         /// </summary>
272         /// <param name="id">L'id de la catégorie</param>
273         /// <returns></returns>
274         public static T_CategoriePro GetCatByID(int id)
275         {
276             CoEco_BDDataContext bd = new CoEco_BDDataContext();
277             T_CategoriePro cat = bd.T_CategoriePro.Single(f => f.idCategorie == id);
278             bd.Dispose();
279             return cat;
280 
281         }
282 
283         /// <summary>
284         /// Méthode permettant d'obtenir un projet selon l'ID fourni. Throw une exception
            si pas trouvé
285         /// </summary>
286         /// <param name="id">L'id du projet dont on veut obtenir les infos</param>
287         /// <returns>Le projet en question</returns>
288         public static T_Projet GetProByID(int id)
289         {
290             List<T_Projet> listeProjet = GetListeProjet();
291 
292             bool trouve = false;
293             int i = -1;
294             while (i + 1 < listeProjet.Count && !trouve)
295             {
296                 i++;
297                 if (listeProjet[i].idProjet == id)
298                 {
299                     trouve = true;
300                 }
301             }
302             if (trouve)
303             {
304                 return listeProjet[i];
305             }
306             return null;
307         }
308 
309         public static List<T_TypeDepense> GetListeTypeDepense()
310         {
311 
312             CoEco_BDDataContext BD = new CoEco_BDDataContext();
313             Table<T_TypeDepense> tableTypeDepense = BD.T_TypeDepense;
314             List<T_TypeDepense> listeTypeDepense = tableTypeDepense.ToList();
315             BD.Dispose();
316             return listeTypeDepense;
317         }
318 
319         public static void CreateNewProjet(T_Projet p_projet)
320         {
321             CoEco_BDDataContext BD = new CoEco_BDDataContext();
322             int? maxID = 0;
323             BD.PS_GetMaxIdProjet(ref maxID);
324             maxID++;
325             if(maxID == null)
326             {
327                 maxID = 1;
328             }
329             p_projet.idProjet = (int)maxID;
330 
331             BD.T_Projet.InsertOnSubmit(p_projet);
332             BD.SubmitChanges();
333             BD.Dispose();
334         }
335 
```

```
336         public static int? getNewIdProject()
337         {
338             CoEco_BDDataContext BD = new CoEco_BDDataContext();
339             int? maxID = 0;
340             BD.PS_GetMaxIdProjet(ref maxID);
341             maxID++;
342             if (maxID == null)
343             {
344                 maxID = 1;
345             }
346 
347             return maxID;
348         }
349 
350         public static void CreateNewEmploye(T_Employe p_employe)
351         {
352             CoEco_BDDataContext BD = new CoEco_BDDataContext();
353             int? maxID = 0;
354             BD.PS_GetMaxIdEmpolye(ref maxID);
355             maxID++;
356             if (maxID == null)
357             {
358                 maxID = 1;
359             }
360 
361             p_employe.idEmploye = (int)maxID;
362 
363             BD.T_Employe.InsertOnSubmit(p_employe);
364             BD.SubmitChanges();
365             BD.Dispose();
366         }
367 
368         public static void CreateNewEmpAtProject(T_EmployeProjet p_emp)
369         {
370             CoEco_BDDataContext BD = new CoEco_BDDataContext();
371             int? maxID = 0;
372             BD.PS_GetMaxIdEmpPro(ref maxID);
373             maxID++;
374 
375             if(maxID == null)
376             {
377                 maxID = 1;
378             }
379 
380             p_emp.idEmpPro = (int)maxID;
381 
382             BD.T_EmployeProjet.InsertOnSubmit(p_emp);
383             BD.SubmitChanges(); //Si ce bug survient, c'est qu'il n'y a pas pas de projet
                #3... a changer dans AjouterEmp.aspx.cs
384             BD.Dispose();
385         }
386 
387         public static void UpdateEmploye(T_Employe p_employe)
388         {
389             CoEco_BDDataContext BD = new CoEco_BDDataContext();
390 
391             T_Employe newEmp = BD.T_Employe.Single(e => e.idEmploye == p_employe.idEmploye);
392 
393             newEmp.prenom = p_employe.prenom;
394             newEmp.nom = p_employe.nom;
395             newEmp.courriel = p_employe.courriel;
396             newEmp.mdp = p_employe.mdp;
397             newEmp.idFonction = p_employe.idFonction;
398             newEmp.idStatus = p_employe.idStatus;
399             newEmp.loginName = p_employe.loginName;
400 
401             newEmp.congesFeries = p_employe.congesFeries;
402             newEmp.congesMaladie = p_employe.congesMaladie;
403             newEmp.congesPersonnels = p_employe.congesPersonnels;
```

```
404             newEmp.heuresAccumuleesOuSansSolde = p_employe.heuresAccumuleesOuSansSolde;
405             newEmp.vacances = p_employe.vacances;
406 
407             BD.SubmitChanges();
408             BD.Dispose();
409         }
410 
411         public static void UpdateProjet(T_Projet p_projet)
412         {
413             CoEco_BDDataContext BD = new CoEco_BDDataContext();
414 
415             T_Projet newProjet = BD.T_Projet.Single(p => p.idProjet == p_projet.idProjet);
416 
417             newProjet.nom = p_projet.nom;
418             newProjet.idStatus = p_projet.idStatus;
419             newProjet.heureAlloue = p_projet.heureAlloue;
420             newProjet.responsable = p_projet.responsable;
421             newProjet.descript = p_projet.descript;
422             newProjet.dateDebut = p_projet.dateDebut;
423             newProjet.dateFin = p_projet.dateFin;
424 
425             BD.SubmitChanges();
426             BD.Dispose();
427         }
428 
429         public static void UpdateFeuilleDeTemps(T_FeuilleDeTemps p_fdt)
430         {
431             CoEco_BDDataContext BD = new CoEco_BDDataContext();
432 
433             T_FeuilleDeTemps newFdt = BD.T_FeuilleDeTemps.Single(p => p.idFeuilleDeTemps ==
                p_fdt.idFeuilleDeTemps);
434 
435             newFdt.idFeuilleDeTemps = p_fdt.idFeuilleDeTemps;
436             newFdt.idEmp = p_fdt.idEmp;
437             newFdt.idCategorie = p_fdt.idCategorie;
438             newFdt.dimanche = p_fdt.dimanche;
439             newFdt.lundi = p_fdt.lundi;
440             newFdt.mardi = p_fdt.mardi;
441             newFdt.mercredi = p_fdt.mercredi;
442             newFdt.jeudi = p_fdt.jeudi;
443             newFdt.vendredi = p_fdt.vendredi;
444             newFdt.samedi = p_fdt.samedi;
445             newFdt.commentaireDimanche = p_fdt.commentaireDimanche;
446             newFdt.commentaireLundi = p_fdt.commentaireLundi;
447             newFdt.commentaireMardi = p_fdt.commentaireMardi;
448             newFdt.commentaireMercredi = p_fdt.commentaireMercredi;
449             newFdt.commentaireJeudi = p_fdt.commentaireJeudi;
450             newFdt.commentaireVendredi = p_fdt.commentaireVendredi;
451             newFdt.commentaireSamedi = p_fdt.commentaireSamedi;
452             newFdt.approbation = p_fdt.approbation;
453 
454             BD.SubmitChanges();
455             BD.Dispose();
456         }
457 
458 
459         public static void CreateNewCategorie(T_CategoriePro p_categorie)
460         {
461             CoEco_BDDataContext BD = new CoEco_BDDataContext();
462             int? maxID = 0;
463             BD.PS_GetMaxIdCategorie(ref maxID);
464             maxID++;
465 
466             if(maxID == null)
467             {
468                 maxID = 1;
469             }
470 
471             p_categorie.idCategorie = (int)maxID;
```

```
472
473             BD.T_CategoriePro.InsertOnSubmit(p_categorie);
474             BD.SubmitChanges();
475             BD.Dispose();
476         }
477
478         public static List<T_Employe> GetEmpByProject(T_Projet p_projet)
479         {
480             CoEco_BDDataContext bd = new CoEco_BDDataContext();
481             Table<T_EmployeProjet> tableEmpPro = bd.T_EmployeProjet;
482             List<T_EmployeProjet> listEmpPro = tableEmpPro.ToList(); //Tous les liens entre
                employés et projets
483
484             List<T_EmployeProjet> resultRech = new List<T_EmployeProjet>(); //Liste de retour
485
486             foreach (T_EmployeProjet employeProjet in listEmpPro)
487             {
488                 if (employeProjet.idPro == p_projet.idProjet) //S'il est dans la liste
489                 {
490                     resultRech.Add(employeProjet);
491                 }
492             }
493
494             List<T_Employe> listEmp = GetListeEmploye(true);
495             List<T_Employe> rtnList = new List<T_Employe>();
496             foreach (T_EmployeProjet employeProjet in resultRech)
497             {
498                 foreach (T_Employe employe in listEmp)
499                 {
500                     if (employeProjet.idEmp == employe.idEmploye)
501                     {
502                         rtnList.Add(employe);
503                     }
504                 }
505             }
506
507             bd.Dispose();
508             return rtnList;
509         }
510         public static void CreateNewFeuilleDeTemps(T_FeuilleDeTemps p_feuilleDeTemps)
511         {
512             CoEco_BDDataContext BD = new CoEco_BDDataContext();
513             int? maxID = 0;
514             BD.PS_GetMaxIdFeuilleTemps(ref maxID);
515             maxID++;
516
517             if(maxID == null)
518             {
519                 maxID = 1;
520             }
521
522             p_feuilleDeTemps.idFeuilleDeTemps = (int)maxID;
523
524             BD.T_FeuilleDeTemps.InsertOnSubmit(p_feuilleDeTemps);
525             BD.SubmitChanges();
526             BD.Dispose();
527         }
528
529         public static List<T_Depense> GetListeDepense()
530         {
531             CoEco_BDDataContext bd = new CoEco_BDDataContext();
532             Table<T_Depense> tableDepense = bd.T_Depense;
533             List<T_Depense> listeDepense = tableDepense.ToList();
534             bd.Dispose();
535             return listeDepense;
536         }
537
538         public static List<T_Kilometrage> GetListeKilometrage()
539         {
```

```
540             CoEco_BDDataContext bd = new CoEco_BDDataContext();
541             Table<T_Kilometrage> tableKilo = bd.T_Kilometrage;
542             List<T_Kilometrage> listeKilo = tableKilo.ToList();
543             bd.Dispose();
544             return listeKilo;
545         }
546 
547         public static T_TauxKilo GetTauxKilo()
548         {
549             CoEco_BDDataContext bd = new CoEco_BDDataContext();
550             Table<T_TauxKilo> tableTauxKilo = bd.T_TauxKilo;
551             List<T_TauxKilo> listTauxKilo = tableTauxKilo.ToList();
552             bd.Dispose();
553             if (listTauxKilo.Count != 0)
554                 return listTauxKilo[listTauxKilo.Count - 1];
555             else
556                 throw new Exception("Aucun taux de kilométrage");
557         }
558 
559         public static List<T_FeuilleDeTemps> GetListeFeuilleDeTemps()
560         {
561             CoEco_BDDataContext bd = new CoEco_BDDataContext();
562             Table<T_FeuilleDeTemps> tableFeuilleDeTemps = bd.T_FeuilleDeTemps;
563             List<T_FeuilleDeTemps> listFeuilleDeTemps = tableFeuilleDeTemps.ToList();
564             bd.Dispose();
565             return listFeuilleDeTemps;
566         }
567 
568         public static void AddDepense(T_Depense p_newDep)
569         {
570             CoEco_BDDataContext bd = new CoEco_BDDataContext();
571             bd.T_Depense.InsertOnSubmit(p_newDep);
572             bd.SubmitChanges();
573             bd.Dispose();
574         }
575 
576         public static T_Depense GetDepenseById(int id)
577         {
578             CoEco_BDDataContext bd = new CoEco_BDDataContext();
579             T_Depense t = bd.T_Depense.Single(f => f.idDepense == id);
580             bd.Dispose();
581             return t;
582         }
583 
584         public static List<T_Projet> GetProjectByEmp(T_Employe emp)
585         {
586             CoEco_BDDataContext bd = new CoEco_BDDataContext();
587 
588             List<T_Projet> listeProjet = new List<T_Projet>();
589             List<T_EmployeProjet> listeEmpProjet = bd.T_EmployeProjet.ToList();
590 
591             foreach (T_EmployeProjet employeProjet in listeEmpProjet)
592             {
593                 if(employeProjet.idEmp == emp.idEmploye)
594                 {
595                     listeProjet.Add(GetProByID(employeProjet.idPro));
596                 }
597             }
598 
599             return listeProjet;
600         }
601 
602         public static void UpdateDepense(T_Depense depense)
603         {
604             CoEco_BDDataContext bd = new CoEco_BDDataContext();
605             T_Depense depenseToMod = bd.T_Depense.Single(p => p.idDepense ==
                depense.idDepense);
606 
607             depenseToMod.idProjet = depense.idProjet;
```

```
608             //depenseToMod.idCategorie = depense.idCategorie;
609             depenseToMod.idType = depense.idType;
610             depenseToMod.descript = depense.descript;
611             depenseToMod.montant = depense.montant;
612             depenseToMod.ddate = depense.ddate;
613             depenseToMod.idEmp = depenseToMod.idEmp;
614             depenseToMod.aprobation = depense.aprobation;
615 
616             bd.SubmitChanges();
617             bd.Dispose();
618         }
619 
620         public static T_FeuilleDeTemps GetFeuilleDeTempsById(int id)
621         {
622             CoEco_BDDataContext bd = new CoEco_BDDataContext();
623             T_FeuilleDeTemps fdt = bd.T_FeuilleDeTemps.Single(f => f.idFeuilleDeTemps == id);
624             bd.Dispose();
625             return fdt;
626         }
627 
628         public static T_StatusProjet GetStatusProjetById(int id)
629         {
630             CoEco_BDDataContext bd = new CoEco_BDDataContext();
631             T_StatusProjet statpro = bd.T_StatusProjet.Single(o => o.noStatusPro == id);
632             bd.Dispose();
633             return statpro;
634 
635         }
636 
637         public static float banqueHeuresProj(T_Projet proj)
638         {
639             float banqueHeure = 0;
640 
641             List<T_CategoriePro> listCat = GetListeCategorie(proj);
642             List<T_FeuilleDeTemps> listeFdt = GetListeFeuilleDeTemps();
643 
644             foreach (T_FeuilleDeTemps fdt in listeFdt)
645             {
646                 foreach (T_CategoriePro cat in listCat)
647                 {
648                     if (cat.idCategorie == fdt.idCategorie)
649                     {
650                         if (fdt.dimanche!=null)
651                         {
652                             banqueHeure += (float)fdt.dimanche;
653                         }
654                         if (fdt.lundi != null)
655                         {
656                             banqueHeure += (float)fdt.lundi;
657                         }
658                         if (fdt.mardi != null)
659                         {
660                             banqueHeure += (float)fdt.mardi;
661                         }
662                         if (fdt.mercredi != null)
663                         {
664                             banqueHeure += (float)fdt.mercredi;
665                         }
666                         if (fdt.jeudi != null)
667                         {
668                             banqueHeure += (float)fdt.jeudi;
669                         }
670                         if (fdt.vendredi != null)
671                         {
672                             banqueHeure += (float)fdt.vendredi;
673                         }
674                         if (fdt.samedi != null)
675                         {
676                             banqueHeure += (float)fdt.samedi;
```

```
                    }
                }
            }
        }

        return banqueHeure;
    }

    public static float GetTauxKiloAuto()
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        List<T_TauxKilo> listeTauxKilo = bd.T_TauxKilo.Where(o => o.idTypeAuto ==
        1).OrderBy(o => o.idTaux).ToList();
        bd.Dispose();
        return listeTauxKilo[listeTauxKilo.Count - 1].taux;
    }

    public static float GetTauxKiloCamion()
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        List<T_TauxKilo> listeTauxKilo = bd.T_TauxKilo.Where(o => o.idTypeAuto ==
        2).OrderBy(o => o.idTaux).ToList();
        bd.Dispose();
        return listeTauxKilo[listeTauxKilo.Count - 1].taux;
    }

    /// <summary>
    /// Ajoute un nouveau taux
    /// </summary>
    /// <param name="newTaux">nouveau taux</param>
    /// <param name="type">1 = Auto, 2 = Camion</param>
    public static void AddTauxKilo(float newTaux, int type)
    {
        T_TauxKilo tauxKilo = new T_TauxKilo();
        tauxKilo.taux = newTaux;
        tauxKilo.ddate = DateTime.Today;
        tauxKilo.idTypeAuto = type;

        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        bd.T_TauxKilo.InsertOnSubmit(tauxKilo);
        bd.SubmitChanges();
        bd.Dispose();
    }

    public static List<T_TypeAuto> GetListTypesVehicules()
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        List<T_TypeAuto> rtnList = bd.T_TypeAuto.ToList();
        bd.Dispose();
        return rtnList;
    }


    public static int GetIdTauxKilo(int type)
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        List<T_TauxKilo> listeTauxKilo = bd.T_TauxKilo.Where(o => o.idTypeAuto ==
        type).OrderBy(o => o.idTaux).ToList();
        bd.Dispose();
        return listeTauxKilo[listeTauxKilo.Count - 1].idTaux;
    }

    public static void AjouterDepKilometrage(T_Kilometrage km)
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        bd.T_Kilometrage.InsertOnSubmit(km);
        bd.SubmitChanges();
        bd.Dispose();
    }
```

```
743
744         public static T_TauxKilo GetTauxKiloById(int id)
745         {
746             CoEco_BDDataContext bd = new CoEco_BDDataContext();
747             T_TauxKilo tauxKilo = bd.T_TauxKilo.Single(o => o.idTaux == id);
748             bd.Dispose();
749             return tauxKilo;
750         }
751
752         public static T_Kilometrage GetKiloById(int id)
753         {
754             CoEco_BDDataContext bd = new CoEco_BDDataContext();
755             T_Kilometrage kilo = bd.T_Kilometrage.Single(o => o.idKilo == id);
756             bd.Dispose();
757             return kilo;
758         }
759
760         public static void UpdateKilometrage(T_Kilometrage newkilo)
761         {
762             CoEco_BDDataContext bd = new CoEco_BDDataContext();
763             T_Kilometrage oldkilo = bd.T_Kilometrage.Single(o => o.idKilo == newkilo.idKilo);
764
765             //oldkilo.idKilo = newkilo.idTaux;
766             oldkilo.nbKilo = newkilo.nbKilo;
767             oldkilo.commentaire = newkilo.commentaire;
768             oldkilo.ddate = newkilo.ddate;
769             oldkilo.approbation = newkilo.approbation;
770             oldkilo.idTaux = newkilo.idTaux;
771             oldkilo.idEmp = newkilo.idEmp;
772             oldkilo.idPro = newkilo.idPro;
773
774
775             bd.SubmitChanges();
776             bd.Dispose();
777         }
778
779         public static void ApprouverDepenseByID(int id, bool etat)
780         {
781             CoEco_BDDataContext bd = new CoEco_BDDataContext();
782             T_Depense dep = bd.T_Depense.Single(f => f.idDepense == id);
783             dep.aprobation = etat;
784             bd.SubmitChanges();
785             bd.Dispose();
786         }
787
788         public static void ApprouverKilometrageById(int id, bool etat)
789         {
790             CoEco_BDDataContext bd = new CoEco_BDDataContext();
791             T_Kilometrage kilo = bd.T_Kilometrage.Single(f => f.idKilo == id);
792             kilo.approbation = etat;
793             bd.SubmitChanges();
794             bd.Dispose();
795         }
796
797         public static List<T_FeuilleDeTemps> GetFDTByProject(int idPro)
798         {
799             CoEco_BDDataContext bd = new CoEco_BDDataContext();
800             T_Projet projet = bd.T_Projet.Single(f => f.idProjet == idPro);
801             List<T_CategoriePro> lstCategories = GetListeCategorie(projet);
802             List<T_FeuilleDeTemps> lstFDT = GetListeFeuilleDeTemps();
803
804             List<T_FeuilleDeTemps> rtnList = new List<T_FeuilleDeTemps>();
805
806             foreach (T_CategoriePro cat in lstCategories)
807             {
808                 foreach (T_FeuilleDeTemps fdt in lstFDT)
809                 {
810                     if (fdt.idCategorie == cat.idCategorie)
811                     {
```

```
                    rtnList.Add(fdt);
                }
            }
        }

        return rtnList;
    }

    public static void ChangeStatusCategorie(int id, int etat)
    {
        CoEco_BDDataContext bd = new CoEco_BDDataContext();
        T_CategoriePro cat = bd.T_CategoriePro.Single(f => f.idCategorie == id);

        cat.idStatusCat = etat;
        bd.SubmitChanges();
        bd.Dispose();

    }
}
```