TP Programmation WPF

# Trame pédagogique.

L’objectif de ce projet est de fournir aux étudiants les bases de la programmation .Net C#, en utilisant la technologie WPF et le pattern MVVM.

Les étudiants vont développer une application de tri bagages (IHM TriBag), de bout en bout, qui servira de fils conducteur tout au long des séances de TP. Nous réaliserons toute l’implémentation de l’IHM pour ce qui concerne les fonctionnalités liées à la gestion des Vols, les étudiants devront de leur côté et en autonomie travailler sur les fonctionnalités liées aux bagages. Les fonctionnalités liées à la gestion des tâches et notamment la réalisation du diagramme de Gantt étant plus complexe à mettre en œuvre, cet items n’est pas demandé dans les livrable applicatifs à fournir lors du rendu des projets.

Les sujets abordés seront les suivants :

* Présentation du pattern MVVM.
  + Gestion des Evenements.
  + Utilisation des commandes.
  + Validation des données.
  + Liaison de données.
* Accès aux données.
  + Réalisation d’une base SQL Server 2012.
  + Création d’une couche standardisée permettant un accès uniforme aux données quel que soit le modèle physique sous-jacent, via l’utilisation du pattern Factory.
  + Implémentation du modèle adapté à une base de données SQL 2012.
  + Ecriture de requêtes SQL.
  + Utilisation des objets Connexion et command.
  + Utilisation de requêtes paramétrées et de procédures stockées.
* Le langage XAML.
  + Base du langage.
  + Template.
  + Style et Etat.
  + Control personnalisé.
  + Graphisme.
  + Animation.

Contenu

[Trame pédagogique. 1](#_Toc369098563)

[Présentation d’un tri bagages. 3](#_Toc369098564)

[Fonctionnalité de l’IHM TRIBAG. 4](#_Toc369098565)

[Gestion des vols. 5](#_Toc369098566)

[Gestion des Bagages. 6](#_Toc369098567)

[Gestion des tâches. 7](#_Toc369098568)

[Annexes. 7](#_Toc369098569)

[Description des formats de données. 7](#_Toc369098570)

[TP Création de la base de données Sql Server. 8](#_Toc369098571)

[Structure de la base de données SQL. 8](#_Toc369098572)

[Création de la base de données : 8](#_Toc369098573)

[Implémentation de la Factory. 9](#_Toc369098574)

[Exemple d’implémentation de requêtes SQL en C#. 10](#_Toc369098575)

# Présentation d’un tri bagages.

Dans un aéroport internationale l’ensemble des bagages des passagers doit être sécurisé afin d’éviter tout risque d’incident. Le parcourt d’un bagage se décompose généralement suivant le schéma suivant : enregistrement du bagage à la banque d’enregistrement, identification du bagage dans le système de tri bagages, sécurisation du bagage, éjection du bagage sur la jetée du vol et enfin chargement du bagage dans le conteneur.

L’objectif d’un tri bagage consiste à sécuriser et trié les bagages des passagers, la sécurisation se fait via le passage du bagage par les équipements de sûreté, le tri bagage est réalisé par l’affectation des différents vols du jour aux ressources (tapis d’éjection). La lecture du TagBag[[1]](#footnote-1) permet l’identification du bagage (récupération de la fiche bagage) afin de déterminer sur quel vol celui-ci est enregistré et sur quelle jetée ce vol est affecté cela permet de définir sur quelle jetée le bagage doit être éjecté.

Pour que l’ensemble fonctionne correctement le tri bagages est alimenté par 2 flux de données, un flux nommé *‘SFM’* permet de récupérer la liste des vols (tous les jours à 00:30 le système reçoit les vols à J+1) ainsi que le modifications éventuelles en cours de journée et un second flux nommé *‘BSM’* pour les informations bagages, ce flux temps réel alimente le processus de tri bagage en y routant l’ensemble des fiches bagages créées aux niveau des banques d’enregistrements. A chaque bagage est donc associé un message contenant toutes les informations du bagage et de son propriétaire. C’est la lecture du TagBag aux points d’identification du tri qui permet de récupérer la fiche bagage associée.

En résumé pour un fonctionnement nominal du tri bagages il est nécessaire que l’ensemble des vols de la journée soit affecté à une ressource (tapis bagage) en début de journée et que tous bagages injectés dans le system soient étiquetés avec un TagBag en banques d’enregistrements et que le message (via le flux BSM) soit bien réceptionné par le system informatique de tri bagages.

Pour réaliser ce traitement un système de tri bagages est composé :

* D’un ensemble de tapis de convoyage permettant le déplacement des bagages,
* De portiques de lecture nécessaire pour lire le ‘tagBag’ (code barre du bagage) et ainsi connaitre le vol associé au dit bagage.
* De postes d’identification manuelle en cas de non lecture du tagBag.
* D’équipements de sureté pour sécuriser les bagages.
* De jetées permettant le chargement des bagages dans les containers du vol.
* De banques d’enregistrements pour l’insertion et l’étiquetage des bagages.

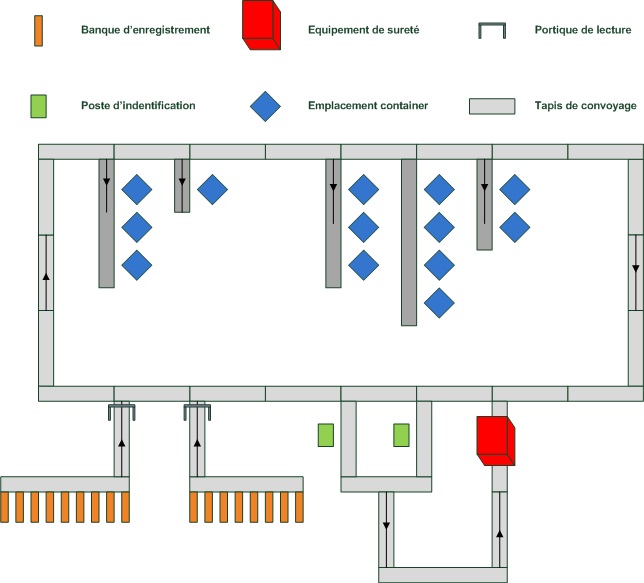
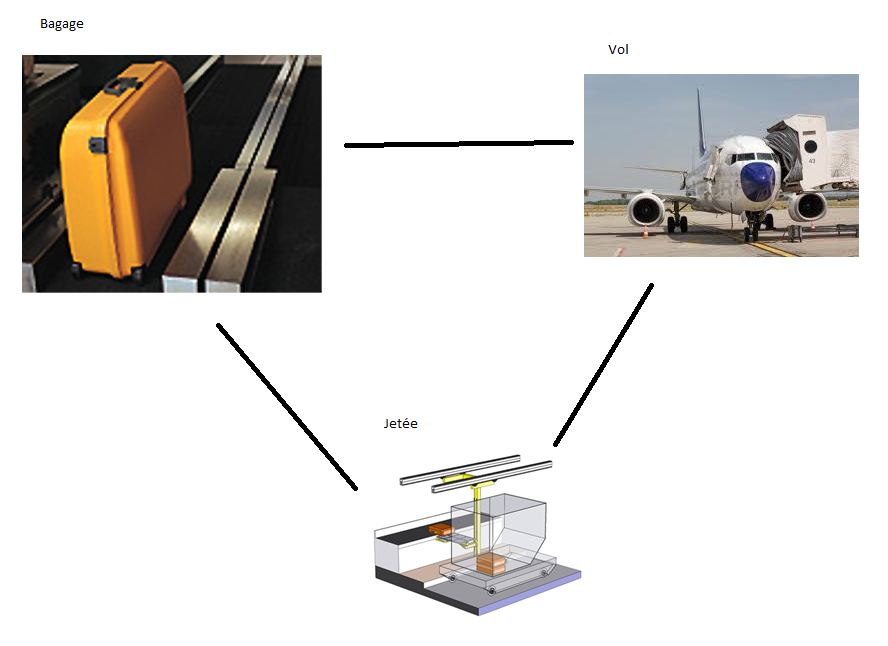


Figure Schéma d'un tri bagages

# Fonctionnalité de l’IHM TRIBAG.

L’application de tri bagage est conçue pour manipuler les 3 types de données nécessaire traitement des bagages : les vols, les bagages et les ressources (Banques et tapis de livraisons). Le principe étant le suivant : Un vol est affecté à 1 tapis de livraisons bagages, un bagage est associé à un vol, un bagage et donc associé à un tapis de livraison. Un tapis de livraison peut être associé à plusieurs vols en simultané et un avion contient (généralement) plusieurs bagages.



# Gestion des vols.

Une fiche vol reçue par le flux de données d’initialisation des vols, contient les informations générales sur le vol, le process de tri bagage ajoute et associe des informations complémentaires à cette fiche afin de représenter au mieux un vol dans notre environnement (liste des bagages associés au vol, ressources utilisées…).

A1. L’IHM doit permettre de rechercher un vol suivant les critères (Compagnie, numéro de vol, date du vol et sortie de tri statut du vol) et de trier le résultat.

A2. L’IHM doit permettre la création d’un vol.

R2. Dans le cas d’une création de vol depuis l’IHM la valeur CRE = TriBag sinon Auto

A3. L’IHM doit permettre l’affichage d’une fiche vol détaillé depuis la liste des recherches. Une fiche détaillée du vol contient en plus des informations du vol, les statistiques bagages ainsi que des liens vers les bagages associés au vol pour chaque statistique.

A4. L’IHM doit permettre l’association d’un vol à une ressource jetée.

R1. Si l’enregistrement du vol est commencé (Voir ressources Chronogramme) alors il n’est plus possible de le changer de ressource jetée.

A5.L’IHM doit permettre de mettre à jours la fiche d’un vol sans avoir à fermer et ré-ouvrir la fiche vol.

A6. L’IHM doit permettre l’impression et l’export (Word, Excel, PDF,…) d’une recherche vol (critères de recherche + liste des vols retournés).

A7. L’IHM doit permettre l’impression et l’export d’une fiche vol.

Les éléments originaux d’une fiche vol sont :

|  |  |  |
| --- | --- | --- |
| Code | Format de données | Description |
| CIE | mm(m) | Code compagnie |
| LIG | fff(f)(a) | Numéro de ligne |
| JEX | ff | Jour d’exploitation 1 à 31 |
| STA | aaa | Code statut du vol |
| TYP | ffff | Type d’avion |
| PAX | ffff | Nombre de passagers |
| IMM | aaaaa | Immatriculation avion |
| PKG | mmm | Parking avion |
| DHC | aaaaaaaaaaaa | Dernier horaire du vol yyyymmjjHHMM |
| ITI | mmm/hhmm\* | Liste des itinéraires du vol avec heure d’arrivée |
| BNQ | mmm(mmm)\* | Liste des banques d’enregistrements réservées |
| MAS | CIE/LIG | Vol Maitre associé (vol opérating) |

Exemple de message :

2013-10-08 00:30:58,870 ; applicatif ; Niveau 3 ; Message SFM recu:

SFM

TIM.201310080030

CMD.INI

VOL.D/AC/2469/20131009

DAT.20131009

SCT.201310090925

DHC.T/201310090925

TYP.764

STA.NORM

NAT.J

DOA.D

CDM.1

TAS.C1

PAX.0

ITI.EWR/1135/J/+/+/N

MAS.UA/057

# Gestion des Bagages.

A chaque enregistrement d’un passager correspond 1 message ‘BSM’ servant à décrire l’ensemble des bagages du passager. Les messages BSM sont donc reçus au fur et à mesure que les passagers s’enregistrent en banque, mais aussi à chaque fois qu’une modification est effectuée sur le statut d’un bagage (autorisation de chargement, suppression du bagage, chargement en soute, …), il est donc nécessaire de pouvoir identifier de manière unique un bagage dans le système.

L’IHM doit permettre diverses actions associé au traitement des bagages.

A1. L’IHM doit permettre de rechercher des bagages et de trier le résultat suivant les critères de recherche s suivant :

* Code IATA une portion de code IATA.
* Liste de compagnie.
* Date de présence du bagage dans le système.
* D’un vol.
* D’une liste de sortie de tri.
* Du type de tâche et du statut temporaire du bagage.
* D’une liste de statut de sûreté.

A2. L’IHM doit permettre d’afficher la fiche détaillée d’un bagage.

A3. L’IHM doit permettre a partir de la fiche détaillée d’un bagage d’accéder à la fiche détaillé du vol associé.

A4. L’IHM doit permettre de mettre à la fiche bagage sans avoir a quitter celle-ci, cette fonctionnalité est importante afin de suivre l’évolution d’un bagage dans le système de tri. (Mise à jour des traces du bagage).

A5. L’IHM doit permettre la visualisation complète de la traçabilité d’un bagage (la traçabilité est en moyenne d’une dizaine de point de contrôle, mais peut parfois être composé d’une centaine de points).

A6. L’IHM doit permettre d’Imprimer et d’exporter (Word, Excel ou Pdf) Une liste de recherche (Critère et résultat) et une fiche bagage (Traçabilité comprise).

# Gestion des tâches.

Afin de planifier les vols sur les jetées, chaque vol maître (ou opérating) est représenté par une ensemble de ‘Tâches’ qui correspondent soit à un statut temporaire du bagage (Anticipé, A temps, dernière minute, raté) soit à un critère de ségrégation (Bagage prioritaire, bagage en correspondance, …). Chacune de ces tâches peut être affectée à une ressource de sortie de tri (une jetée). Le coordinateur de ressource peut à tout moment déplacer une tâche d’une ressource vers une autre tant que le début de la tâche n’est pas commencé.

# Annexes.

## Description des formats de données.

a => Caractère alpha [a-zA-Z]

f => Caractère Numérique [0-9]

m => Caractère AlphaNumérique. [A-Za-z0-9]

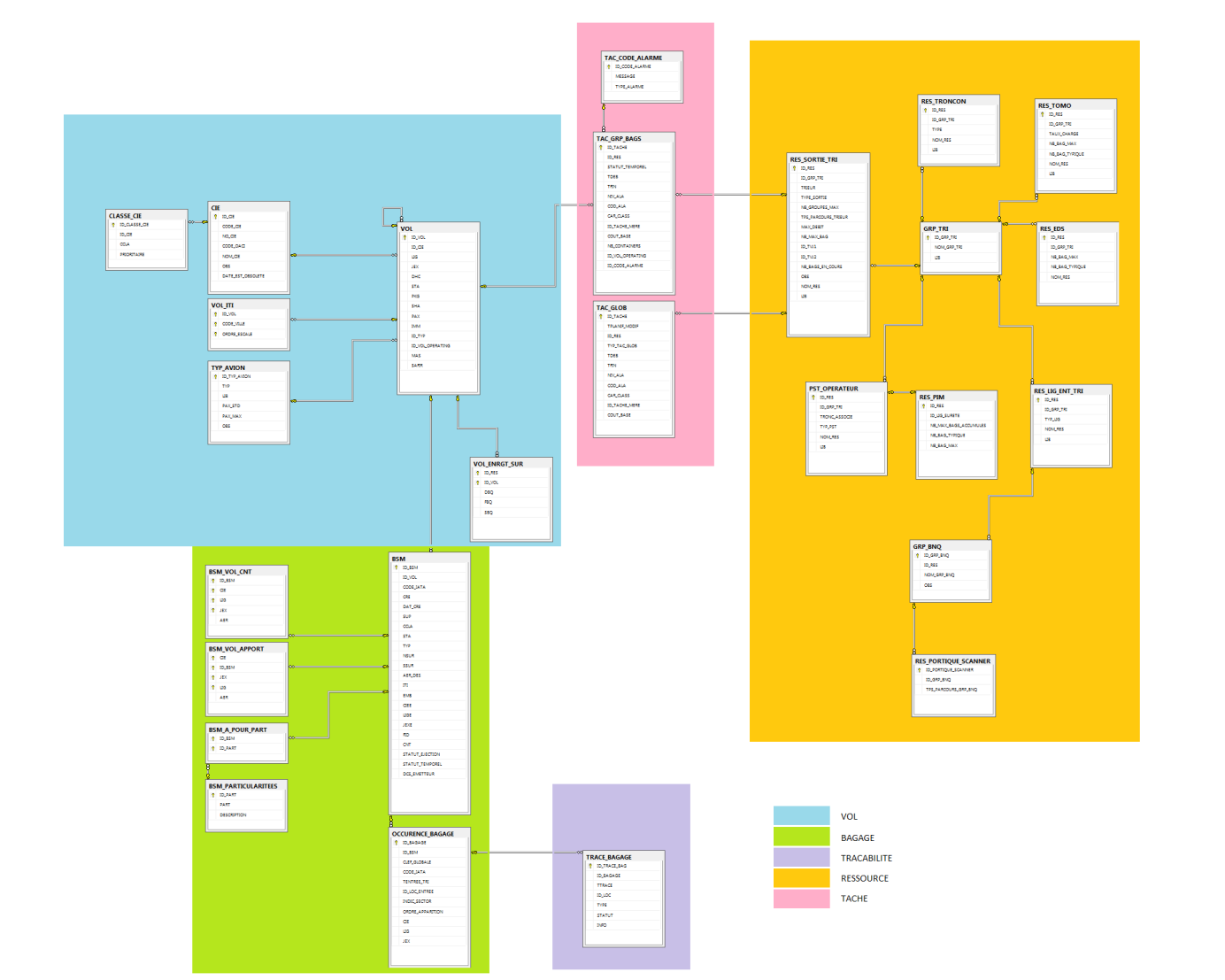
T => Caractère texte

# TP Création de la base de données Sql Server.

Notre application doit disposer d’un mécanisme de stockage et d’accès aux données, plusieurs solutions techniques sont envisageables (SQL Server, MySql, Xml, Acces,…).

## Structure de la base de données SQL.

Durant cette première séance de TP nous allons donc créer une base de données SQL Server (nommée Airport) afin d’y stocker l’ensemble des données. Pour cela il va nous falloir créer plusieurs Tables selon le diagramme ci-dessous :

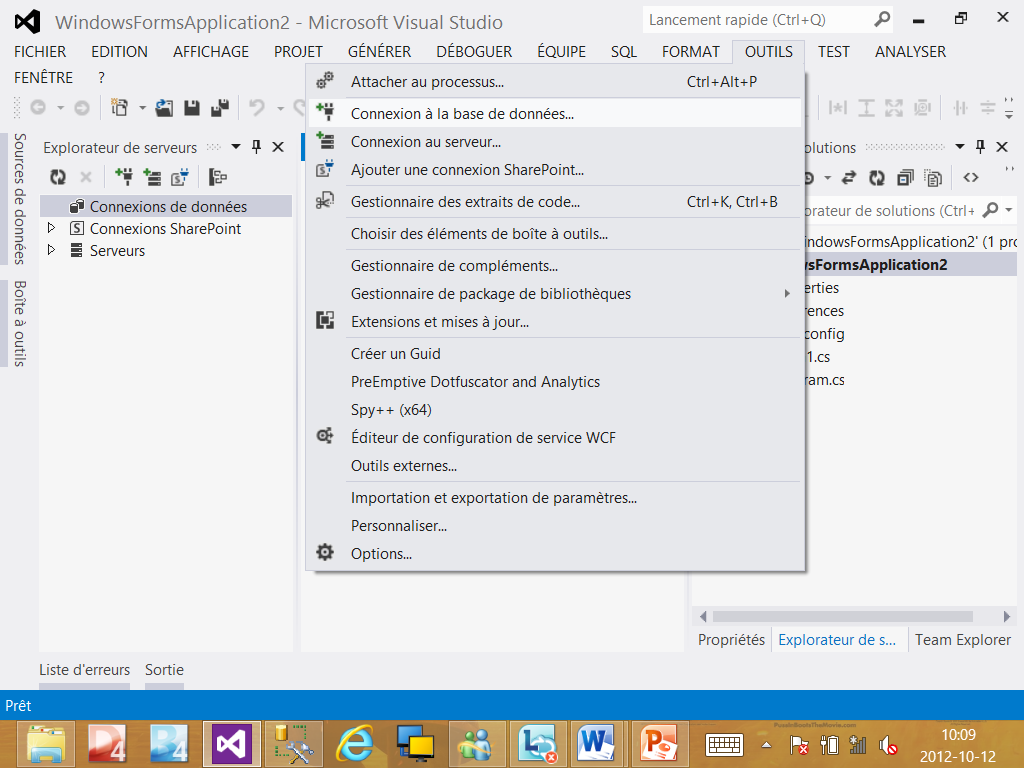


## Création de la base de données :

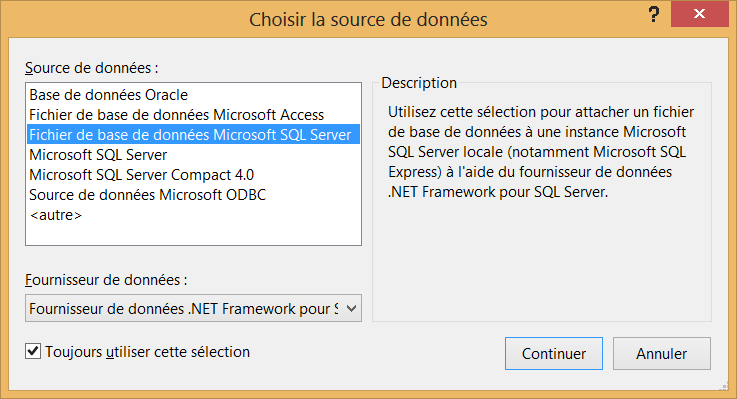
Il existe plusieurs solutions pour générer la base de données suivant les outils à disposition, soit par Visual Studio, soit par Microsoft SQL Server Management Studio. Vous avez à votre disposition soit un script de création de la base de données (sans les datas) soit une sauvegarde de la base de données.

Depuis Visual Studio :

Dans Visual Studio, cliquer sur le menu Outils-> Connexion a une base de données.



Dans source de données : choisir Fichier de base de données Microsoft SQL Server.



## Implémentation de la Factory.

L’utilisation du pattern Factory nous permet d’avoir accès aux ressources manière uniforme quel que soit le modèle physique réel des données, c’est la factory qui se chargera d’appeler le bon constructeur à notre place. Cela implique que toutes les implémentations des modèles de données doivent respecter des spécifications communes. La factory est définie soit en tant qu’interface, soit en tant que classe abstraite.

Les spécifications devant être implémentées par les différents modèles sont déduites des actions listées dans le chapitre : Fonctionnalité de l’IHM TRIBAG.

Pour les vols par exemple nous obtenons la liste ci-dessous (liste non exhaustive)

**A1.** List<Vol> GetVols();

**A1.** List<Vol> GetVols(DateTime debut, DateTime fin, string cie, int ligne);

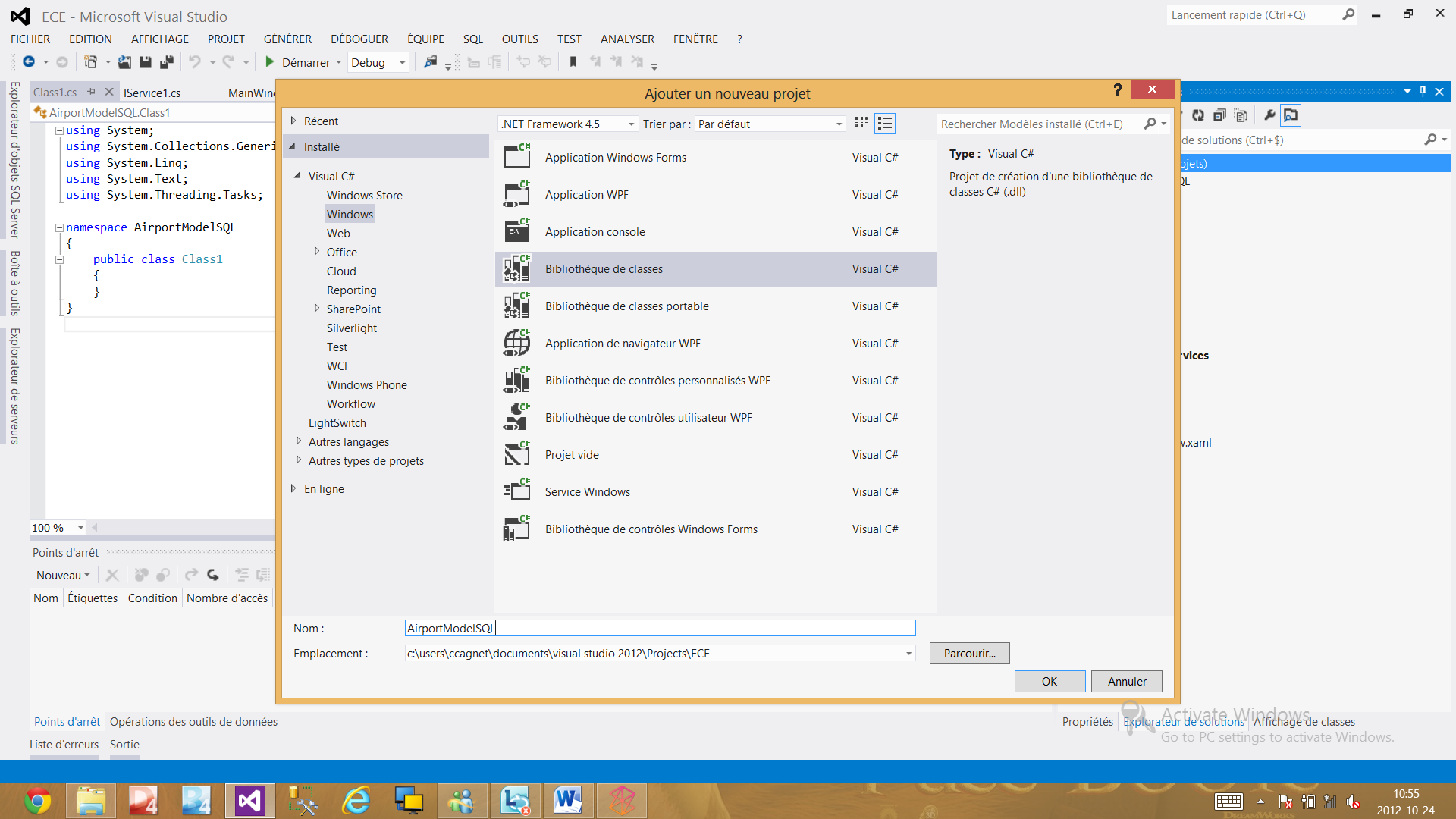
**A1.** List<Vol> FindVols(CriteresVol criteres);

**A2.** Void CreateVol(Vol vol);

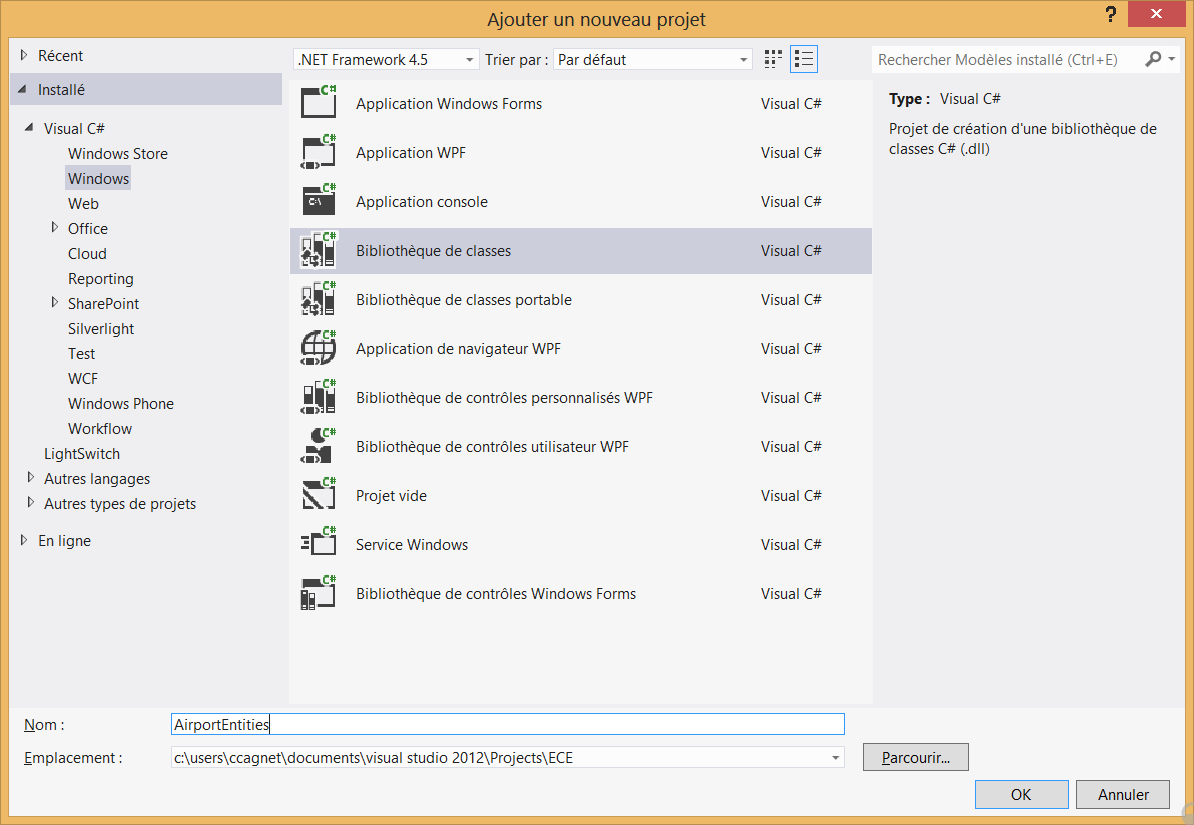
**A3.** Vol GetDetailVol(int idVol);

## Exemple d’implémentation de requêtes SQL en C#.

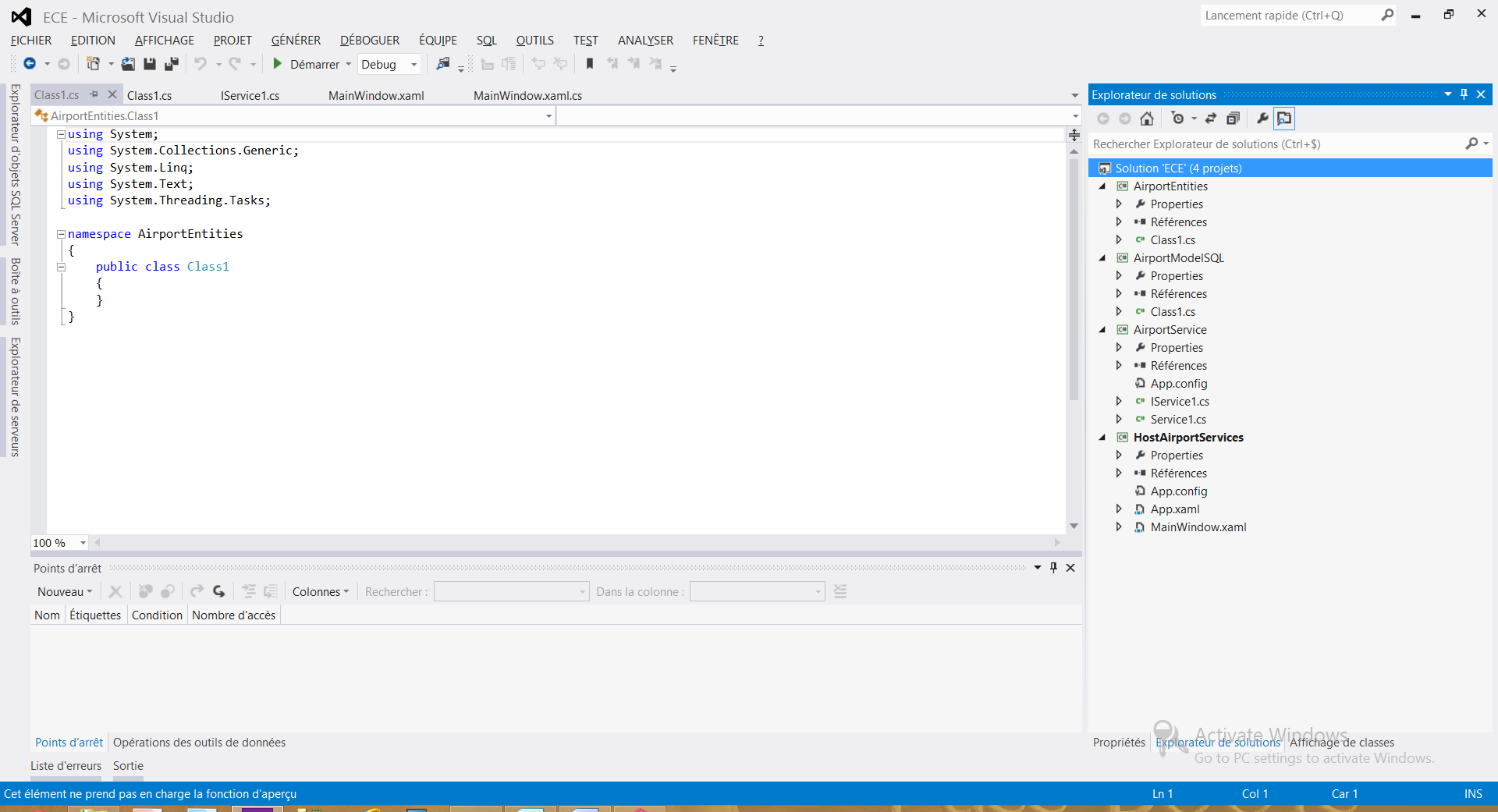
Ajouter un nouveau projet de type bibliothèque de classe (Permettant d’exposer le modelSQL au service)



Et enfin ajouter un nouveau projet contenant les définitions de nos entités métier.



La solution doit ressembler à la capture ci-dessous



Dans le projet AirportModelSql ajouter une référence sur AirportEntities.

Dans le projet AirportService ajouter une référence sur AirportEntities.

Dans le projet AirportService ajouter une référence sur AirportEntities.

Dans le projet AirportEntities ajouter les classes

Création des classes

Bagage, Vol, Trace,

1. TagBag Le TAG bagage corresponds au code barre imprimé sur l’étiquette du bagage, il s’agit d’un numéro à 6 chiffre qui sert à identifier un bagage. [↑](#footnote-ref-1)