

## PROJET DIGITAL FISHING

## OBJECTIFS

- Choisir une solution technique ;
- Mettre en place un ensemble de classes ;
- Manipuler des collections (list) ;
- Interconnecter une application avec une base de données via une classe passerelle.

## CONTEXTE DU PROJET

La société Digital Fishing est une SARL éditrice d'un magazine numérique à parution bimestrielle. Dans chaque numéro, le rédacteur en chef commande des articles à des pigistes qui vont devoir les rédiger. Les pigistes reçoivent en retour un contrat, autrement appelée lettre accord, qui engage les 2 parties contractuellement. Le pigiste est ensuite payé. L'application à réaliser concerne la gestion de cette partie administrative : assurer le suivi des contrats et des paiements pour chaque numéro et chaque pigiste.

## CAHIER DES CHARGES

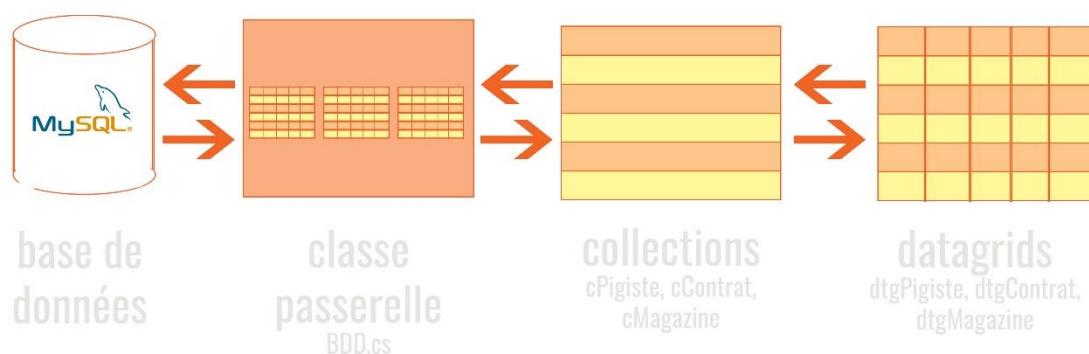
- Une première analyse a abouti au diagramme de classes présenté en page suivante. Ce diagramme devra être implémenté dans une bibliothèque de classes.
- Créer un projet WPF. Son interface devra proposer :
  - Un onglet permettant la gestion des magazines ;
  - Un onglet permettant la gestion des pigistes ;
  - Un onglet permettant la gestion des contrats ;
  - Dans chaque onglet, un datagrid affiche l'ensemble des informations ;
  - Lors de la sélection d'une ligne dans le datagrid, des textboxes permettant l'édition vont se remplir avec les informations de la ligne sélectionnée ;

**Particularités : la lettre accord a un numéro séquentiel du type la-1m2p-XX-YY où XX correspond au numéro de magazine et YY à l'identifiant du pigiste.**

**Compte tenu du temps imparti, vous ne tiendrez pas compte de la vérification des saisies avant que l'application globale ne soit opérationnelle.**

#### ARCHITECTURE DE L'APPLICATION

- La manipulation des données se fait avec des objets provenant de classes : Contrat, Pigiste et Magazine ;
- Les instances de classes (objets) sont stockées temporairement dans des collections ;
- Le datagrid utilise la technologie de binding pour afficher les instances stockées dans les collections ;
- Les données sont stockées dans une base de données MySQL ;
- La manipulation de la base de données (Liste, Ajout, Modification, Suppression) se fait à l'aide d'une classe passerelle.




#### ETAPE 1 : CREATION DES CLASSES

Vous avez à disposition la structure de la base de données actuelles sous MySQL suivante.

##### Travail à faire :

- 1 - Créer les classes correspondantes nommées `Pigiste.cs`, `Contrat.cs` et `Magazine.cs`.
- 2 - Définir la partie privée ainsi que la partie publique (méthodes et attributs) en respectant les conventions de nommage. (Voir [http://lms.lycee-mathias.fr/wiki/doku.php?id=csharp:convention\\_de\\_nommage](http://lms.lycee-mathias.fr/wiki/doku.php?id=csharp:convention_de_nommage))
- 3 - Tester les objets dans le programme principal à travers un test unitaire : déclaration d'un objet, appel du constructeur de base, ajout d'informations et affichage dans une messagebox. (Voir [http://lms.lycee-mathias.fr/wiki/doku.php?id=poo:definition\\_des\\_objets](http://lms.lycee-mathias.fr/wiki/doku.php?id=poo:definition_des_objets) et [http://lms.lycee-mathias.fr/wiki/doku.php?id=poo:utilisation\\_des\\_objets](http://lms.lycee-mathias.fr/wiki/doku.php?id=poo:utilisation_des_objets))
- 4 - Coder un constructeur paramétré.
- 5 - La partie se connectant à la base de données n'étant pas encore fonctionnelle, créer plusieurs instances de chaque classe qui serviront de jeu d'essai (à positionner au lancement de l'application).

## Annexes :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
<input type="checkbox"/> 1	NumCollaboration 	int(6)			Non	Aucune		AUTO_INCREMENT
<input type="checkbox"/> 2	LettreAccordCollaboration	varchar(16)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 3	EtatCollaboration	int(2)			Non	Aucune		
<input type="checkbox"/> 4	AgressaCollaboration	tinyint(1)			Non	Aucune		
<input type="checkbox"/> 5	FactureCollaboration	tinyint(1)			Non	Aucune		
<input type="checkbox"/> 6	MontantCollaboration	int(5)			Non	Aucune		
<input type="checkbox"/> 7	MontantNCollaboration	decimal(10,1)			Non	Aucune		
<input type="checkbox"/> 8	DatePaiementCollaboration	varchar(10)	utf8_general_ci		Non	Aucune		
<input type="checkbox"/> 9	NumPigiste	int(4)			Non	Aucune		
<input type="checkbox"/> 10	NumMagazine	int(4)			Non	Aucune		

Structure de la base de données MySQL concernant les contrats.

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
<input type="checkbox"/> 1	NumPigiste 	int(4)			Non	Aucune		AUTO_INCREMENT
<input type="checkbox"/> 2	NomPigiste	varchar(25)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 3	PrenomPigiste	varchar(25)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 4	AdressePigiste	varchar(50)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 5	CPPigiste	varchar(6)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 6	VillePigiste	varchar(30)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 7	MailPigiste	varchar(50)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 8	NumSecuPigiste	varchar(15)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 9	ContratCadrePigiste	varchar(25)	utf8_unicode_ci		Non	Aucune		

Structure de la base de données MySQL concernant les pigistes.

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
<input type="checkbox"/> 1	NumMagazine 	int(4)			Non	Aucune		AUTO_INCREMENT
<input type="checkbox"/> 2	DateBouclageMagazine	varchar(10)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 3	DateSortieMagazine	varchar(10)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 4	DatePaiementMagazine	varchar(10)	utf8_unicode_ci		Non	Aucune		
<input type="checkbox"/> 5	BudgetMagazine	int(6)			Non	Aucune		

Structure de la base de données MySQL concernant les magazines.

## ETAPE 2 : REALISATION DE L'INTERFACE

L'interface souhaitée se compose d'onglets permettant d'afficher les informations relatives aux contrats, aux pigistes et aux magazines, l'onglet par défaut sera le plus utilisé, à savoir celui concernant les contrats.

Dans chaque onglet, on retrouvera les éléments suivants :

- Un datagrid, affichant les informations relatives sous forme de tableau.
- Des textboxes, permettant l'affichage, la modification et la saisie des informations provenant de la ligne sélectionnée dans le datagrid.
- Des comboboxs permettant une sélection rapide des éléments dans l'onglet Contrats.

**Digital Fishing - Gestion Pigistes**

Contrats Magazine Pigiste

NumContrat	LettreAccordContrat	Mont
374	1m2p-la-25-19	176
372	1m2p-la-25-35	88
370	1m2p-la-25-14	88
369	1m2p-la-25-38	132
368	1m2p-la-25-10	88
367	1m2p-la-25-30	176
366	1m2p-la-25-31	88
365	1m2p-la-25-6	176
363	1m2p-la-25-17	132
362	1m2p-la-25-4	374
361	1m2p-la-25-32	176
360	1m2p-la-25-29	176
359	1m2p-la-25-36	88
358	1m2p-la-25-22	88
357	1m2p-la-25-8	176
356	1m2p-la-25-11	308
355	1m2p-la-25-5	88
354	1m2p-la-25-2	198
353	1m2p-la-24-2	44
352	1m2p-la-24-5	88
351	1m2p-la-24-11	44

Numéro de contrat: 374

Référence Lettre Accord: 1m2p-la-25-19

Montant Brut: 176

Montant Net: 160

Etat du contrat: Contrat à récupérer

Pigiste: Bailly-Basin Eric

Magazine: 25

☐ Facture

☒ Déclaration AGESEA

Ajouter

Modifier

Supprimer

DtgContrat

TextBoxNumContrat – TextBoxLA – TextBoxMtBrut – TextBoxMtNet

CboContrat – CboPigiste – CboMagazine

## ETAPE 3 : LES COLLECTIONS

1 - Le datagrid est synchronisé en fonction des éléments présents dans les différentes collections (Lists).  
L'opération s'appelle le binding et s'effectue en insérant 2 ligne de codes, une en XAML et une autre mdans le programme principal :

```
<DataGrid x:Name="DtgMonDataGrid" ItemsSource="{Binding}">
```

```
DtgMonDataGrid.ItemsSource = cMaCollection;
```

2 - Les collections utilisées se nommeront : cPigiste, cMagazine et cContrat et contiendront respectivement des objets Pigiste, Magazine et Contrat.

3 – Ecrire les procédures évènementielles relatives aux 3 boutons d'ajout, de modification e de suppression.

Digital Fishing - Gestion Pigistes

Contrats Magazine Pigiste

NumContrat	LettreAccordContrat	Montant
374	1m2p-la-25-19	176
372	1m2p-la-25-35	88
370	1m2p-la-25-14	88
369	1m2p-la-25-38	132
368	1m2p-la-25-10	88
367	1m2p-la-25-30	176
366	1m2p-la-25-31	88
365	1m2p-la-25-6	176
363	1m2p-la-25-17	132
362	1m2p-la-25-4	374
361	1m2p-la-25-32	176
360	1m2p-la-25-29	176
359	1m2p-la-25-36	88
358	1m2p-la-25-22	88
357	1m2p-la-25-8	176
356	1m2p-la-25-11	308
355	1m2p-la-25-5	88
354	1m2p-la-25-2	198
353	1m2p-la-24-2	44
352	1m2p-la-24-5	88
351	1m2p-la-24-11	441

Numéro de contrat: 374

Référence Lettre Accord: 1m2p-la-25-19

Montant Brut: 176

Montant Net: 160

Etat du contrat: Contrat à récupérer

Pigiste: Bailly-Basin Eric

Magazine: 25

☐ Facture

☒ Déclaration AGESEA



Ajouter

Modifier

Supprimer

BtnAjoutContrat – BtnModifContrat - BtnSupprContrat

## COMPETENCES MISES EN ŒUVRE

-  C1.2.1.1 Recenser et caractériser des solutions répondant au cahier des charges (adaptation d'une solution existante ou réalisation d'une nouvelle)
-  C1.2.1.3 Rédiger un dossier de choix et un argumentaire technique

---

-  C1.2.2.1 Recenser les composants nécessaires à la réalisation de la solution retenue
-  C1.2.2.2 Décrire l'implantation des différents composants de la solution et les échanges entre eux
-  C1.2.2.3 Rédiger les spécifications fonctionnelles et techniques de la solution retenue dans le formalisme exigé par l'organisation

---

-  C1.2.4.1 Recenser les tests d'acceptation nécessaires à la validation du service et les résultats attendus
-  C1.2.4.2 Préparer les jeux d'essai et les procédures pour la réalisation des tests



---

-  C1.3.1.1 Mettre en place l'environnement de test du service
-  C1.3.1.2 Tester le service
-  C1.3.1.3 Rédiger le rapport de test

---

-  C4.1.1.1 Identifier les composants logiciels nécessaires à la conception de la solution
-  C4.1.1.2 Estimer les éléments de coût et le délai de mise en œuvre de la solution



---

-  C4.1.2.1 Définir les spécifications de l'interface utilisateur de la solution applicative
-  C4.1.2.2 Maquetter un élément de la solution applicative
-  C4.1.2.3 Concevoir et valider la maquette en collaboration avec des utilisateurs

---

-  C4.1.4.1 Recenser et caractériser les composants existants ou à développer utiles à la réalisation de la solution applicative dans le respect des budgets et planning prévisionnels

---

-  C4.1.5.1 Choisir les éléments de la solution à prototyper
-  C4.1.5.2 Développer un prototype
-  C4.1.5.3 Valider un prototype


---

-  C4.1.6.1 Mettre en place et exploiter un environnement de développement
-  C4.1.6.2 Mettre en place et exploiter un environnement de test

---

-  C4.1.7.1 Développer les éléments d'une solution



---

-  C4.1.8.1 Élaborer et réaliser des tests unitaires
-  C4.1.8.2 Mettre en évidence et corriger les écarts

---

-  C5.2.1.1 Évaluer le degré de conformité des pratiques à un référentiel, à une norme ou à un standard adopté par le prestataire informatique
-  C5.2.1.2 Identifier et partager les bonnes pratiques à intégrer

---

-  C5.2.4.1 Se documenter à propos d'une technologie, d'un composant, d'un outil ou d'une méthode
-  C5.2.4.2 Identifier le potentiel et les limites d'une technologie, d'un composant, d'un outil ou d'une méthode par rapport à un service à produire