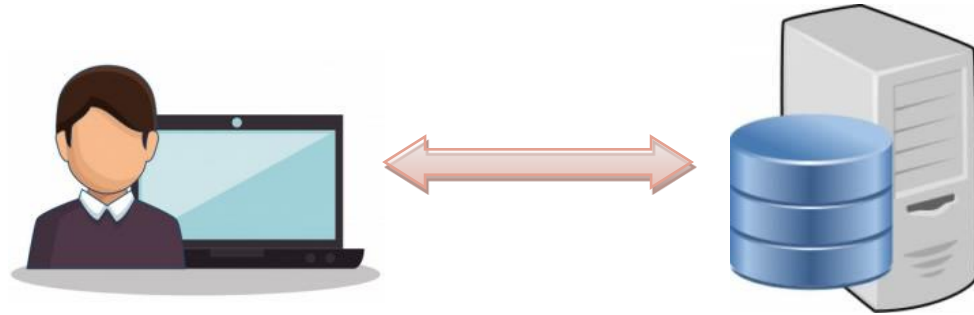


Projet MULTIBURO

Product Backlog

	N°	En tant que...	Je souhaite...	Priorité
App. Bureau [Back-Office]	1	utilisateur	me connecter	Moyenne
	2	gestionnaire	créer une ressource	Haute
	3	gestionnaire	gérer les ressources	Moyenne
	4	gestionnaire	gérer les types de ressources	Basse
App. Web [Front-Office]	5	visiteur	créer un compte client	Moyenne
	6	visiteur	me connecter	Moyenne
	7	client	visualiser la liste des ressources disponibles	Haute
	8	client	réserver une ressource	Haute
	9	client	gérer mes réservations	Moyenne

Application Back-Office Windows Forms



Ordinateur de bureau
(Desktop)

Serveur de base
de données MySQL

Application "WinMultiBuro"

Base "BDD_MULTIBURO"

Périmètre fonctionnel :

- Créer une ressource
- Consulter la liste des ressources
- Gérer les ressources
- Gérer les types de ressources
-

Architecture en couches

COUCHE ACCÈS AUX DONNÉES (DAL - Data Access Layer)

Elle prend en charge l'interaction avec la source de données (BDD MySQL)

Application WinMultiBuro : la couche d'accès aux données assure la connexion au serveur de base de données MySQL et la gestion des interactions avec les tables de la BDD

COUCHE MÉTIER (Business Objects Layer)

Les objets métier correspondent à tous les objets spécifiques qui sont manipulés dans l'application. On part du principe qu'une table dans la base de données représente une classe et que chaque colonne de la table représente un attribut de la classe. Les objets métier pourront remplir le rôle de conteneurs.

Application WinMultiBuro : la couche métier comporte une classe métier pour chaque table de la base de données

COUCHE PRÉSENTATION (GUI Layer)

Cette couche gère la présentation des données et les interactions avec les utilisateurs. Elle correspond à l'interface graphique (GUI - Graphic User Interface) de l'application.

Application WinMultiBuro : la couche présentation correspond aux formulaires de l'application

Mise en place de la solution Visual Studio

- La solution **WinMultiBuro** contient :
 - Un projet de type *Bibliothèque de classes* : MB_DONNEES
 - Un projet de type *Bibliothèque de classes* : MB_METIER
 - Un projet de type *Windows Forms* : MB_PRESENTATION

Sprint 3

	N°	En tant que...	Je souhaite...	Priorité
App. Bureau [Back-Office]	1	utilisateur	me connecter	Moyenne
	2	gestionnaire	créer une ressource	Haute
	3	gestionnaire	gérer les ressources	Moyenne
	4	gestionnaire	gérer les types de ressources	Basse
App. Web [Front-Office]	5	visiteur	créer un compte client	Moyenne
	6	visiteur	me connecter	Moyenne
	7	client	visualiser la liste des ressources disponibles	Haute
	8	client	réserver une ressource	Haute
	9	client	gérer mes réservations	Moyenne

Sprint 3

<i>User Story</i>	<i>Scénario / Critères d'acceptation</i>
<p>[App. Bureau]</p> <p>User Story N°2</p> <p>En tant que gestionnaire, je souhaite créer une ressource</p>	<p>Etant donné que je suis connecté à l'application de bureau, lorsque je demande de créer une ressource alors je peux saisir les caractéristiques de la ressource</p> <p>✓ : la création de la ressource est confirmée, la ressource est enregistrée en BDD</p> <p>✗ :</p>

Base de données MySQL "BDD_MULTIBURO"

- Modélisation de la base de données
 - Diagramme de classes UML
- Création de la table RESSOURCE
 - Script SQL

```
RESSOURCE (idR, libelleR, ...)  
idR : Clé primaire
```


Couche Présentation (MB_PRESENTATION)

- Formulaire de création d'une ressource
 - Maquettage du formulaire avec FIGMA
 - Création du formulaire sur Visual Studio
 - Nommage des contrôles graphiques
 - Création du gestionnaire d'événement `btnValider_Click`
 - Instanciation d'un objet métier `Ressource` à partir des informations saisies par l'utilisateur dans le formulaire
 - Appel de la méthode `InsertRessource()` de la classe `RessourceDAO` pour insérer la ressource dans la table `Ressource`




Ajouter au projet MB_PRESENTATION une référence vers le projet MB_METIER afin de pouvoir utiliser la classe Ressource et une référence vers le projet MB_DONNEES afin de pouvoir utiliser la classe RessourceDAO

Couche Métier (MB_METIER)

- **Classe métier** `Resource`
 - Déclaration des attributs privés (*un attribut pour chaque colonne de la table*)
 - Implémentation du constructeur
 - Implémentation des accesseurs (Getters / Setters)

Couche d'accès aux Données (MB_DONNEES)

- Classe Connexion

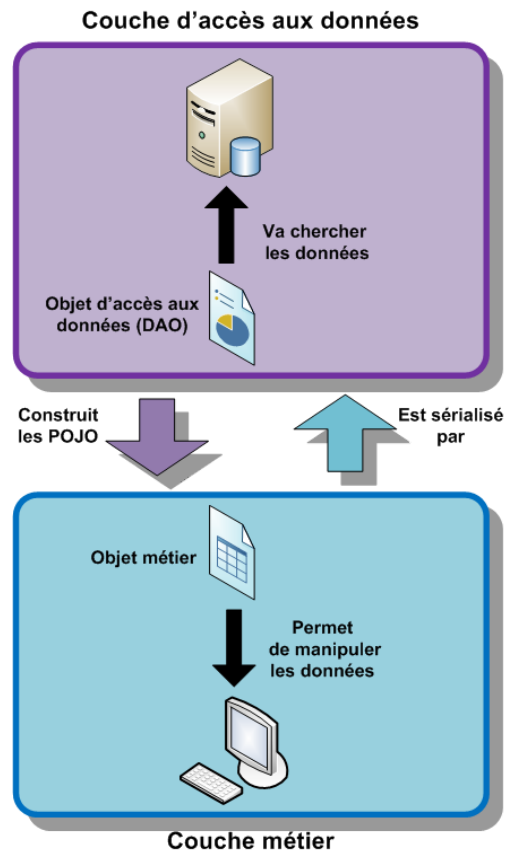
 Ajouter au projet le connecteur
MySQL.Data.dll à l'aide du
Gestionnaire de package Nuget

- Création de la classe statique Connexion
- Implémentation de la méthode statique Connect ()

```
/// <summary>
/// Méthode qui établit une connexion à la base de données MySQL BDD_MULTIBURO
/// </summary>
/// <returns>Un objet MySqlConnection fermé</returns>
public static MySqlConnection Connect()
{
    MySqlConnection connex = null;
    string chaineConnex = "server=localhost; user id=root; password=; database=bdd_multiburo;";
    try
    {
        connex = new MySqlConnection(chaineConnex);
    }
    catch (Exception e)
    {
        Console.WriteLine("Erreur lors de la connexion à la base de données." + e.Message);
    }
    return connex;
}
```

Couche d'accès aux Données (MB_DONNEES)

- Pattern DAO (Data Access Object)



Le **Pattern (patron de conception) DAO (Data Access Object)** permet de faire la correspondance entre les tables de la base de données et les classes métier.

Principe :

- Une classe DAO correspond à une table de la base de données
- Chaque classe DAO contient les méthodes qui permettent de lire, insérer, modifier et supprimer les données de la table correspondante (opérations CRUD : Create, Read, Update, Delete)

Couche d'accès aux Données (MB_DONNEES)

- Classe `RessourceDAO`
 - Implémentation de la méthode statique `InsertRessource()`



Ajouter au projet MB_DONNEES une référence vers le projet MB_METIER afin de pouvoir utiliser la classe `Ressource`

```
/// <summary>
/// Permet d'ajouter une ressource dans la table Ressource
/// </summary>
/// <param name="uneRessource">Objet Ressource à ajouter</param>
public static void InsertRessource(Ressource R)
{
    try
    {
        MySqlConnection connexion = Connexion.Connect();
        connexion.Open();
        MySqlCommand cmdInsert = new MySqlCommand();
        cmdInsert.Connection = connexion;

        cmdInsert.CommandText = "INSERT INTO Ressource(idR, libelleR,...)"
                                + "VALUES (@id, @libelle, ...)";

        cmdInsert.Parameters.AddWithValue("@id", R.GetId());
        cmdInsert.Parameters.AddWithValue("@libelle", R.GetLibelle());
        ...
        int res = cmdInsert.ExecuteNonQuery();
        connexion.Close();
    }
    catch (Exception e)
    {
        Console.WriteLine("Erreur lors de l'insertion" + e.Message);
    }
}
```