Ramzi :
J’ai besoin de la table artiste, campagne et thème, pour les mettre dans des listes déroulantes.

Couche BO :

Evenement.cs
Attributs : id, dateDebut, dateFin, uneCampagne, unTheme, unArtiste
Campagne.cs
Attributs : id, libelle, dateDebut, dateFin, objectif
Theme.cs
Attributs : id, libelle
Artiste.cs
Artistes : id, nom, siteWeb

Couche DAL :

EvenementDAO.cs
AddEvenement()

CampagneDAO.cs
GetLesCampagnes()

ThemeDAO.cs
GetLesThemes()

ArtisteDAO.cs
GetLesArtistes()

Couche BLL :

EvenementManager.cs
GetLesEvenements()

CampagneManager.cs
AddCampagne()

ThemeManager.cs
GetLesThemes()

ArtisteManager.cs
GetLesArtistes()

Stéphane:

J'aurais besoin de la table Artiste, CourantArtistique, ArtisteEvenement

Couche BO :

Artiste.cs
Attributs : id, nom, sitesWeb, CourantArtistique
Deux constructeurs : id, nom, siteWeb et un deuxième : id

CourantArtistique.cs
Attributs : id, libelle, private Artiste LArtiste
Deux constructeurs : id, libelle, LArtiste et un deuxième : id

ArtisteEvenement.cs
Attributs: idArtiste, idEvenement, cachet
Deux constructeur : idArtiste, idEvenement, cachet et un idArtiste, idEvenement

Couche BLL:

ArtisteManager.cs
AddLesArtiste()

CourantArtistique.cs
GetLesCourantArtistique()

ArtisteEvenement.cs
GetLeCachet()

Couche DAL :

ArtisteDAO.cs
AddArtiste()

CourantArtistiqueDAO.cs
GetCourantArtistique()

ArtisteEvenementDAO.cs
GetCachet()

Mathis :
J’ai besoin de la table Agence et la table Ville

Couche BO:
Agence.cs
Attributs : id, nom, rue, telephone, emailContact, siteWeb, typeAgence, Ville uneVille
Un constructeur
Ville.cs
Attributs : numInsee, codePostal, libelle
Un constructeur

Couche BLL:
AgenceManager.cs
AddAgence()

VilleManager.cs
GetLesVilles()

Couche DAL:
AgenceDAO.cs
AddAgence()

VilleDAO.cs
GetLesVilles.cs

Clément :

Base de données :

- Ajout d'une table Campagne dans la base de données :
  - Campagne(id,libelle,dateDeb,dateFin,objectif,idEmploye,id_AgenceEvenementiel, id_AgenceCommunication)
    Clé primaire : id.
    Clés étrangères : idEmploye en référence à id de Employe.
    id_AgenceEvenementiel en référence à id de Agence.
    id_AgenceCommunication en référence à id de Agence.
- Ajout d'une table Employe dans la base de données :
  - Employe(id,nom,prenom,pseudo,idProfil,idService)
    Clé primaire : id.
    Clés étrangères : idProfil en référence à id de Profil.
    idService en référence à id de Service.
- Ajout d'une table Agence dans la base de données :
  - Agence(id,nom,rue,telephone,emailContact,siteWeb,typeAgence,numero_insee)
    Clé primaire : id.
    Clé étrangère : numero_insee en référence à numero_insee de Ville.
- Ajouter des enregistrements sur la table Employe et la table Agence.

Couche BO :

- Ajout d'une classe Client avec ces attributs suivants : id,nom,prenom,pseudo,password.
- Modification de la classe Client :
  - 2 constructeurs avec l'un valorisant tous les attributs et l'autre avec l'id, nom et prénom)
  - (Ajouter un attribut qui sera référence à la classe Profil (private Profil unProfil) et Service (private Service unService)
  - Modification du constructeur pour prendre en compte ses deux attributs
  - Ajouter une propriété sur le nouvel attribut unProfil et sur le nouvel attribut unService.
- Ajout d'une classe Profil avec 2 attributs : id et libelle. + 1 constructeur et des propriétés.
- Ajout d'une classe Service avec 2 attributs : id et libelle. + 1 constructeur et des propriétés.

- Ajout d'une classe Agence avec ces attributs suivants : id,nom,rue,telephone,emailContact,siteWeb,typeAgence.
- Modification de la classe Agence :
    - 3 constructeurs avec l'un valorisant tous les attributs, l'autre avec l'id et l'autre avec id et nom.
    - Ajouter un attribut qui sera référence à la classe Profil (private Ville uneVille)
    - Modification du constructeur pour prendre en compte cet attribut
    - Ajouter une propriété sur le nouvel attribut uneVille
- Ajout d'une classe Profil avec 2 attributs : id et libelle. + 1 constructeur et des propriétés.
- Ajout d'une classe Service avec 2 attributs : id et libelle. + 1 constructeur et des propriétés.
- Ajout d'une classe Campagne avec les attributs suivants : id, libelle,dateDeb, dateFin,objectif. + 2 constructeur avec l'un valorisant tous les attributs et l'autre uniquement sans l'id et des propriétés.
- Modification de la classe Campagne :
    - Ajouter un attribut qui sera référence à la classe Employe (private Employe unEmploye) + deux attributs qui seront référence à la classe Agence (private Agence uneAgenceEvenementiel) et (private Agence uneAgenceCommunication)
    - Modification du constructeur pour prendre en compte ces nouveaux attributs
    - Ajouter une propriété sur les nouveaux attributs unEmploye, uneAgenceEvenementiel et uneAgenceCommunication.

Couche GUI pour la fonctionnalité "Ajout d'une campagne" :

- Ajout de 2 TextBox avec leurs label pour rentrer le libellé de la campagne et objectif. Et 2 datetimepicker pour la date de debut et la date de fin.
- Ajout d'un combobox accompagné de son label pour sélectionner un employé.
- Ajout d'un combobox accompagné de son label pour sélectionner un agence événementiel.
- Ajout d'un combobox accompagné de son label pour sélectionner une agence de communication.
- Ajout d'un bouton "Enregistrer" permettant d'enregistrer la campagne.
- Dans le contrôleur du formulaire FrmAddCampagne :
    - On appelle la méthode GetLesEmployes qui va me donner une collection de type(List) contenant les employés dans la combobox.
    - On appelle la méthode GetLesAgencesEvenementiels qui va me donner une collection de type(List) contenant les agences événementielles dans la combobox.

- On appelle la méthode GetLesAgencesCommunications qui va me donner une collection de type(List) contenant les agences de communications dans la combobox.

- Dans la méthode btnAdd_Click (méthode appelée lors du clic sur le bouton)
  - Récupérer les id de la catégorie sélectionnée dans la liste des employés, des agences événementielles et de communication.
  - Créer l'appel de la méthode AddCampagne pour lui passer ses arguments(libellé, date dé début, date de fin, objectif, id de l'employé, id de l'agence évènementielle et id de l'agence de communication)

Couche BLL pour la fonctionnalité "Ajout d'une campagne":

- Créer la classe CampagneManager
  - Dans la classe, ajout d'une méthode AddCampagne et créer un objet de la classe Employe (avec l'id uniquement), un objet de la classe Agence avec l'id des agences évènementielles et un objet de la classe Agence avec l'id des agences de communications et passer ces objets lors de la création de la campagne avec le libellé, date dé début, date de fin, objectif.

- Créer la classe ClientManager
  - Dans la classe, ajout d'une méthode GetLesEmployes qui va retourner une collection(List) contenant tous les Employes(nom + prénom).
- Créer la classe AgenceManager
  - Dans la classe, ajout d'une méthode GetLesAgencesEvenementiels qui va retourner une collection(List) contenant toutes les agences évènementielles (nom).
  - Dans la classe, ajout d'une méthode GetLesAgencesCommunications qui va retourner une collection(List) contenant toutes les agences de communications (nom).

Couche DAL pour la fonctionnalité "Ajout d'une campagne":

- Création d'une DAO CampaganeDAO
  - Ajout de la méthode AddCampagne contenant une procédure stockée INSERT pour ajouter une campagne dans la table Campagne de la base de données.

- Création d'une DAO EmployeDAO
  - Ajout d'une méthode GetLesEmployes qui va retourner une collection(List) des employés (nom + prénom) avec un procédure stockée SELECT.

- Création d'une DAO AgenceDAO
    - Ajout d'une méthode GetLesAgencesEvenementiels qui va retourner une collection(List) des agences événementielles avec une procédure stockée SELECT...WHERE typeAgence='E'.
    - Ajout d'une méthode GetLesAgencesCommnications qui va retourner une collection(List) des agences de communications avec une procédure stockée SELECT...WHERE typeAgence='C'.

Couche GUI pour la fonctionnalité "Consultation des campagnes" :

- Ajout d'un datagridview contenant toutes les données des campagnes lors du clic sur le bouton.
- Ajout d'un bouton "Afficher les informations" permettant d'afficher les campagnes dans le datagridview.
- Dans la méthode btnAfficher_Click (méthode appelée lors du clic sur le bouton)
    - Appel de la méthode GetLesCampagnes permettant d'obtenir toutes les caractéristiques des campagnes.
    - Association de la liste obtenue au datasource du datagridview.

Couche BLL pour la fonctionnalité "Consultation des campagnes":

- Dans la classe CampagneManager, ajout d'une méthode GetLesCampagnes qui appelle la méthode GetLesCampagnes de la DAL et retourne une collection de toutes les caractéristiques des campagnes de la base de données.

Couche DAL pour la fonctionnalité "Consultation des campagnes":

- Dans la classe CampagneDAO, ajout d'une méthode GetLesCampagnes qui exécute une procédure stockée permettant d'obtenir les caractéristiques de toutes les campagnes : SELECT… FROM Campagne JOIN Employe… JOIN Agence... . Et création d'une instance pour chaque campagne et ajout de cette instance dans la collection qui sera retournée.

Couche GUI pour la fonctionnalité "Modification d'une campagne" :

- Ajout d'un combobox avec son label pour sélectionner la campagne à modifier.
- Ajout d'un panel qui s'affiche lorsque l'on sélectionne une campagne.
  Ce panel contient :
    - 2 TextBox avec leurs label pour modifier le libellé de la campagne et l'objectif. Et 2 datetimepicker pour modifier la date de début et la date de fin.
    - 1 combobox accompagné de son label pour modifier la sélection d'un employé.
    - 1 combobox accompagné de son label pour modifier la sélection d'une agence événementielle.

    - 1 combobox accompagné de son label pour modifier la sélection d'une agence de communication.
    - 1 bouton “Enregistrer” permettant d’enregistrer la modification de la campagne.
    - 1 bouton “Annuler” permettant d’annuler la modification et de retourner à la sélection de la modification d’une campagne
- Dans le traitement à faire à la création du formulaire (constructeur) :
    - Appel de la méthode GetLesCampagnes de la BLL et association de la liste obtenue au datasource du combobox pour obtenir dans celle-ci toutes les campagnes (nom).
    - Appel de la méthode GetLesEmployes de la BLL et association de la liste obtenue au datasource du combobox pour obtenir dans celle-ci une fois révélé dans le panel tous les employés.
    - Appel de la méthode GetLesAgencesEvenementiels de la BLL et association de la liste obtenue au datasource du combobox pour obtenir dans celle-ci une fois révélé dans le panel toutes les agences évènementielles.
    - Appel de la méthode GetLesAgencesCommunication de la BLL et association de la liste obtenue au datasource du combobox pour obtenir dans celle-ci une fois révélé dans le panel toutes les agences de communication.

- Dans le traitement à faire lors de la sélection d’une valeur dans la combobox (évènement selectionChangeCommitted) :
    - Appel de la méthode GetUneCampagneId de la BLL avec l’id de la campagne sélectionnée.
    - Association des caractéristiques de la campagne aux champs du formulaire.

- Dans le traitement à faire lors du clic sur le bouton “Enregistrer”:
    - Appel de la méthode UpdateCampagne de la BLL en transmettant les valeurs saisies dans les différents champs, combobox et datetimepicker et id de la campagne.

<u>Couche BLL pour la fonctionnalité “Modification d’une campagne”:</u>

- Traitement de la méthode GetLesCampagnes de la BLL qui la méthode GetLesCampagnes de la DAL pour retourner les campagnes.
- Ajout d’une méthode GetUneCampagneId qui appelle la méthode GetUneCampagneId de la DAL et transmet l’id de la campagne sélectionnée.
- Ajout de la méthode UpdateCampagne :
    - qui crée une instance de la classe Campagne avec l’id, le libellé, la date de début, la date de fin, l’objectif, l’id de l'employé, l’id de l’agence événementielle et l’id de l’agence de communication transmis.

- Appel de la méthode UpdateCampagne de la DAL en transmettant l'instance de la campagne créée précédemment.

<u>Couche DAL pour la fonctionnalité "Modification d'une campagne":</u>

- Traitement de la méthode GetLesCampagnes de la DAL :
  - qui exécute une procédure stockée permettant d'obtenir les caractéristiques de toutes les campagnes : SELECT… FROM Campagne JOIN Employe… JOIN Agence... Et création d'une instance pour chaque campagne et ajout de cette instance dans la collection qui sera retournée.

- Ajout d'une méthode GetUneCampagneId qui exécute une procédure stockée permettant d'obtenir les caractéristiques d'une campagne : SELECT… FROM Campagne WHERE id= l'id de la campagne sélectionnée. Ensuite cette méthode crée une instance pour la campagne.
- Ajout d'une méthode UpdateCampagne qui exécute un update dans la table Campagne : UPDATE… FROM CAMPAGNE WHERE id = id de la campagne sélectionnée.