BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	SESSION 2013

E5SD : PRODUCTION ET FOURNITURE DE SERVICES

Durée : 4 heures **Coefficient : 5**

CAS EQUIDA

Ce sujet comporte 18 pages dont un dossier documentaire de 11 pages.
Le candidat est invité à vérifier qu'il est en possession d'un sujet complet.

Aucun matériel ni document autorisé

Missions

Mission 1	Analyser la demande	10 pts
Mission 2	Analyser, corriger et optimiser l'envoi de courriels	30 pts
Mission 3	Concevoir une nouvelle version de l'envoi des courriels	20 pts
Mission 4	Adapter la base de données à la gestion des ventes	20 pts
Mission 5	Proposer une solution pour le développement de l'application mobile	20 pts
	Total	100 pts

Dossier documentaire

1. Cahier des charges du service d'élaboration du catalogue des ventes
2. Extrait d'un schéma modélisant les acteurs et leurs besoins
3. Schéma de l'architecture en couches de l'application PLANNING
4. Formulaire d'envoi de courriels et traitements associés
5. Description de la base de données
6. Exemple d'utilisation d'une collection
7. Description des classes métier utilisées pour l'envoi des courriels (extrait)
8. Extrait de la classe *Manager* utilisée pour l'envoi des courriels
9. Extrait des classes utilisées pour l'accès aux données
10. Documents concernant une vente
11. Propositions chiffrées reçues de la société *MobileEfficience* pour la réalisation d'une application mobile
12. Offres de formation de la société *TIC-Formation*

Présentation générale

Créée en 2006, Equida est une société spécialisée dans la vente aux enchères de chevaux de course. Avec un effectif de vingt-sept personnes, la société a réalisé en 2012 un chiffre d'affaires de 87 millions d'euros. Ses clients sont des vendeurs de chevaux, principalement des haras, des entraîneurs et de grands propriétaires de chevaux, situés en France et à l'étranger. Pour être plus proche de sa clientèle étrangère, elle s'appuie sur une quinzaine de correspondants répartis dans de nombreux pays comme l'Irlande, la Turquie, ou encore le Japon.

Equida organise environ dix ventes aux enchères par an sur deux sites :
- l'établissement « Elie de Brignac » situé à proximité de l'hippodrome de Deauville ;
- le manège « Boussac », situé sur l'hippodrome de Saint Cloud.

L'offre de vente aux enchères est variée, elle s'organise en catégories : vente mixte de février, vente d'été, vente d'automne, vente d'élevage, etc.

Lors d'une vente aux enchères, Equida propose différents types de chevaux :
- des *yearlings*, pur-sang anglais âgés d'un an ;
- des inédits, pur-sang anglais de deux ans n'ayant pas encore participé à une course mais déjà travaillés sur le plan de la condition physique ;
- des chevaux à l'entraînement, pur-sang arabes de plus de deux ans ayant déjà participé à des courses ;
- des étalons, chevaux dédiés à la reproduction ;
- des poulinières, juments dédiées à la reproduction.

La société Equida met à la disposition de sa clientèle un calendrier annuel des ventes aux enchères. Chaque client, désireux de mettre aux enchères un cheval, va s'adresser au directeur commercial d'Equida. Une évaluation du cheval est menée par le service technique d'Equida. Si elle s'avère positive, la mise aux enchères du cheval est intégrée dans le catalogue de la vente correspondante.

Cartographie des applications

Pour gérer son activité, Equida utilise :
- un site *web* qui présente la société Equida et permet la consultation du calendrier des ventes, et son téléchargement au format *PDF* ;
- une application PLANNING qui permet de gérer les clients et le calendrier des ventes ;
- un logiciel de comptabilité qui permet de traiter le règlement des ventes.

Une base de données gérée avec *SQL Server* héberge toutes les données nécessaires à ces applications.

Le site *web* et les applications ont été développés par la SSII *SoftSys*, située à Caen. Le calendrier des ventes est géré à l'aide de l'application PLANNING et rendu disponible sur le site *web*. Il indique chaque vente programmée en mentionnant le lieu, la catégorie de la vente et les types de chevaux concernés.

Le catalogue d'une vente est établi par la directrice technique de la société Equida à l'aide d'un logiciel bureautique et mis à disposition sur le site Internet sous la forme d'un fichier au format *PDF*. Il précise pour une vente donnée l'ensemble des informations concernant les chevaux proposés par les clients.

Pour faire face à l'évolution de ses activités et aux demandes de ses principaux clients, la société Equida a souhaité des évolutions de l'application PLANNING, évolutions qui ont été regroupées dans un cahier des charges remis au prestataire informatique *SoftSys*. Employé-e chez *SoftSys*, vous devez intervenir dans la réalisation des évolutions demandées par Equida.

Afin d'être plus réactive aux demandes de changement de la part de ses clients, la société *SoftSys* a adopté la méthode agile *Scrum* pour des projets de petite et moyenne taille, limités à des équipes de dix personnes. Le projet de modernisation du catalogue des ventes pour Equida en fait partie.

La méthode *Scrum* est basée sur une succession d'itérations d'une durée inférieure à un mois appelées *sprints*. L'animateur de l'équipe *Scrum* (*Scrum Master*) a prévu de mener le projet de modernisation du catalogue des ventes en quatre *sprints* d'un mois.

Mission 1 - Analyser la demande

Documents à utiliser : 1, 2 et 3

Vous intégrez le projet de modernisation du catalogue des ventes dans sa phase préparatoire pendant laquelle sont recensés les besoins qui seront retenus pour l'ensemble du projet.

Afin de pouvoir participer à la priorisation des différents besoins, l'animateur de l'équipe *Scrum* (*Scrum Master*) vous confie le cahier des charges pour analyser la demande du client et réfléchir à l'architecture applicative à adopter pour le service d'inscription des chevaux aux ventes par les clients.

TRAVAIL À FAIRE	
1.1	Modéliser pour chaque acteur les besoins relatifs au nouveau service d'élaboration du catalogue des ventes en adoptant le formalisme utilisé par *SoftSys*.
1.2	Rédiger une courte note à destination de vos collègues expliquant en quoi le choix d'une architecture applicative en couches facilitera la réalisation des deux modalités d'accès pour le service d'inscription des chevaux aux ventes par les clients.

Mission 2 – Analyser, corriger et optimiser l'envoi des courriels

Documents à utiliser : 4, 5, 6, 7, 8 et 9

IMPORTANT : le candidat peut choisir de présenter les éléments de code à l'aide du langage de programmation de son choix ou de pseudo-code algorithmique.

La phase préparatoire du projet de modernisation du catalogue des ventes a permis de définir quatre itérations. Une des exigences prioritaires prises en charge par la première itération concerne l'envoi de courriels par le directeur commercial. Ce dernier pourra ainsi envoyer

automatiquement un courriel à chaque client potentiellement intéressé par une vente aux enchères.

Cette première itération est actuellement en fin de phase de réalisation. Le formulaire de l'application PLANNING est opérationnel, les classes de la couche métier ainsi que la classe *Manager* ont été enrichies et la base de données existante a été complétée.

Les tests de validation ont été passés sur l'application obtenue et ont permis de relever un dysfonctionnement : la création des courriels produit une erreur d'exécution lorsque la catégorie de la vente n'intéresse pas tous les clients.

Par ailleurs, une de vos collègues, Mme Byron, a évoqué lors de la dernière mêlée quotidienne (*Daily Scrum*) une optimisation possible dans la méthode « EnvoiCourriel ». Comme en moyenne environ 40% de la totalité des clients doivent être informés par courriel d'une vente, votre collègue vous propose de ne plus appeler la méthode « GetClients » de la classe « ClientDAO », mais d'appeler une nouvelle méthode « GetClientsCateg » de cette classe qui ne retournerait que les clients potentiellement intéressés par la catégorie de la vente.

Le *Scrum Master* vous charge de procéder, d'une part, à l'analyse et correction du dysfonctionnement détecté sur la création de courriels, d'autre part, à l'optimisation du traitement proposée par Mme Byron.

TRAVAIL À FAIRE	
2.1	Rédiger une courte note expliquant au *Scrum Master* la raison du dysfonctionnement signalé et indiquant les corrections à apporter à la méthode « EnvoiCourriel » de la classe « Manager ».
2.2	Réaliser l'optimisation proposée par Mme Byron en écrivant la méthode « GetClientsCateg » et en réécrivant la méthode « EnvoiCourriel ».

Mission 3 - Concevoir une nouvelle version de l'envoi des courriels

Documents à utiliser : 3, 5, et 7

IMPORTANT : le candidat peut choisir de présenter les évolutions de la structure de la base de données sous la forme de son choix (schéma entité-association, diagramme de classes, schéma relationnel, etc.).

À la demande du directeur commercial d'Equida, représentant des clients dans l'équipe projet, une seconde version de l'envoi des courriels est inscrite dans la liste des besoins retenus pour la deuxième itération.

Actuellement, lorsque plusieurs courriels sont envoyés pour une même vente, seul le dernier est mémorisé.

La nouvelle version doit permettre d'envoyer plusieurs courriels relatifs à une vente et d'en conserver la trace. De plus, tous les courriels doivent pouvoir être accompagnés d'un ou plusieurs fichiers en pièce jointe, ces fichiers pouvant être sélectionnés par l'utilisateur dans le formulaire d'envoi.

En début de réalisation de cette deuxième itération, le *Scrum Master* vous confie la mission de concevoir les adaptations nécessaires pour cette nouvelle version de l'envoi des courriels.

TRAVAIL À FAIRE	
3.1	Modifier la structure de la base de données pour prendre en compte l'évolution demandée.
3.2	Pour chaque couche de l'architecture présentée dans le dossier documentaire, indiquer si elle doit subir une évolution et présenter succinctement les modifications nécessaires (sans aller jusqu'à leur réalisation).
3.3	Adapter le diagramme des classes métier utilisées pour l'envoi des courriels pour prendre en compte cette demande d'évolution.

Mission 4 - Adapter la base de données à la gestion des ventes

Documents à utiliser : 5 et 10

La deuxième itération voit aussi le démarrage de la réalisation des fonctionnalités relatives à la gestion des ventes. Il s'agit de faire évoluer le site *web* pour prendre en compte :

- la consultation du catalogue des ventes ;
- la consultation des résultats d'une vente ;
- la consultation en ligne de statistiques sur les ventes passées.

Avant une vente, l'ensemble des informations concernant la vente et le détail des chevaux proposés aux acheteurs devront être consultables sur le site *web*. À l'issue d'une vente, le prix final et l'acheteur de chaque cheval seront mis en ligne. Le détail de chaque enchère passée au cours de la vente doit être mémorisé.

Le *Scrum Master* vous charge de compléter la structure de la base de données.

TRAVAIL À FAIRE	
4.1	Proposer les évolutions nécessaires de la structure de la base de données. *IMPORTANT : le candidat peut choisir de présenter les évolutions de la structure de la base de données sous la forme de son choix (schéma entité-association, diagramme de classes, schéma relationnel, etc.).*

Mission 5 – Proposer une solution pour le développement de l'application mobile

Documents à utiliser : 1, 11 et 12

Le « cahier des charges du service d'élaboration du catalogue des ventes » précise que l'inscription de chevaux à une vente devra pouvoir être faite à l'aide d'une application pour terminal de poche. La société *SoftSys* ne dispose pas en interne des compétences nécessaires à la réalisation de cette application mobile.

Un rapide sondage auprès des clients d'Equida a mis en évidence que 75% d'entre eux utilise un terminal de poche fonctionnant sous *Android*.
Sur le plan technique, deux possibilités sont à l'étude :
- réaliser une application *Android* qui conviendra à la majorité des clients ;
- réaliser un site *web* accessible aux terminaux de poche en utilisant les technologies *Ajax* et *HTML5*.

Deux solutions sont envisagées :
- confier le développement à un prestataire externe (la société *MobileEfficience* a été contactée et a fourni des propositions chiffrées pour le développement de cette solution) ;
- acquérir les compétences en interne en inscrivant l'un des développeurs de l'équipe à une formation adéquate.

Pour la solution de développement prise en charge en interne, il est possible d'envisager :
- la réalisation d'une application Android en interne qui nécessiterait qu'un développeur de SoftSys suive d'abord le module de formation « DM007 » proposé par la société TIC-Formation. On estime ensuite le temps de réalisation à 8 jours/homme ;
- la formation à la réalisation d'un site web mobile qui nécessiterait qu'un développeur de SoftSys suive d'abord les modules « DM015 » et « DM102 ». Le temps de réalisation du site web est ensuite estimé à 10 jours/homme.

Le coût d'un jour/homme de développement interne est fixé chez *SoftSys* à 250 €.

Vous êtes chargé-e de préparer le choix d'une solution s'offrant à *SoftSys* pour produire le service attendu par Equida.

TRAVAIL À FAIRE	
5.1	Présenter les avantages et les inconvénients de chacune des solutions en comparant : • la solution *Android* et la solution *web* d'une part, • le développement en interne et le recours à un prestataire externe d'autre part.
5.2	Proposer la solution qui vous paraît la plus intéressante en termes d'évolutivité, en justifiant la réponse.

DOSSIER DOCUMENTAIRE

Document 1 – Cahier des charges du service d'élaboration du catalogue des ventes

1.1 Processus actuel de l'élaboration du catalogue d'une vente

- Les caractéristiques des clients et le calendrier des ventes aux enchères sont enregistrés dans l'application PLANNING respectivement par le directeur commercial et la directrice technique tout au long de l'année.
- Deux à trois mois avant chaque vente, des bordereaux d'inscription sont envoyés par courrier aux clients, vendeurs potentiels de chevaux. Ces derniers doivent retourner les bordereaux complétés (chevaux proposés à la vente, pères et mères, etc.) au service commercial.
- Avant d'inscrire un cheval à une vente, celui-ci doit être contrôlé par un salarié d'Equida pour déterminer s'il répond aux critères des ventes de la société (origines, caractéristiques physiques, etc.). Le directeur commercial établit le planning des visites de contrôle des chevaux et centralise les comptes-rendus de visite.
- Pour des raisons pratiques, tous les chevaux inscrits ne peuvent pas être retenus : le directeur général est la seule personne habilitée à sélectionner les chevaux qui figureront au catalogue.
- Lorsqu'il est finalisé, le catalogue d'une vente est publié sur le site *web* sous la forme d'un fichier *PDF*.

1.2 Évolutions souhaitées dans le processus d'élaboration du catalogue d'une vente

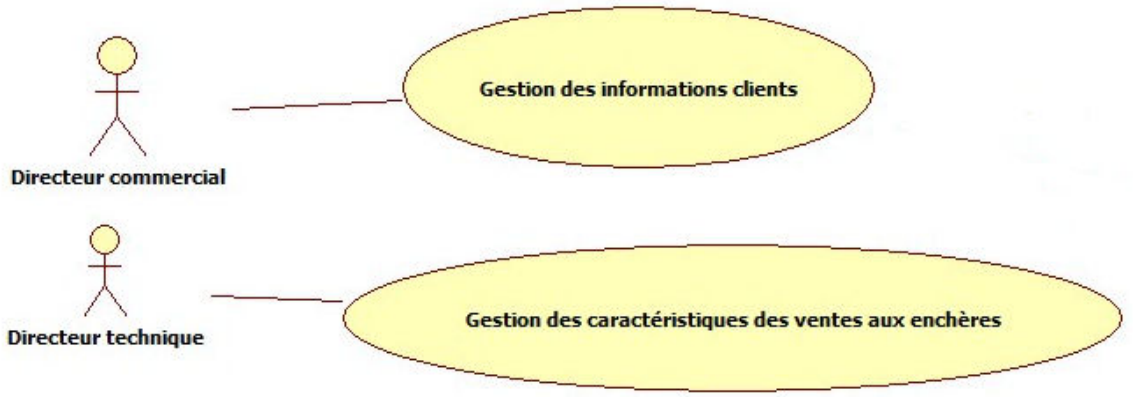
- Dorénavant, chaque client sera informé par courriel de la date de début d'inscription à une vente. Actuellement on dispose de l'adresse de messagerie de chaque client. La fonctionnalité de gestion des clients devra être modifiée car on souhaite que le client puisse choisir la (ou les) catégories(s) de vente aux enchères pour lesquelles il souhaite recevoir un courriel automatiquement.
- Un formulaire permettant de déclencher l'envoi automatique des courriels aux clients doit être ajouté : seul le directeur commercial y aura accès.
- Les clients disposeront sur le site *web* d'un formulaire d'inscription dans lequel ils saisiront eux-mêmes les caractéristiques des chevaux qu'ils souhaitent proposer à la vente ; l'identifiant et le mot de passe de connexion leur seront accessibles à partir d'un lien figurant dans le courriel.
- La gestion du planning des visites de contrôle sera toujours du ressort du directeur commercial, mais se fera à l'aide de l'application PLANNING. Chaque salarié-e ayant réalisé une visite de contrôle devra saisir son compte-rendu dans l'application.
- La sélection finale des chevaux inscrits au catalogue de la vente sera réalisée par le directeur général à l'aide de l'application PLANNING.

1. 3 Modalités d'accès des clients pour l'inscription de chevaux à une vente

Afin de valoriser son image auprès de ses clients, Equida souhaite que les accès proposés au catalogue de ventes soient multiples et en adéquation avec la variété des solutions techniques d'accès dont disposent les clients. Deux modalités d'accès sont donc à prévoir pour l'inscription de chevaux à une vente, que les clients pourront utiliser indifféremment :

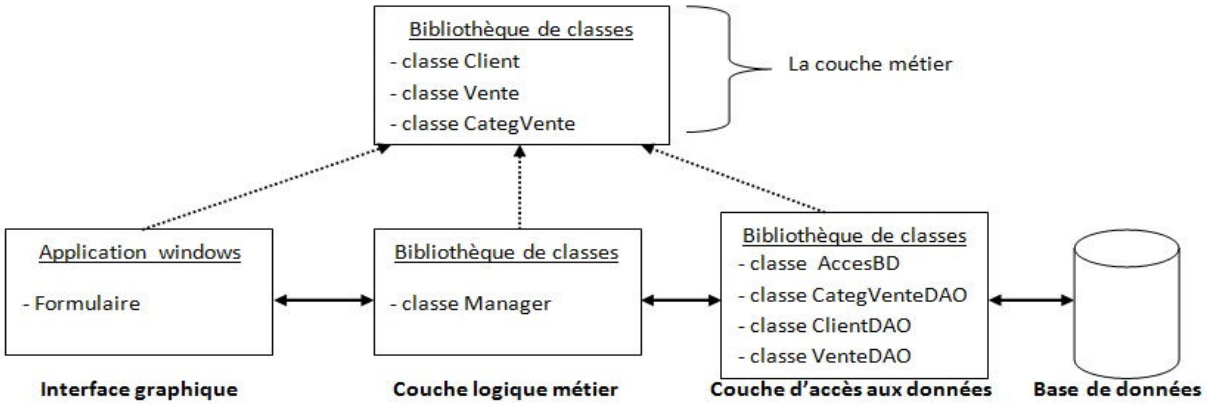
- le site *web* ;
- une application pour terminal de poche (*smartphone* ou tablette).

Document 2 - Extrait d'un schéma modélisant les acteurs et leurs besoins



Le cas « Gestion des informations clients » regroupe l'ajout, la modification et la suppression des caractéristiques d'un client et sera utilisé par le directeur commercial.

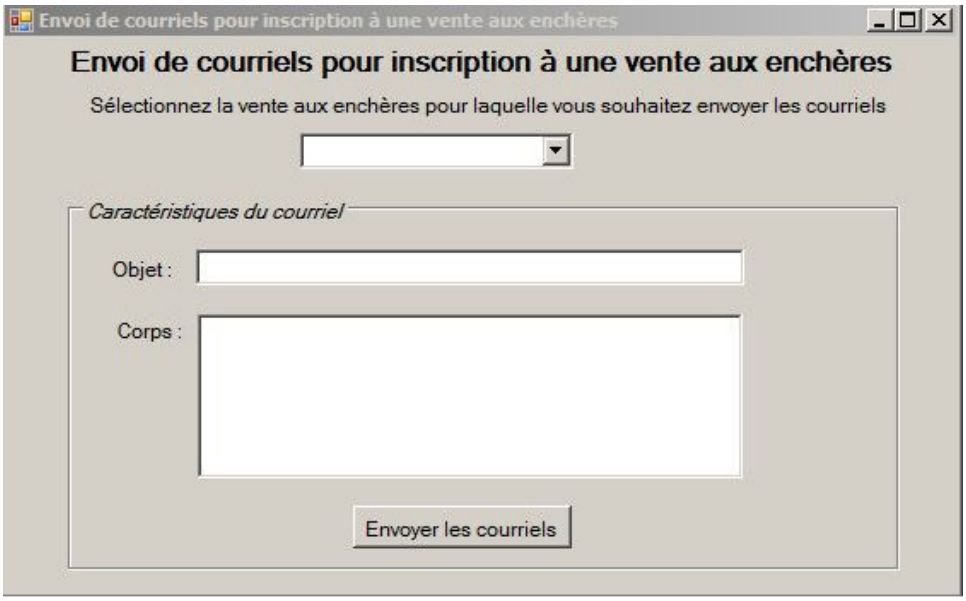
Document 3 - Schéma de l'architecture en couches de l'application PLANNING



- La couche présentation, ou interface graphique, contenant le formulaire est responsable de l'affichage des données à l'utilisateur, données qu'elle obtient de la couche logique métier. Les données saisies dans l'interface graphique sont transmises à la couche logique métier.
- La couche logique métier (BLL : *Business Logic Layer*), responsable du traitement, contrôle les données saisies dans l'interface graphique et dialogue avec la couche d'accès aux données pour obtenir et modifier les données.
- La couche métier contient la description de tous les objets métiers qui seront manipulés dans l'application. Nous aurons donc une classe *Client*, une classe *Vente* et une classe *CategVente*. En général, chaque classe correspond à une table de la base de données.
- La couche d'accès aux données (DAL : *Data Access Layer*) est responsable de la création des objets métiers à partir des données de la base de données et de la mise à jour des données de la base de données à partir des objets métiers. Chaque classe de la couche d'accès aux données est responsable de la lecture, de l'ajout, de la modification et de la suppression d'une table spécifique de la base de données.

Document 4 - Formulaire d'envoi des courriels et traitements associés

Le formulaire ci-dessous a été ajouté à l'application PLANNING :



Lorsque l'utilisateur clique sur le bouton « Envoyer les courriels », la méthode *TraitInfosEnvoi* de la classe *Manager* est appelée. La méthode *TraitInfosEnvoi* fait appel successivement à deux méthodes qui sont *MiseAjourVente* et *EnvoiCourriel*. Le code de la méthode *EnvoiCourriel* est présenté ci-dessous.

Méthode *EnvoiCourriel* de la classe *Manager*

```
01      static public void EnvoiCourriel(Vente uneVente)
02      {
03              int idCateg = uneVente.GetLaCateg().GetId();
04              // Appel de la méthode GetClients qui retourne tous les clients ;
05              // ils seront stockés dans la collection lesClients
06              List<Client> lesClients = ClientDAO.GetClients();
07              List<Client> clientSelectionnes = new List<Client>();
08              foreach (Client unClient in lesClients)
09              {
10                      List<CategVente> lesCategs = unClient.GetLesCategs();
11                      int ind = 0;
12                      bool trouve = false;
13                      while (trouve == false)
14                      {
15                              if (lesCategs[ind].GetId() == idCateg)
16                                      trouve = true;
17                              else    ind++;
18                      }
19                      if (trouve == true) clientSelectionnes.Add(unClient);
20              }
21              CreationCourriel(uneVente, clientSelectionnes);
22      }
```

Document 5 - Description de la base de données

Client (id, titre, nom, prenom, adressePartie1, adressePartie2, codePostal, ville, pays, adresseMessagerie)
clé primaire : id

CategVente (id,libelle)
clé primaire : id
// répertorie les différentes catégories des ventes (vente mixte de février, vente d'été,
// vente d'automne, vente d'élevage, etc.).

Lieu (id, libelle, nbBoxes, commentaires)
clé primaire : id
// contient deux occurrences : Deauville et St Cloud.

Vente (id, nom, idCategVente, idLieu, dateDebutVente, dateFinVente, dateDebutInscription, dateEnvoiMessage, objetMessage, corpsMessage)
clé primaire : id
clé étrangère : idCategVente en référence à id de CategVente
clé étrangère : idLieu en référence à id de Lieu

Interesser (idClient, idCategVente)
clé primaire : idClient, idCategVente
clé étrangère : idCategVente en référence à id de CategVente
clé étrangère : idClient en référence à id de Client
// répertorie les catégories de vente pour lesquelles le client souhaite être informé par courriel
// du début des inscriptions.

TypeCheval (id, libelle, description)
clé primaire : id
// répertorie les différents types de chevaux (yearling, poulinière, étalon, foal,, cheval à
// l'entrainement, pur-sang arabe, pur-sang anglais , etc.).

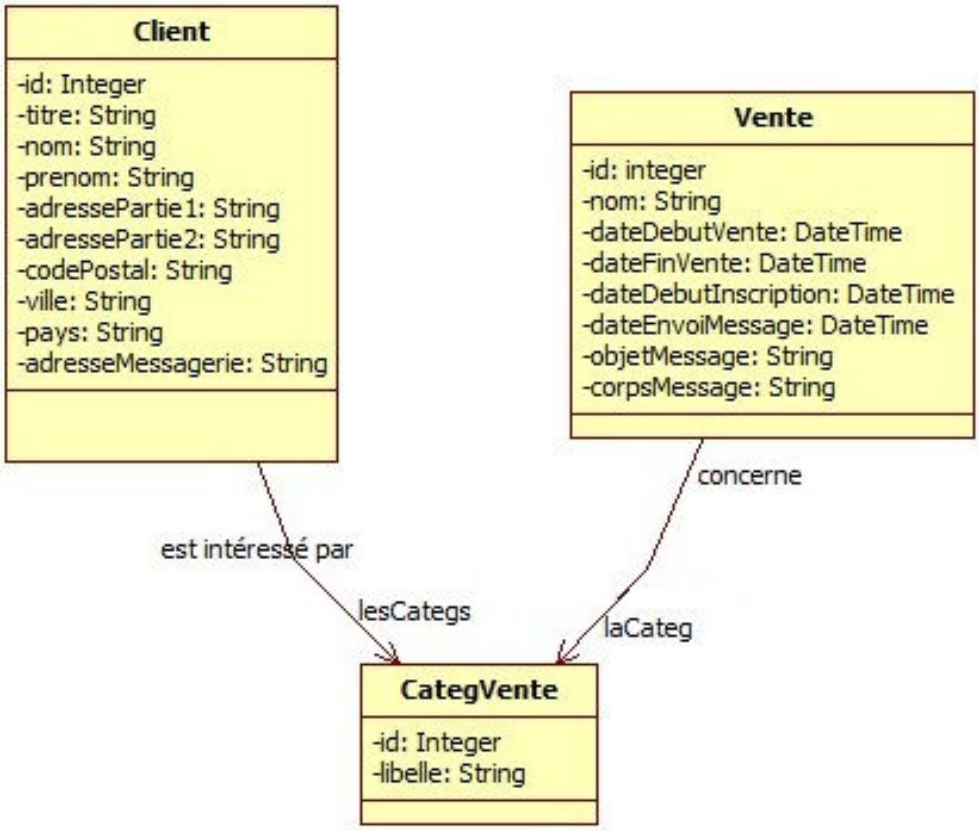
Concerner (idVente, IdTypeCheval)
clé primaire : idVente, idTypeCheval
clé étrangère : idVente en référence à id de Vente
clé étrangère : idTypeCheval en référence à id de TypeCheval
// répertorie les types de chevaux mis en vente lors d'une vente.

Document 6 – Exemple d'utilisation d'une collection

L'exemple ci-dessous permet de manipuler une collection de chaînes de caractères. Le principe est le même quelque soit le type des éléments.

```
List<string> mesChaines; // déclaration d'une collection de chaînes de caractères
mesChaines = new List<string>();          // instanciation de la collection
mesChaines.Add("un");                     // ajout d'une chaîne à la collection
mesChaines.Add("deux");
mesChaines.Add("trois");
foreach (string uneChaine in mesChaines)  // parcours de la collection
{
   Console.WriteLine(uneChaine);          // affichage de l'élément courant
}
mesChaines.RemoveAt(1);                   // suppression du 2ème élément (indice 1)
Console.WriteLine(mesChaines[0]);         // affichage du 1er élément (indice 0)
```

Document 7 - Description des classes métier utilisées pour l'envoi des courriels (extrait)



Remarque :

- Les constructeurs et les accesseurs ne sont pas représentés sur le diagramme.
- En général chaque classe correspond à une table de la base de données.

class Vente

```
{
        // attributs privés
        private int id;
        …
        private DateTime dateEnvoiMessage;
        private string objetMessage, corpsMessage;
        // catégorie de la vente
        private CategVente laCateg;
        // constructeur qui valorise tous les attributs privés
        public Vente(int numero,  [ ...]  CategVente laCateg) {...}
        // méthodes get et set permettant la lecture et la modification des attributs privés
        // permet d'obtenir la catégorie de la vente
        public CategVente GetLaCateg() {...}
        [...]
}
```

class Client

```
{
        // attributs privés
        private int id;
        …
        private string adresseMessagerie;
        // collection des catégories de vente choisies par le client
        private List<CategVente> lesCategs;
        // constructeur permettant de valoriser un objet avec ses catégories de vente
        public Client(  int numero, string titre, string nom, string prenom, string adr1,
                        string adr2, string codePostal, string ville, string pays, string mail,
                        List<CategVente> lesCategs) {...}
        // constructeur permettant de valoriser un objet sans ses catégories de vente
        public Client(  int numero, string titre, string nom, string prenom, string adr1,
                        string adr2, string codePostal, string ville, string pays, string mail) {...}
        // permet d'obtenir les catégories de vente pour lesquelles le client souhaite être  informé
        public List<CategVente> GetLesCategs() {...}
        // méthodes get et set permettant la lecture et la modification des attributs privés
        [...]
}
```

class CategVente

```
{
        // attributs privés
        private int id;
        private string libelle;
        // constructeur qui valorise tous les attributs privés
        public CategVente(int numero, string unLibelle) {...}
        // méthodes get et set permettant la lecture et la modification des attributs privés
        // permet d'obtenir l'id de la catégorie
        public int GetId() {...}
        [...]
}
```

Document 8 - Extrait de la classe *Manager* utilisée pour l'envoi des courriels

class Manager

```
{
        // méthode appelée lors de la validation du formulaire :
        public  static void TraitInfosEnvoi(Vente laVente, string objet, string corps)
        {
                MiseAJourVente (laVente, objet, corps);
                EnvoiCourriel (laVente);
        }
        private static void EnvoiCourriel(Vente uneVente) // voir document 4
        {...}
        // crée et envoie un courriel à chaque client de la collection « lesClients »
        private static void CreationCourriel(Vente uneVente, List<Client> lesClients)
        {...}
        public static void MiseAJourVente(Vente laVente, string objet, string corps)
        {
                laVente.SetDateEnvoiMessage(DateTime.Now);
                laVente.SetObjetMessage(objet);
                laVente.SetCorpsMessage(corps);
                VenteDAO.MiseAJourVente(laVente);
        }
}
```

Document 9 - Extrait des classes utilisées pour l'accès aux données

class Connect

```
{
        // retourne un objet représentant la connexion à la base de données
        public static SqlConnection Get() {…}
}
```

class ClientDAO

```
{
        // met à jour la table Client (requête update) à partir des valeurs contenues
        // dans l'objet unClient et retourne 1 si la mise à jour a eu lieu, 0 sinon
        public static int MiseAJourClient (Client unClient)  { ....}
        // ajoute un enregistrement à la table Client à partir des valeurs contenues
        // dans l'objet unClient et retourne 1 si la création a eu lieu, 0 sinon
        public static int CreationClient (Client  unClient)  { ....}
        //  supprime l'enregistrement de la table Client,
        //  et retourne 1 si la suppression a eu lieu, 0 sinon
        public static int SuppressionClient (Client unClient)  { ....}
        // retourne les nom et prénom d'un client à partir de son Id
        public static string GetNomClient (int idClient)
        {
                string res;
                // constitution d'une commande basée sur une requête SQL
                // en vue d'être exécutée sur une connexion donnée
                string req = "select nom, prenom from client where id=" + idClient;
                SqlCommand cmd = new SqlCommand(req,Connect.Get());
                // demande d'exécution de la commande produisant un jeu d'enregistrements résultats
                SqlDataReader jeu = cmd.ExecuteReader();
                // lecture de la première ligne du jeu d'enregistrements résultats,
                // opération à renouveler pour lire les enregistrements suivants éventuels
                if (jeu.Read())
                {
                        res = jeu[0] + " " + jeu[1];
                }
                else
                {
                        res = "";
                }
                // libération du jeu d'enregistrements résultats
                jeu.Close();
                return res;
        }
        // retourne tous les enregistrements de la table Clients
        public static List<Client> GetClients()  { ....}
        [...]
}
```

Document 10 – Documents concernant une vente

Extrait du catalogue de la vente d'été du 12 juillet à Saint-Cloud

N° lot	Sexe	Type	Nom	Père	Mère	Vendeur	Prix départ
1	M	pur-sang anglais	Valdack	Houri	Yastale	Brouselle	10 000 €
2	M	yearling	Trais d'or		Doune	Longuet	7 000 €
3	F	pur-sang anglais	Herricka	Houri	Hussa	Brouselle	56 000 €
4	M	yearling	Nuage	Kop	Jarria	Louvel	6 500 €
5	M	yearling	Desperado	Lappon		Gorlier	60 000 €
…	…	…	…	…	…	…	

Détail des informations concernant le lot n° 3

Lot n°3 :	Herricka, pur-sang anglais femelle N° SIRE : 0808.000.020Z	
Mise à prix : 56 000 euros	Vendeur : Brouselle	
Père	Houri, pur-sang anglais	
Mère	Hussa, pur-sang anglais	
Résultats en course Prix Dahman le 10/06/2012 à Dax : 3 ème place Prix Danbik le 25/05/2012 à Aurillac : 1 ère place Prix Pierre Pechdo le 06/05/2012 à Pompadour : 4 ème place		

Détail des enchères concernant le lot n°3 lors de la vente d'été

Lot n°3 :	Herricka, pur-sang anglais femelle N° SIRE : 0808.000.020Z	
N° enchère	Acheteur	Montant
1	Brindeuil	58 000 €
2	Louvel	65 000 €
3	Brindeuil	75 000 €
4	Malpart	80 000 €
5	Louvel	95 000 €
Résultat : adjugé à Mme Louvel Jeanine pour 95 000 €		

Remarque :

- N° SIRE : numéro d'identification international d'un cheval.
- Il est nécessaire de connaître l'adresse complète de chaque vendeur et de chaque acheteur ainsi que leur adresse de messagerie.
- Un vendeur est un client d'Equida. Certains clients peuvent également être acheteurs de chevaux (c'est par exemple le cas de Mme Louvel).
- Si aucune enchère n'est proposée, le cheval n'est pas vendu. Dans le cas contraire, il est automatiquement attribué à l'acheteur ayant fait la dernière enchère.

Document 11 – Propositions chiffrées reçues de la société *MobileEfficience* pour la réalisation d'une application mobile

Proposition 1 : développement d'une application *Android*
Développement de l'application ciblant les Smartphones fonctionnant sous *Android* Interfaçage avec l'existant Mise en service et tests Maintenance corrective garantie six mois
Coût total : 2 500 € HT

Proposition 2 : développement d'un site web accessible aux *smartphones*
Réalisation du site *web* intégrant les technologies HTML5 et *JQuery Mobile* Interfaçage avec l'existant Mise en service et tests Maintenance corrective garantie six mois
Coût total : 3 500 € HT

Document 12 – Offres de formation de la société *TIC-Formation*

Module DM007 : développez des applications d'entreprises pour *Google Android*
Durée : 3 jours Coût : 1 500 € HT
Le rachat de la startup *Android* (inventeur du système d'exploitation éponyme) par *Google* en 2005 a permis au géant californien de se poser en concurrent sérieux de l'*iPhone*. La force d'*Android* ? Une plateforme ouverte à tous, tant à l'industrie, qu'aux développeurs et aux utilisateurs eux-mêmes et un kit de développement (SDK) qui rend le développement d'applications embarquées aussi facile que celui de sites *web*. Les participants à cette formation *Android* apprendront à développer une application fonctionnant sur la plateforme *Android* en utilisant le SDK.
Objectifs
- Être capable de développer une application fonctionnant sur la plateforme *Android* en utilisant le SDK fournit par *Google* - Connaître les spécificités du développement mobile et en particulier d'*Android* - Savoir utiliser les fonctionnalités spécifiques aux téléphones *Android*
Contenu
- Le développement *Android* : les premiers pas - Architecture d'une application *Android* - Composer une interface utilisateur - Utiliser des menus - Gestion des données - Services et multithreading - Spécificité du développement mobile *Android* - Déployer une Application *Android*

Module DM015 : Développer des interfaces *web* riches avec *Ajax*
Durée : 2 jours Coût : 1 120 € HT
L'émergence du *web* 2.0 s'accompagne de nombreuses avancées technologiques. Parmi celles-ci, *Ajax* (*Asynchronous JAvascript* +XML) permet d'étendre les possibilités de *scripting* dans les navigateurs et ainsi de rendre les pages *web* plus interactives et conviviales pour l'internaute. Cette formation a pour objectif de fournir aux chefs de projet et développeurs une approche rationalisée et enrichie de premiers retours d'expériences sur le phénomène.
Objectifs
- Découvrir les apports d'*Ajax* sur les interfaces *web* - Comprendre en quoi *Ajax* constitue une avancée technologique majeure - Identifier les pièges à éviter en phase de conception - Savoir utiliser les scripts *Ajax* pour rendre les pages *web* plus interactives
Contenu
- Bien développer pour le *web* 2.0 - *Javascript* : un vrai langage à maîtriser - Ajax par l'exemple - Panorama des utilisations d'*Ajax* - Risques et limites à l'utilisation d'*Ajax* - Motifs de conception *Ajax* - Être productif avec *Ajax*

Module DM102 : Développer des applications web pour *smartphone*
Durée : 3 jours Coût : 1 620 € HT
Framework de développement d'applications mobiles, *jQuery Mobile* offre un ensemble d'outils basés sur les technologies du *web* permettant de créer des applications qui présentent la particularité non négligeable de fonctionner sur toutes les plateformes mobiles actuelles (*iPhone*, *Android*, *BlackBerry*, *Windows Phone*...). Autre particularité, le développement avec le *framework* est axé sur des fonctionnalités riches mais basé sur un code léger. En conséquence, *jQuery Mobile* est idéal pour concevoir rapidement des applications évoluées. Les participants à cette formation apprendront à développer des applications mobiles en tirant parti de la richesse du *framework*.
Objectifs
- Maîtriser le *framework* jQuery Mobile - Identifier les caractéristiques relatives à l'ergonomie des applications mobiles - Disposer des compétences nécessaires au développement d'applications mobiles avec *jQuery Mobile*
Contenu
- Les fondements de *jQuery Mobile* - L'ergonomie des applications mobiles - Les composants de *jQuery Mobile* - Les API - Autour de *jQuery Mobile*

D'après le site www.ib-formation.fr