

Livrable EQUIDA

Mai 2019

(Valentin, Maxence, Paul, Minh Kiêu)

Table des matières

A. Présentation

- I. Contexte
- II. Cahier des charges
- III. Outils utilisés

B. Manuel d'utilisation

- I. Maquettes
- II. UML
- III. Guide d'interface web

C. Manuel d'évolution technique

- I. Schémas de modélisation
- II. Architecture fichiers du site
- III. Fonctionnements techniques

A. Présentation

I. Contexte

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.

II. Cahier des charges

La société Equida nous a mandaté afin de réaliser un site web. Son utilité au sein de la société est de permettre la gestion des ventes aux enchères de chevaux.

Cette application unique permet de gérer :

- Les informations sur les chevaux.
- La consultation du catalogue des ventes.
- La consultation des résultats d'une vente.
- L'ajout d'une enchère.

Cette application contient également les paramètres suivant :

- Une authentification préalable nécessaire pour l'accès au contenu par certains utilisateurs selon leurs droits.
-

III. Outils utilisés

Le site est développé en JAVA, avec J2EE.

L'application est déployé dans un environnement Apache Tomcat, sous Debian.

La base de données est géré par MySQL.

Pour la gestion des versions, l'outil Git est utilisé avec un dépôt Github qui se trouve à l'adresse :

Pour l'interface graphique, nous avons utilisé Materialize.

L'UML et le MCD ont été réalisé sous WinDesign.


B. Manuel d'utilisation

I. Maquette

Accueil

Barre de Navigation

Bienvenue sur Equida!



Nouveaux chevaux	Ventes à venir
Nom cheval race prix	Nom vente catégorie de vente date de début d'inscription
Nom cheval race prix	Nom vente catégorie de vente date de début d'inscription
Nom cheval race prix	Nom vente catégorie de vente date de début d'inscription
Nom cheval race prix	Nom vente catégorie de vente date de début d'inscription
Nom cheval race prix	Nom vente catégorie de vente date de début d'inscription
Plus d'infos	Plus d'infos

Pied de page

Ajouter un client

Barre de Navigation

Nouveau client

Nom :

Prénom :

Adresse :

Code postal :

Ville :

Pays :
Angleterre

Catégorie Vente :

Valider

Pied de page

© 2018 Copyright

Consultation d'un cheval

Barre de Navigation

Nom Cheval

Race :
Sexe :
Sire :
Mère : Sire mère
Père : Sire père
Prix départ (si non vendu) / Vendu à (si vendu) :

Les courses

Aucune course n'a été enregistré pour ce cheval
ou

Nom course	classement	année
Nom course	classement	année
Nom course	classement	année

Infos Vente

Si ni vendu ni en vente

Ce cheval n'est pas encore en vente
ou
Ce cheval sera présent à cette vente [lien](#)
ou
Ce cheval a été vendu à X pour X € lors de cette vente [lien](#)

Modifier

seulement visible par le directeur général ou le propriétaire du cheval

Pied de page

© 2018 Copyright

Consultation d'un client

Barre de Navigation

Nom Client

adresse
code postal Ville
[adresse email](#)
pays
Catégorie de vente :
archivé

lien vers la boîte email

si archivé

Modifier

archiver

seulement visible par le directeur général ou le propriétaire du cheval
s'affiche si le client n'est pas encore archivé

Pied de page


© 2018 Copyright

Consultation d'une vente

Nom Vente


Lieu, nb boxe disponible
Categorie vente
Date debut d'inscription
Date de la vente

Modifier




seulement visible par le directeur général

Les Chevaux




Race :
Sexe :
Prix de départ :

VOIR PLUS



Race :
Sexe :
Prix de départ :

VOIR PLUS



Race :
Sexe :
Prix de départ :

VOIR PLUS

Liste des clients

Liste des clients

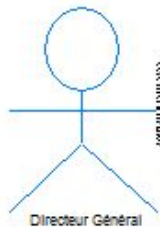
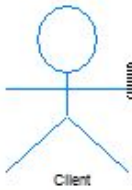
Nom	Prenom	Pays	Rue	Ville	Code Postal	Mail	
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier
nom	prénom	pays	adresse	ville	code postal	adresse email	Modifier

Liste des races des chevaux

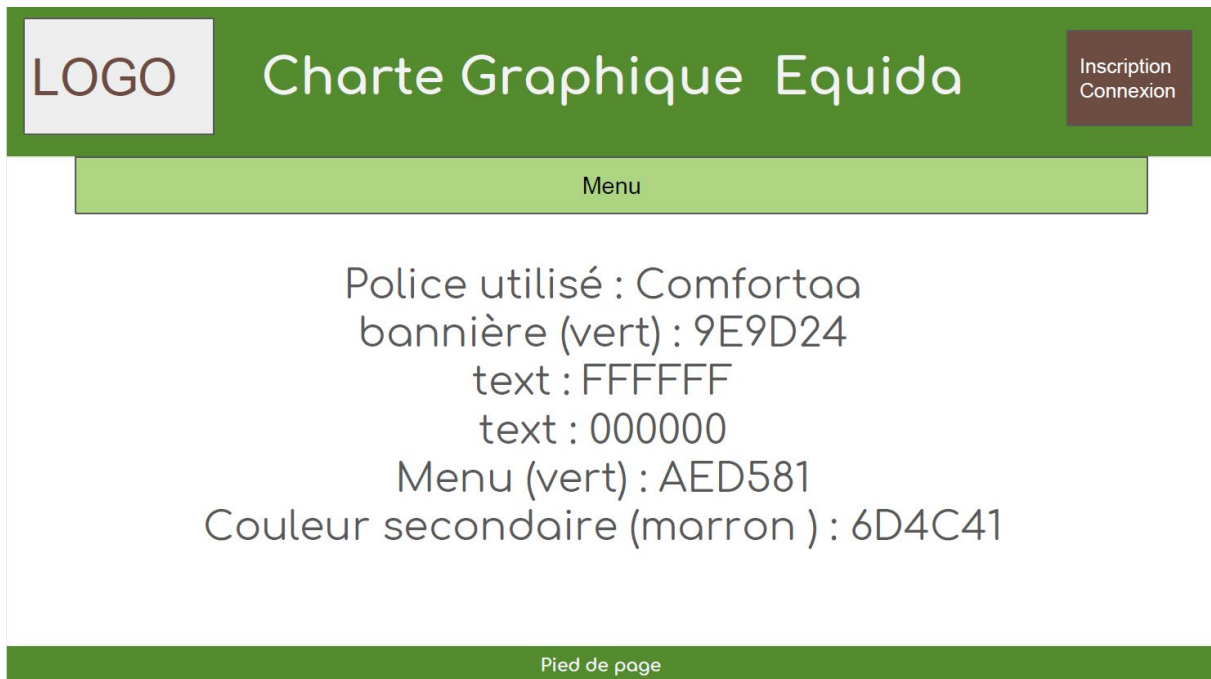
Liste des races de chevaux

Libelle	Description	Modifier
Arabo-frison	L'Arabo-frison est une race chevaline récente, sélectionnée sur plusieurs générations depuis les années 1960 pour obtenir la morphologie du Frison moderne associée aux qualités d'endurance du Pur-sang arabe.	Modifier
Lombok	Le Lombok (indonésien : Kuda lombok), est la race de poneys présente sur l'île de Lombok, en Indonésie.	Modifier
Halla	Le Halla est une race de chevaux de course originaire de Corée du Sud,	Modifier

II. UML

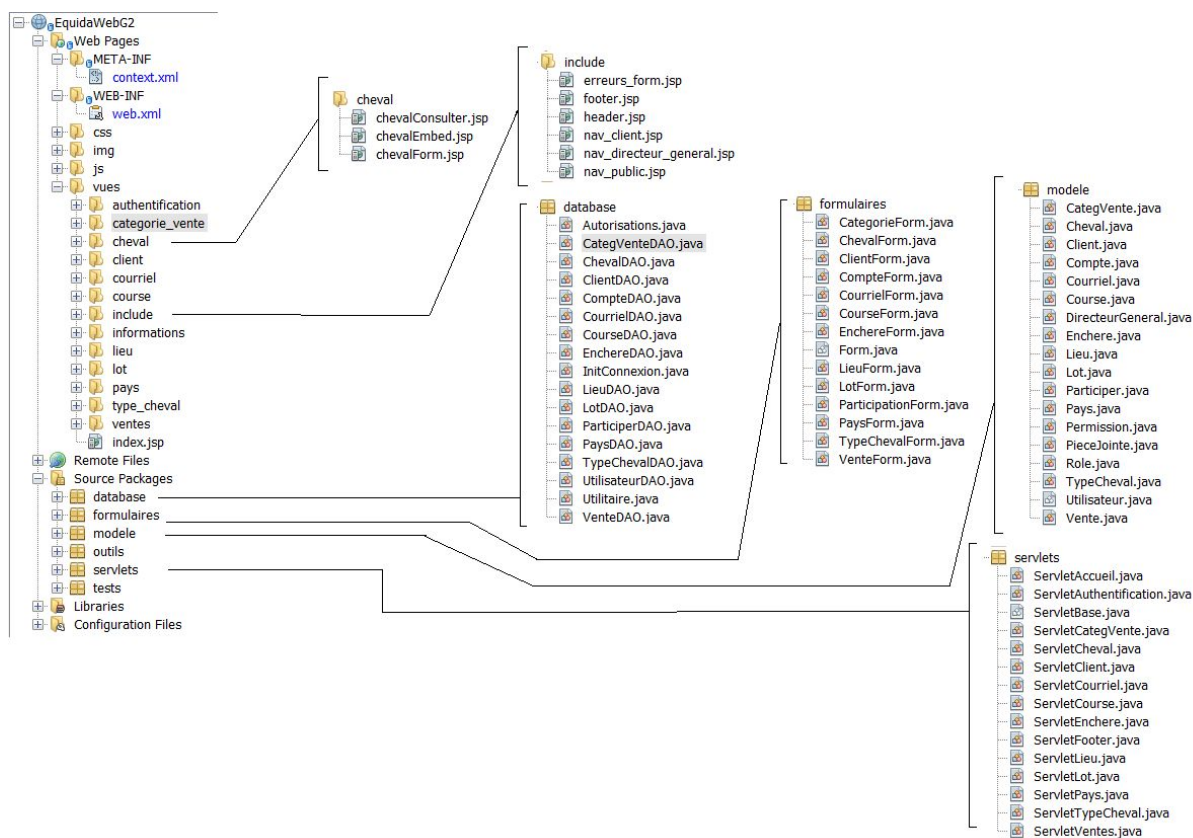


III. Interface Web





II. Architecture fichiers du site



Le dossier 'vues' contient tous les sous-dossiers des catégories où se trouvent les pages de présentation en format .jsp.

Le dossier 'database' contient tous les fichiers en format .java permettant de communiquer avec la base de données grâce à des méthodes.

Le dossier 'formulaires' contient tous les fichiers en format .java permettant l'ajout d'un objet à la base.

Le dossier 'modele' contient toutes les classes métiers .java.

Le dossier 'servlet' contient les contrôleurs permettant de tester les méthodes du dossier 'database' et de renvoyer les données vers les pages .jsp.

III. Fonctionnements techniques

1. Le listing de chevaux

a. Modèle

- Création et implémentations des classes métiers .java avec ses propriétés, ses constructeurs et ces getters et setters

```
package modele;

import java.util.ArrayList;

public class Cheval {

    private int id;
    private String nom;
    private boolean male;
    private String sire;
    private TypeCheval typeCheval;
    private ArrayList<Lot> lots;
    private Cheval pere;
    private Cheval mere;
    private Client client;

    public Cheval() {
        this(0, "", false, "", null, null, null, null);
    }

    public Cheval(int id, String nom, boolean male, String sire, TypeCheval typeCheval, ArrayList<Lot> lots, Cheval pere, Cheval mere) {
        this.id = id;
        this.nom = nom;
        this.male = male;
        this.sire = sire;
        this.typeCheval = typeCheval;
        this.lots = lots;
        this.pere = pere;
        this.mere = mere;

        if (this.lots == null) {
            this.lots = new ArrayList<>();
        }
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }
}
```

b. Vue

- Dans le fichier Lot/listerLots.jsp, boucle listant les lots (chevaux) en vente incluant la page chevalEmbed.jsp permettant de récupérer les informations importantes des chevaux.

listerLots.jsp

```
<%@page import="modele.Lot"%>
<%@page import="java.util.ArrayList"%>
<%@page import="modele.Utilisateur"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<jsp:include page="/vues/include/header.jsp" />

<%
    Utilisateur user = (Utilisateur) request.getSession().getAttribute("user");
    ArrayList<Lot> lots = (ArrayList<Lot>) request.getAttribute("pLots");
%>

<h2 class="center-align">Chevaux en vente</h2>

<div class="row">
    <div class="col s12">
        <%
            for (Lot lot : lots) {
                request.setAttribute("lot", lot);
            }
        %>
        <jsp:include page="/vues/cheval/chevalEmbed.jsp"/>
        <%
            }
        %>
    </div>
</div>

<jsp:include page="/vues/include/footer.jsp" />
```

chevalEmbed.jsp

```
<%@page import="modele.Utilisateur"%>
<%@page import="modele.Cheval"%>
<%@page import="java.util.ArrayList"%>
<%@page import="modele.Lot"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%
    Lot lot = (Lot) request.getAttribute("lot");
%>

<div class="col s12 m4 l3">
    <div class="card">
        <div class="card-image">
            
            <span class="card-title"><%= lot.getCheval().getNom() %></span>
        </div>
        <div class="card-content">
            <p>Race : <%= lot.getCheval().getTypeCheval().getLibelle() %></p>
            <p>Sexe : <%= (lot.getCheval().getMale()) ? "Male" : "Femelle" %></p>
            <p>Prix de départ : <%= lot.getPrixDepart() %>€</p>
        </div>
        <div class="card-action">
            <a href="<%= request.getContextPath() %>/ServletCheval/consulterCheval?id=<%= lot.getCheval().getId() %>">Voir plus</a>
        </div>
    </div>
</div>
```

- Dans web.xml, ajout de la route menant à cette page

```
<servlet>
    <servlet-name>ServletLot</servlet-name>
    <servlet-class>servlets.ServletLot</servlet-class>
</servlet>
```

c. Contrôleur

- Dans le fichier LotDAO.java, ajout de la méthode 'getLesLotsNonVendues'


```

public static ArrayList<Lot> getLesLotsNonVendu(Connection connection) {
    ArrayList<Lot> lesLots = new ArrayList<Lot>();
    try {
        //preparation de la requete
        PreparedStatement requete = connection.prepareStatement("SELECT * FROM lot, cheval WHERE lot.idCheval = cheval.id AND lot.id NOT IN (SELECT lot FROM encheres) "
            + "AND cheval.archiver = 0 AND lot.validation IS NOT NULL ORDER BY prixDepart DESC");

        //executer la requete
        ResultSet rs = requete.executeQuery();

        //On hydrate l'objet métier Lot avec les résultats de la requête
        while (rs.next()) {
            Lot unLot = new Lot();
            unLot.setId(rs.getInt("id"));
            unLot.setCheval(ChevalDAO.getCheval(connection, rs.getInt("idCheval")));
            unLot.setVente(VenteDAO.getVente(connection, rs.getInt("idVente")));
            unLot.setPrixDepart(rs.getFloat("prixDepart"));

            lesLots.add(unLot);
        }
    } catch (SQLException e) {
        e.printStackTrace();
        //out.println("Erreur lors de l'établissement de la connexion");
    }
    return lesLots;
}

```

- Dans le fichier ServletLot.java, dans la méthode doGet, ajout du lien de la méthode DAO vers la page de la vue .jsp

```

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    super.doGet(request, response);

    Utilisateur user = (Utilisateur) request.getSession().getAttribute("user");
    String url = request.getRequestURI();
    if (url.equals(URL_LISTER_LOTS)) {
        ArrayList<Lot> lesLots = LotDAO.getLesLotsNonVendu(connection);

        request.setAttribute("pLots", lesLots);
        changerTitrePage("Lister les lots", request);

        getServletContext().getRequestDispatcher("/vues/lot/listerLots.jsp").forward(request, response);
    }
}

```

d. Résultat

The screenshot shows the Equida web application interface. At the top, there's a green header with the Equida logo and navigation links: "Consulter les chevaux" and "Consulter les ventes". The main content area is titled "Chevaux en vente" and displays three horse listings. Each listing includes a photo of a horse, its name, race, sex, and price of departure, along with a "VOIR PLUS" button. The footer is green and contains links for "Qui sommes-nous?", "Mentions légales", and "Contact", as well as a copyright notice for 2019.

2. La consultation d'un cheval

a. Vue

- Dans le fichier ChevalEmbed.jsp, mise en place d'un lien permettant la consultation plus précise d'un cheval.

```
<div class="card-action">
  <a href="<%= request.getContextPath() %>/ServletCheval/consulterCheval?id=<%= lot.getCheval().getId() %>">Voir plus</a>
</div>
```

- Dans le fichier ChevalConsulter.jsp, mise en page des données pour la consultation des informations d'un cheval

```
<jsp:include page="/vues/include/header.jsp" />

<%
    Utilisateur user = (Utilisateur) request.getSession().getAttribute("user");
    Cheval unCheval = (Cheval) request.getAttribute("pCheval");
    ArrayList<Participer> lesParticipations = (ArrayList<Participer>) request.getAttribute("pParticipations");
    ArrayList<Enchere> lesEncheres = (ArrayList<Enchere>) request.getAttribute("pEncheres");
    ArrayList<Client> encherisseur = (ArrayList<Client>) request.getAttribute("pClients");
    Lot lot = (Lot) request.getAttribute("pLot");
%>

<div class="row">
  <div class="col s12 s16 right valign-wrapper">
    
  </div>

  <div class="col s12 s16 left">
    <h2><%= unCheval.getNom() %></h2>
    <p class="tooltip" data-position="bottom" data-tooltip="<%= unCheval.getTypeCheval().getDesc() %>">Race : <%= unCheval.getTypeCheval().getLibelle() %></p>
    <p>Sexe : <%= (unCheval.getMale() != null) ? "Male" : "Femelle" %></p>
    <p>Sire : <%= (unCheval.getSire() != null) ? unCheval.getSire().getId() + " " + unCheval.getSire().getSire() + "</a>" : "Non renseigné" %></p>
    <p>Mère : <%= (unCheval.getMere() != null) ? "<a href=?id=?> unCheval.getMere().getId() + ">" + unCheval.getMere().getSire() + "</a>" : "Non renseigné" %></p>
    <p>Père : <%= (unCheval.getPere() != null) ? "<a href=?id=?> unCheval.getPere().getId() + ">" + unCheval.getPere().getSire() + "</a>" : "Non renseigné" %></p>

    <%
        if (user instanceof DirecteurGeneral) {
            out.println("<p><a href=?> ServletCheval.URL_ARCHIVER_CHEVAL + "?id=?> unCheval.getId() + "?>");
            out.println("Archiver");
            out.println("</a></p>");

            if (lot != null) {
                if (lot.getValidation() == null && lot.getVente() != null) {
                    out.println("<p>Voulez-vous valider le cheval afin de l'ajouter à la vente ?");
                    out.println("<p><a href=?> ServletCheval.URL_VALIDER_CHEVAL + "?id=?> lot.getId() + "?>");
                    out.println("Valider");
                    out.println("</a></p>");
                }
            }
        }
    %>
  </div>
</div>

<div class="row">
```

- Dans le fichier 'web.xml', mise en place de la route

```
<servlet-mapping>
  <servlet-name>ServletCheval</servlet-name>
  <url-pattern>/ServletCheval/consulterCheval</url-pattern>
</servlet-mapping>
```

b. Contrôleur

- Dans le fichier 'ChevalDAO.java', implémentation de la méthode de consultation d'un cheval avec la requête de lecture des données d'un cheval en prenant son Id en paramètre

```

public static Cheval getCheval(Connection connection, int idCheval) {
    Cheval unCheval = null;

    try {
        //preparation de la requete
        PreparedStatement requete = connection.prepareStatement("SELECT * FROM cheval WHERE id=?;");
        requete.setInt(1, idCheval);

        //executer la requete
        ResultSet rs = requete.executeQuery();

        //On hydrate l'objet métier Cheval avec les résultats de la requête
        while (rs.next()) {
            unCheval = new Cheval();
            unCheval.setId(idCheval);
            unCheval.setNom(rs.getString("nom"));
            unCheval.setSire(rs.getString("sire"));
            unCheval.setMale(rs.getBoolean("sexe"));

            int typeCheval = rs.getInt("typeCheval");
            if (typeCheval != 0) {
                unCheval.setTypeCheval(TypeChevalDAO.getTypeCheval(connection, typeCheval));
            }

            int idPere = rs.getInt("pere");
            if (idPere != 0) {
                unCheval.setPere(getCheval(connection, idPere));
            }

            int idMere = rs.getInt("mere");
            if (idMere != 0) {
                unCheval.setMere(getCheval(connection, idMere));
            }

            unCheval.setClient(ClientDAO.getClient(connection, rs.getInt("client")));
        }
    } catch (SQLException e) {
        e.printStackTrace();
        //out.println("Erreur lors de l'établissement de la connexion");
    }
    return unCheval;
}

```

➤ Dans le fichier 'ServletCheval.java', mise en relation de la méthode de consultation d'un cheval vers la page Jsp.

```

if (url.equals(URL_CONSULTER_CHEVAL)) {
    int idCheval = 0;
    try {
        idCheval = Integer.valueOf(request.getParameter("id"));
    } catch (Exception e) {
        redirigerVersAccueil(response);
        return;
    }

    Cheval cheval = ChevalDAO.getCheval(connection, idCheval);
    ArrayList<Participer> lesParticipations = ParticiperDAO.getLesParticipationsCoursesCheval(connection, idCheval);
    ArrayList<Enchere> lesEncheres = EnchereDAO.getLesEncheres(connection, idCheval);
    Lot lot = LotDAO.getLotCheval(connection, idCheval);
    ArrayList<Client> lesClients = ClientDAO.getLesClientsPrDirGen(connection);

    request.setAttribute("pClients", lesClients);
    request.setAttribute("pLot", lot);
    request.setAttribute("pEncheres", lesEncheres);

    request.setAttribute("pParticipations", lesParticipations);
    request.setAttribute("pCheval", cheval);
    changerTitrePage("Cheval " + cheval.getNom(), request);

    getServletContext().getRequestDispatcher("/vues/cheval/chevalConsulter.jsp").forward(request, response);
}

```

c. Résultat

Coquelicot

Race : Lombok

Sexe : Male

Sire : 1243768956

Mère : Non renseignée

Père : Non renseigné



Les courses

Aucune course n'a été enregistrée pour ce cheval

Informations sur la vente

Ce cheval n'a pas été vendu.

3. L'ajout d'un cheval

a. *Modèle*

- Dans le fichier ChevalForm.java, implémentation des méthodes de formulaire d'ajout d'un cheval

```
public class ChevalForm extends Form {  
  
    //méthode de validation du champ de saisie nom  
    private void validationNom(String nom) throws Exception {  
        if (nom != null && nom.length() < 3) {  
            throw new Exception("Le nom d'utilisateur doit contenir au moins 3 caractères.");  
        }  
    }  
  
    public Cheval getCheval(HttpServletRequest request, Connection connection) {  
        Cheval unCheval = new Cheval();  
  
        String nomChampId = "id";  
        String nomChampNom = "nom";  
        String nomChampSexe = "sexe";  
        String nomChampSire = "sire";  
        String nomChampTypeCheval = "typeCheval";  
        String nomChampPere = "pere";  
        String nomChampMere = "mere";  
  
        String idChevalStr = getDataForm(request, nomChampId);  
        String nom = getDataForm(request, nomChampNom);  
        boolean male = Integer.valueOf(getDataForm(request, nomChampSexe)) == 1;  
        String sire = getDataForm(request, nomChampSire);  
        int idTypeCheval = Integer.valueOf(getDataForm(request, nomChampTypeCheval));  
        String pereSire = getDataForm(request, nomChampPere);  
        String mereSire = getDataForm(request, nomChampMere);  
  
        if (idChevalStr != null) {  
            unCheval.setId(Integer.valueOf(idChevalStr));  
        }  
  
        if (nom == null) {  
            ajouterErreur(nomChampNom, "Le champ nom est obligatoire.");  
        } else {  
            if (nom.length() < 3 || nom.length() > 50) {  
                ajouterErreur(nomChampNom, "La longueur du nom doit être comprise entre 3 et 50 caractères.");  
            }  
        }  
  
        if (sire == null) {  
            ajouterErreur(nomChampSire, "Le champ SIRE est obligatoire.");  
        } else {  
            // Validation de la longueur du SIRE (non présente dans l'image)  
        }  
    }  
}
```

b. *Vue*

- Dans le fichier 'ChevalForm.jsp', ajout de la forme du formulaire d'ajout des informations d'un cheval

```

<%
    //On doit afficher les informations du cheval dans les inputs si "unCheval" est différent de null.
    //On doit également changer l'action selon que "unCheval" soit null ou non.
    //De manière générale, si le "unCheval" est null alors on en ajoute un. Sinon on le modifie.
    ArrayList<TypeCheval> lesTypeCheval = (ArrayList) request.getAttribute("pLesTypeCheval");
    Cheval unCheval = (Cheval) request.getAttribute("pCheval");

    ChevalForm form = null;
    try {
        form = (ChevalForm) ServletBase.getForm(request);
    } catch (ClassCastException e) {
    }

    request.setAttribute("form", form);
%>

<jsp:include page="/vues/include/header.jsp" />
<h2 class="center-align"><%= (unCheval == null) ? "Nouveau" : "Modifier un" %> cheval</h2>
<jsp:include page="/vues/include/erreurs_form.jsp" />
<div class="row">
    <form class="col s10 push-s1 l9 push-l2 center-align" action="<%= (unCheval == null) ? "ajouterCheval" : "chevalModifier" %>" method="POST">
        <input type="hidden" name="id" value="<%= (unCheval != null) ? unCheval.getId() : "" %>" />
        <div class="row">
            <div class="input-field col l6 s12">
                <input id="nom" type="text" name="nom" size="30" minlength="3" maxlength="30" value="<%= (unCheval != null) ? unCheval.getNom() : "" %>" class="validate">
                <label for="nom">Nom : </label>
            </div>
            <div class="input-field col l6 s12">
                <select id="sexe" name="sexe">
                    <option value="0"><
                        if (unCheval != null) {
                            if (!unCheval.getMale()) {
                                out.print("selected");
                            }
                        }
                    %>Femelle</option>

```

➤ Dans le fichier 'web.wml', ajout de la route

```

<servlet-mapping>
    <servlet-name>ServletCheval</servlet-name>
    <url-pattern>/ServletCheval/ajouterCheval</url-pattern>
</servlet-mapping>

```

c. Contrôleur

➤ Dans le fichier 'ChevalDAO.java', ajout de la méthode d'ajout d'un cheval

```

// Methode ajouterCheval
public static int ajouterCheval(Connection connection, Cheval unCheval, HttpServletRequest request) {
    int idGenere = -1;
    try {
        //preparation de la requete
        // id (clé primaire de la table client) est en auto_increment, donc on ne renseigne pas cette valeur
        // la paramètre RETURN_GENERATED_KEYS est ajouté à la requête afin de pouvoir récupérer l'id généré par la bdd (voir ci-dessous)
        // supprimer ce paramètre en cas de requête sans auto_increment.
        PreparedStatement requete = connection.prepareStatement("INSERT INTO cheval ( nom, sexe, sire, typeCheval, pere, mere, client)\n"
            + "VALUES (?, ?, ?, ?, ?, ?)", PreparedStatement.RETURN_GENERATED_KEYS);
        requete.setString(1, unCheval.getNom());
        requete.setInt(2, unCheval.getMale() ? 1 : 0);
        requete.setString(3, unCheval.getSire());
        requete.setInt(4, unCheval.getTypeCheval().getId());
        if (unCheval.getPere() != null) {
            requete.setInt(5, unCheval.getPere().getId());
        } else {
            requete.setNull(5, Types.INTEGER);
        }

        if (unCheval.getMere() != null) {
            requete.setInt(6, unCheval.getMere().getId());
        } else {
            requete.setNull(6, Types.INTEGER);
        }

        Utilisateur user = (Utilisateur) request.getSession().getAttribute("user");
        if (user != null) {
            requete.setInt(7, user.getId());
        }
        /* Exécution de la requête */
        requete.executeUpdate();

        // Récupération de id auto-généré par la bdd dans la table cheval
        ResultSet rs = requete.getGeneratedKeys();
        while (rs.next()) {
            idGenere = rs.getInt(1);
            unCheval.setId(idGenere);
        }
    }
}

```


- Dans le fichier 'ServletCheval.java', ajout du lien de la méthode vers la page jsp dans la méthode doGet

```
if (url.equals(URL_AJOUTER_CHEVAL)) {
    if (user instanceof Client) {
        ArrayList<TypeCheval> lesTypeCheval = TypeChevalDAO.getLesTypeCheval(connection);

        request.setAttribute("pLesTypeCheval", lesTypeCheval);
        changerTitrePage("Ajouter un cheval", request);

        this.getServletContext().getRequestDispatcher("/vues/cheval/chevalForm.jsp").forward(request, response);
    } else {
        redirigerVersAccueil(response);
    }
}
```

- Dans le fichier 'ServletCheval.java', ajout du lien de la méthode vers la page jsp dans la méthode doPost

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    super.doPost(request, response);

    Utilisateur user = (Utilisateur) request.getSession().getAttribute("user");
    String url = request.getRequestURI();
    if (url.equals(URL_AJOUTER_CHEVAL)) {
        if (user instanceof Client) {
            /* Préparation de l'objet formulaire */
            ChevalForm form = new ChevalForm();

            /* Appel au traitement et à la validation de la requête, et récupération du bean en résultant */
            Cheval unCheval = form.getCheval(request, connection);

            if (form.getErreurs().isEmpty()) {
                int idCheval = ChevalDAO.ajouterCheval(connection, unCheval, request);
                response.sendRedirect(URL_CONSULTER_CHEVAL + "?id=" + idCheval);
            } else {
                request.getSession().setAttribute("form", form);
                response.sendRedirect(URL_AJOUTER_CHEVAL);
            }
        } else {
            redirigerVersAccueil(response);
        }
    }
}
```

d. Résultat

Nouveau cheval

Sexe :	
Nom :	Femelle ▼
SIRE :	
Race:	
Arabo-frison	▼
Mère :	Père :
<input type="button" value="Valider"/>	

