Projet CAE

Spécifications techniques

CANZERINI Mathieu - EDDASSER Omar - GERARD Florent - TORRES Hadrien

Table des matières

1. Contexte

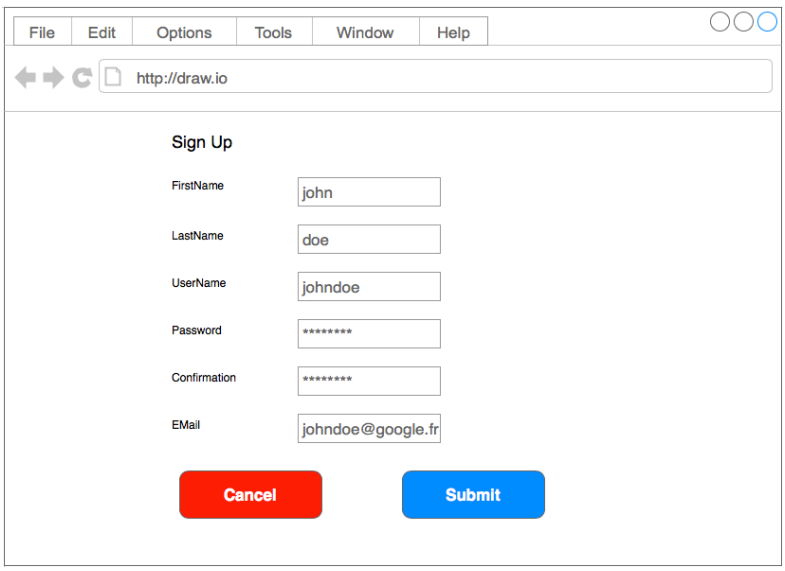
L'application réalisée est une **plate-forme d'achat d'articles de presse**. Son rôle est de mettre en relation des visiteurs/clients et des journalistes sous la surveillance de rédacteurs en chef et d'administrateurs.

2. Description générale

L'application réalisée doit permettre à des visiteurs de s'inscrire sur notre site afin d'avoir accès à des articles rédigés par un ensemble de rédacteurs présents sur notre site (pré-visualisation d'article, achat d'articles ou lots d'articles). Elle doit également permettre à des rédacteurs en chef de gérer (contrôler, valider, supprimer) les articles rédigés et soumis par les rédacteurs afin qu'ils puissent être proposé sur notre plateforme. Enfin, l'application doit permettre à des administrateurs de gérer les différents comptes utilisateurs de l'application.

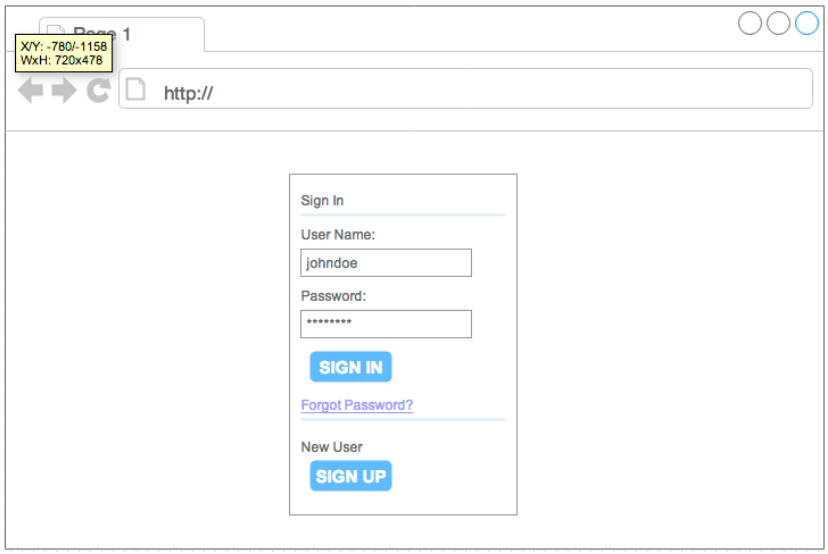
3. Fonctions de service

S'inscrire



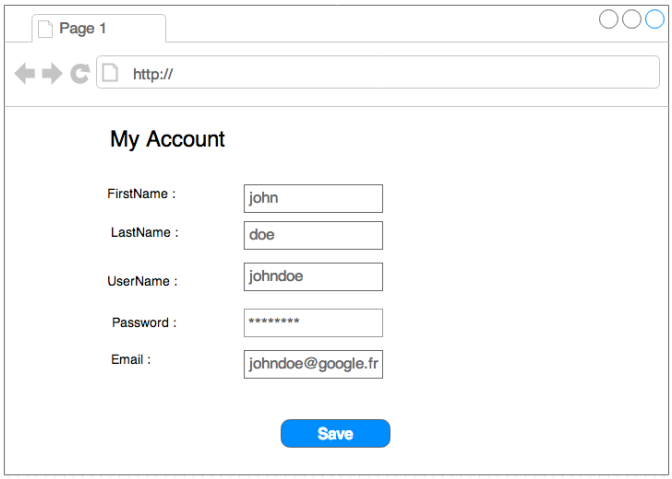
La fonction permet à un visiteur non enregistré de **s'inscrire sur le site**. Via un formulaire, l'utilisateur entre son nom, prénom, mail et mot de passe. L'utilisateur doit saisir son mot de passe une seconde fois afin de limiter les erreurs de saisies lors de l'inscription. **Tous les champs sont requis** et le mail lui servira de connexion au moment de la connexion. Pour des questions de sécurité, le **mot de passe entré sera crypté en base** en utilisant l'algorithme MD5 et on ajoutera un *"Salt"* pour empêcher le décryptage du mot de passe haché.

Se connecter



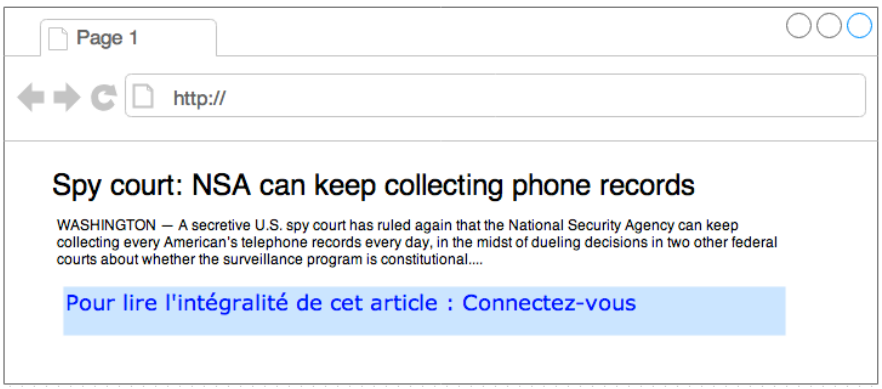
La fonction permet à un visiteur enregistré mais non authentifié de **se connecter à l'application**. Les données en entrée sont fournies grâce à un formulaire de connexion. Ces données sont **le mail (qui sert d'identifiant) et le mot de passe.** Lors de la connexion, l'application va contrôler en base la concordance de l'identifiant et du mot passe préalablement haché et modifié avec notre "*Salt*". Si la concordance est trouvée, on connecte l'utilisateur (on mémorise l'utilisateur en variable de session) et on lui propose alors les actions disponibles aux lecteur enregistrés.

Modifier les informations



Via un formulaire, un utilisateur enregistré **peut modifier son profil**. Il peut modifier son nom, son prénom, son mail et son mot de passe. Une fois les nouvelles données entrées, il valide le formulaire qui met à jour ses informations en base et l'utilisateur est redirigé vers son nouveau profil.

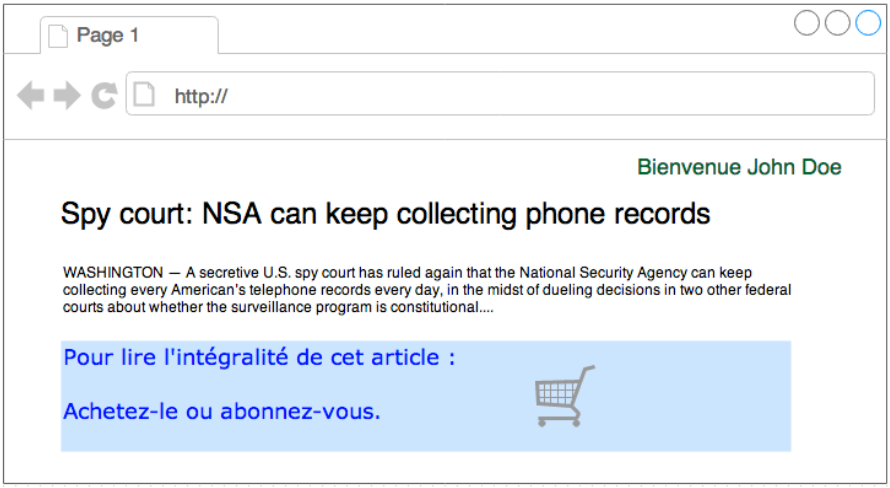
Pré-visualiser un article



L'application permet à un utilisateur de **choisir via un clic souris un article parmi les articles** disponibles

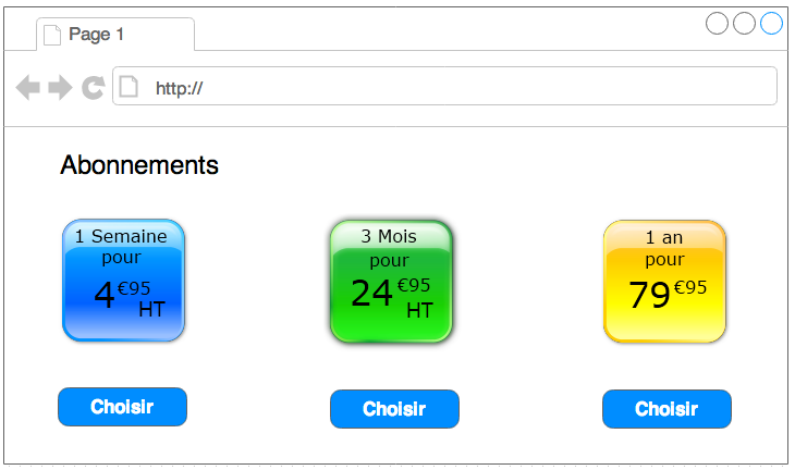
- Si l'utilisateur n'est pas connecté, il sera redirigé vers une page contenant l'article tronqué et il lui sera proposé de se connecter ou de s'inscrire au site.
- Si l'utilisateur est connecté, l'application va contrôler si cet l'article a déjà été acheté par ce lecteur ou si celui est abonné au dossier de cet article. Si il n'est pas autorisé à acheter cet article, il sera redirigé vers une page contenant une sous partie de l'article et il lui sera proposé d'acheter l'article ou de s'abonner.

Acheter un article



Si un utilisateur a décidé d'acheter un article à l'unité, il doit **valider son achat via un bouton** de confirmation. C**ette action est irréversible** et l'argent sera débité. On le redirige alors vers l'article complet qu'il a acquis.

S'abonner



L'utilisateur a la possibilité d**e souscrire à un abonnement** lui permettant d'avoir accès a tous les articles de la plate-forme. **Différentes durée d'abonnement** lui sont proposé avec différents avantages en fonction de la durée de souscription.

Une fois l'abonnement choisi, une redirection vers un module de paiement sécurisé s'effectue et permet de terminer son achat.

L'achat terminé, une redirection s'effectue vers la page préférence de l' utilisateur afin qu'il puisse constater que le site a bien pris en compte son abonnement.

Se déconnecter

Un utilisateur authentifié peut se déconnecter à tout moment en cliquant simplement sur un lien présent sur toutes les pages de l'application.

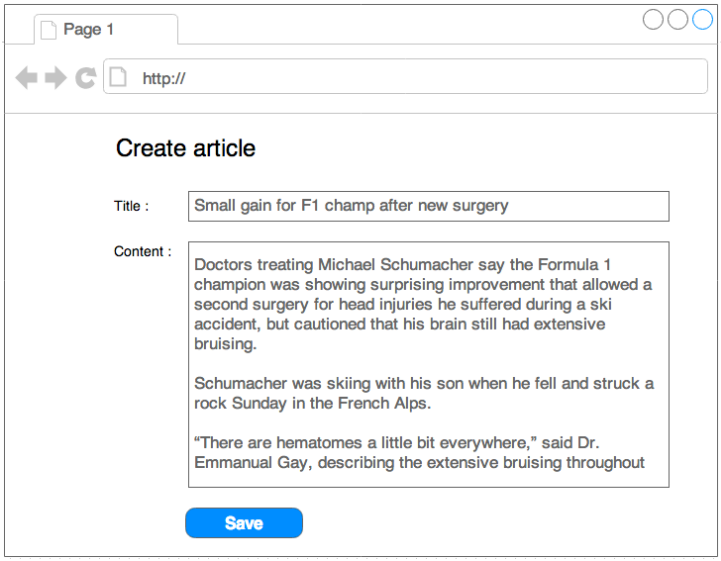
Consulter un article

Un utilisateur ayant acheté des articles, il lui est alors possible de les consulter a partir d'une vue listant ses différents achats. Il lui suffi alors de cliquer sur le titre d'un article pour afficher ce dernier.

Se désabonner

L'utilisateur a la possibilité de mettre fin à son inscription a tout moment afin d'empêcher la reconduction automatique de son abonnement. Pour cela il a à sa disposition un bouton dans l'interface récapitulant ses préférences.

Écrire un article



Un journaliste peut, de manière très simple, écrire des article grâce à une interface ou il peut saisir un titre et un contenu pour l'article. Il peut ensuite sauvegarder son article grâce a un bouton sur cette même interface.

Modifier un article

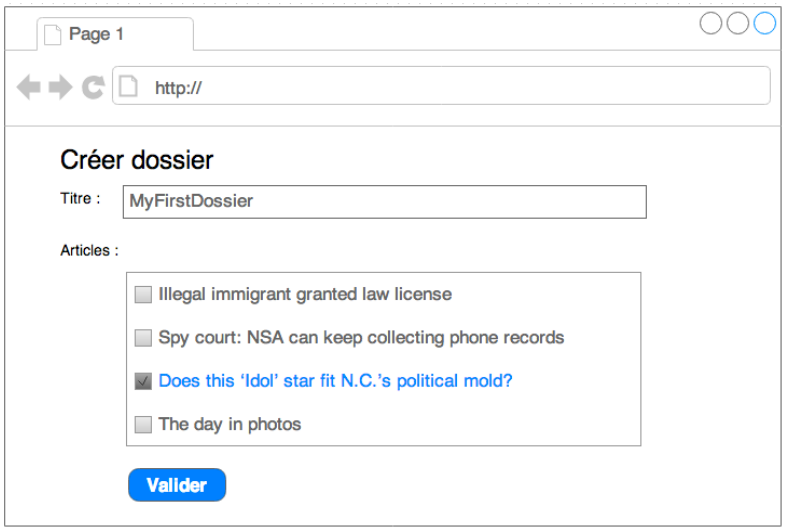
Un journaliste a accès à une vue récapitulant les différents articles qu'il a enregistré. Il peut effectuer une recherche grâce a un champ de saisie afin de retrouver rapidement un article dans la liste de ses articles enregistrés.

Il peut, à partir de cette liste, modifier un article qu'il a écrit. Il se retrouve alors sur la vue de rédaction de l'article avec les champs pré remplis du contenu déjà enregistré par le journaliste.

Soumettre un article

Depuis la même vue, l'utilisateur peut également, lorsqu'il estime que son article est terminé, le soumettre à l'approbation d'un rédacteur en chef grâce à un simple bouton en face du titre de l'article.

Définir un dossier

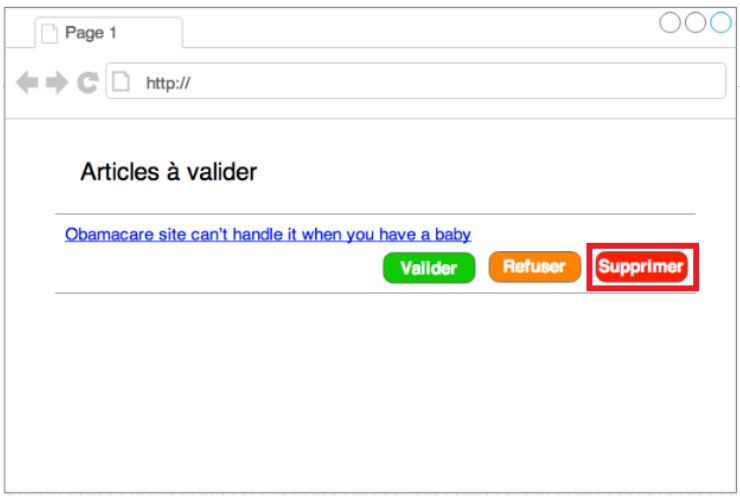


Un rédacteur en chef a la possibilité de créer via une interface un ensemble d'article qu'il pourra soumettre à la vente. Pour cela il lui suffi de saisir un titre pour le dossier et de cocher dans la liste des articles tous les articles qu'il souhaite intégrer au dossier. Il peut ensuite valider sa création grâce à un bouton juste en dessous de la liste des articles.

Valider un article

Un rédacteur en chef a également à sa disposition une vue listant les différents articles soumis par les journalistes. Il peut **examiner ces articles** en clinquant simplement sur le titre de chacun des articles. En dessous de chaque article, trois boutons permettent au rédacteur en chef de v**alider, refuser ou supprimer un article**. En cas de refus, le rédacteur en chef doit motiver son refus avec un message qui serra transmis au journaliste qui a rédigé l'article en cause.

Supprimer un article



Un rédacteur en chef peut également **supprimer un article** si son contenu contient des propos ne correspondant pas à la charte de déontologie signée par les journalistes lors de leur ajout au site (en cas de propos racistes ou à caractère discriminatoire par exemple), ou si ceux-ci ne respectent pas les conditions générales d'utilisations. Le rédacteur en chef devra alors également motiver son refus avec un message.

4. Contraintes

4.1. Fonctionnelles

L'établissement des spécifications fonctionnelles et technique se déroulera du 06/01/2014 au 20/01/2014 et soumettra à la validation du client un dossier de spécification fonctionnelles détaillées ainsi qu'un dossier des spécifications techniques détaillées. La date de livraison du livrable final du projet est fixé au 24/02/2014. A cette date aura lieu une présentation finale du projet au client.

4.2. Organisationnelles

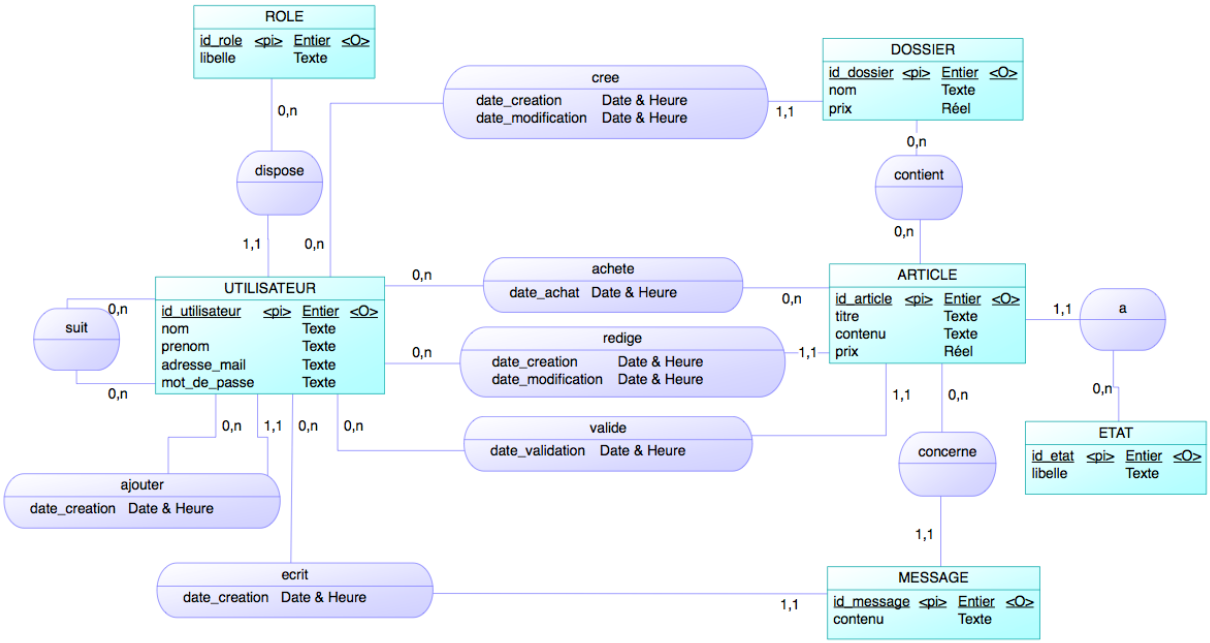
L' équipe de développement de se projet se compose de 4 ressources travaillant à plein temps sur le projet durant toute sa durée (du 06/01/2014 au 24/02/2014)

4.3. Techniques

Cette application mettra en oeuvre coté serveur l'architecture de composant logiciel suivant :
Entreprise Java Bean 3.0 (EJB)

5. Description des données

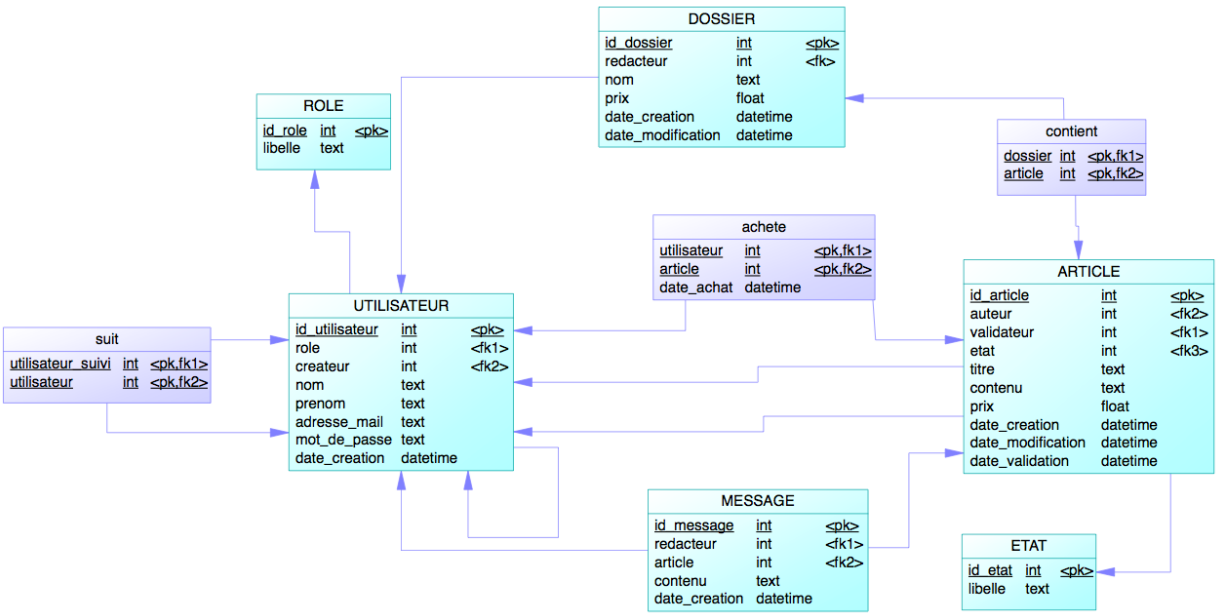
5.1. Modèle conceptuel des données (MCD)



5.2. Modèle logique des données (MLD)

CONTIENT(**#id_article**, **#id_dossier**)
MESSAGE(**id_message**, contenu, #id_utilisateur, date_creation, #id_article)
UTILISATEUR(**id_utilisateur**, nom, prenom, adresse_mail, mot_de_passe, date_creation, #id_role)
ROLE(**#id_role**, libelle)
ACHETE(**#id_utilisateur**, #id_article, date_achat)
SUIT(**#id_utilisateur**, **#id_utilisateur**)
ARTICLE(**id_article**, titre, contenu, prix, #id_utilisateur, date_creation, date_modification, #id_utilisateur, date_validation, #id_etat)
ETAT(**id_etat**, libelle)
DOSSIER(**id_dossier**, nom, prix, #id_utilisateur, date_creation, date_modification)

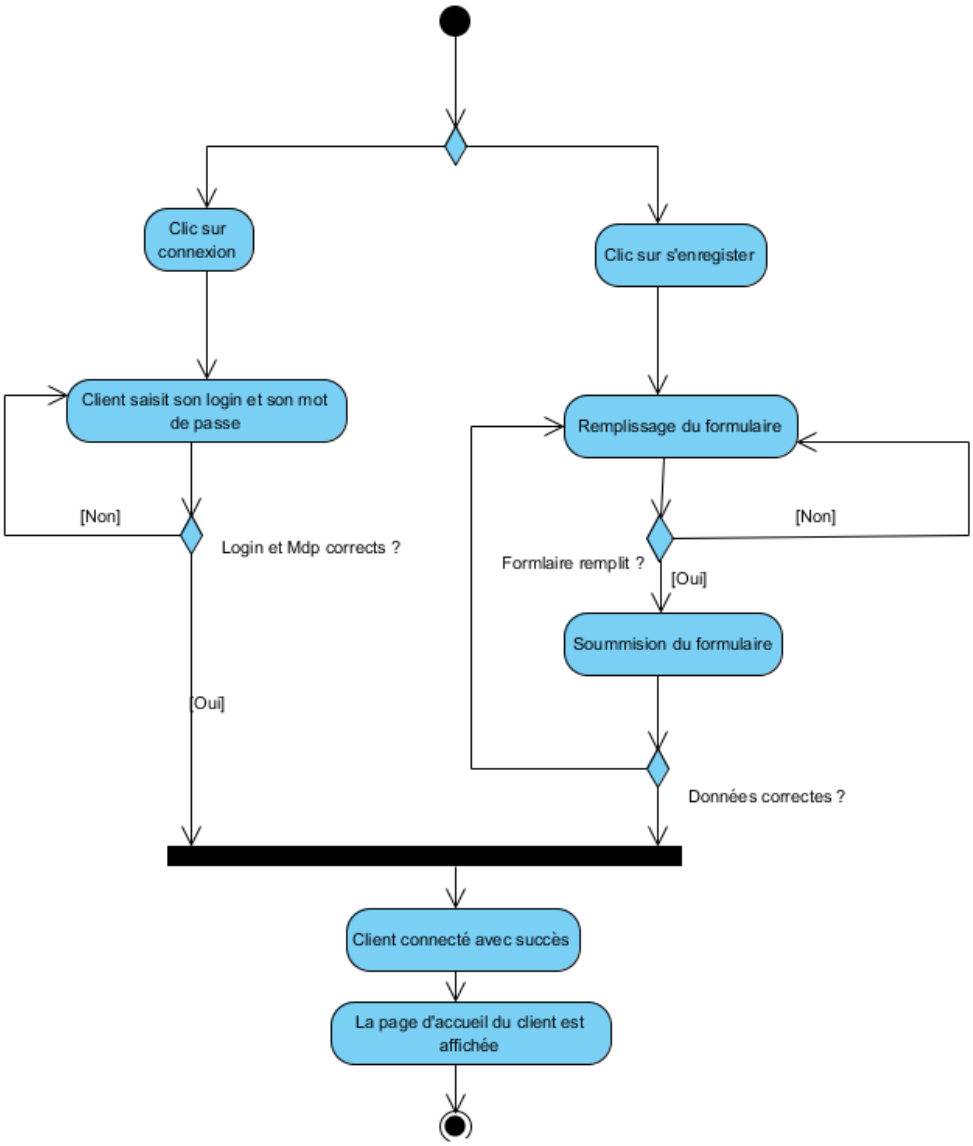
5.3. Modèle physique des données (MPD)



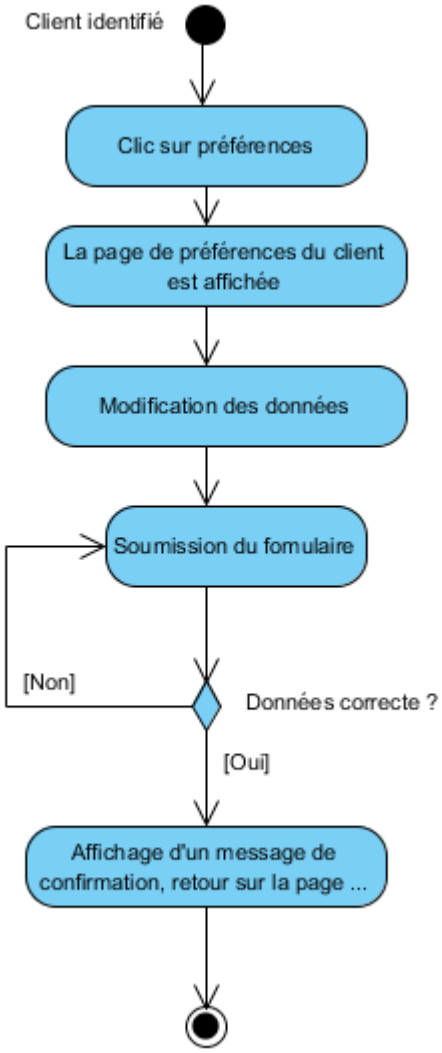
6. Description des traitements

6.1. Diagrammes d'activités

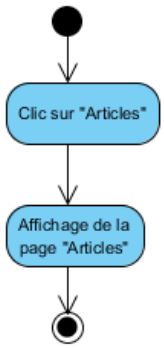
S'inscrire / se connecter



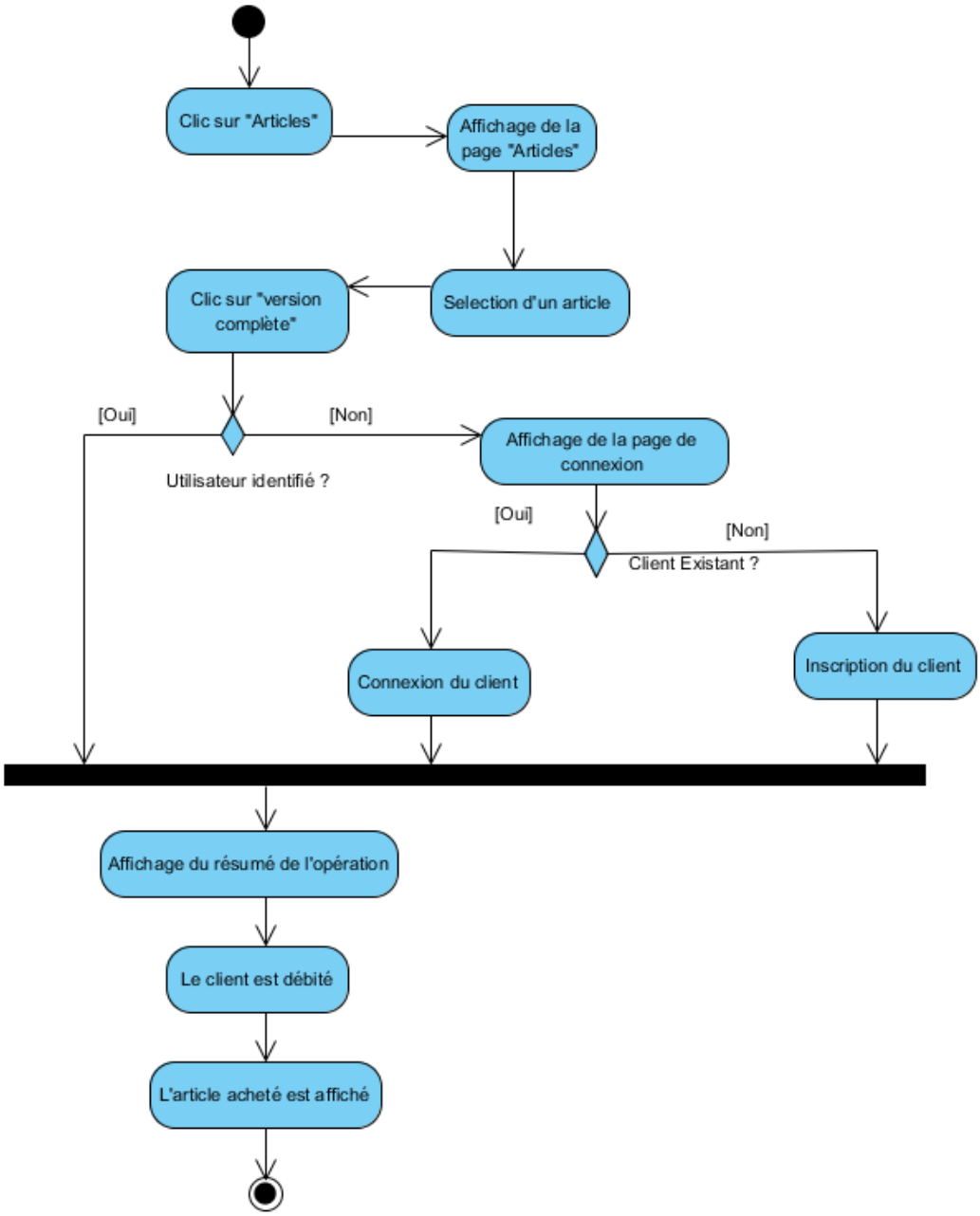
Modifier les informations



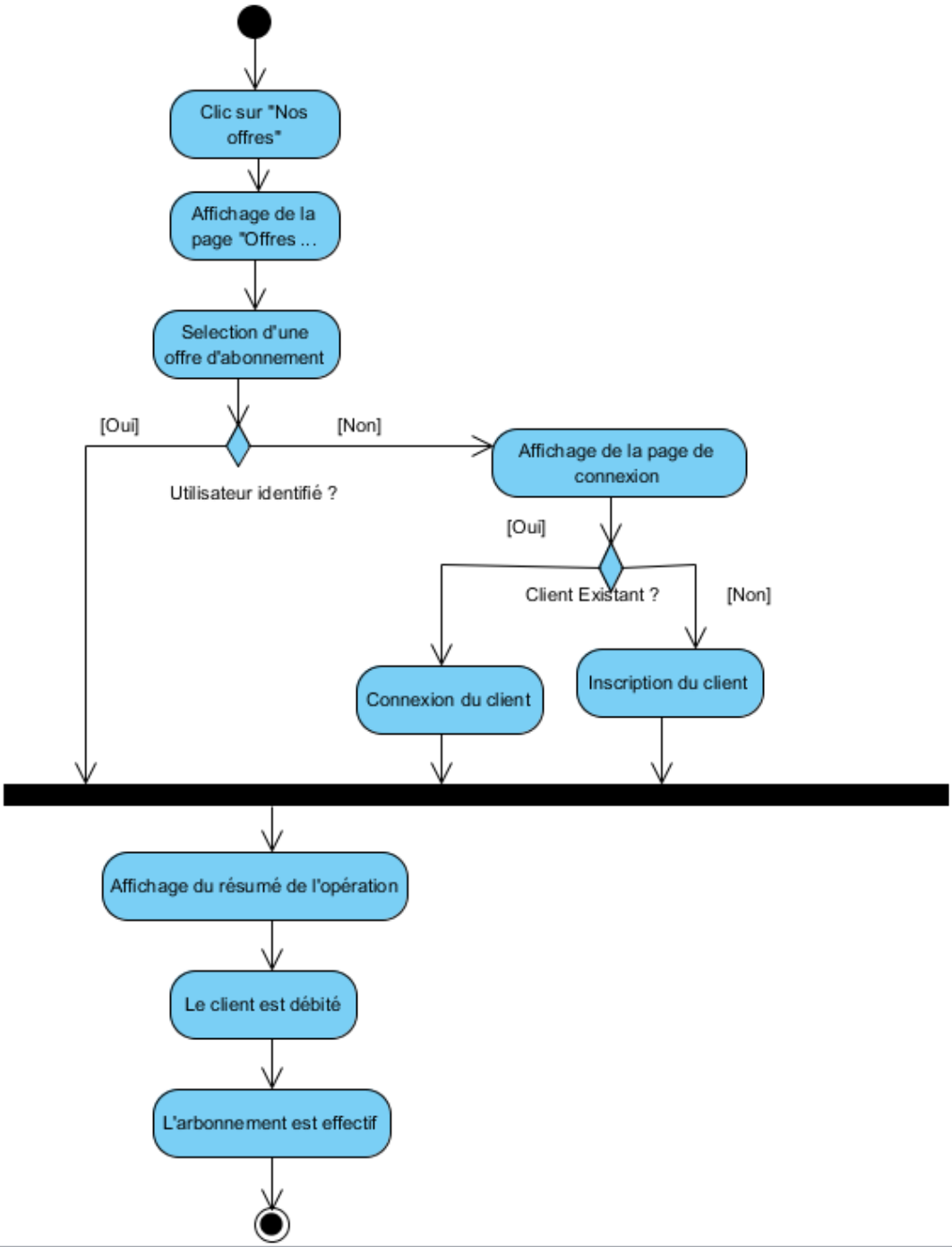
Pré-visualiser un article



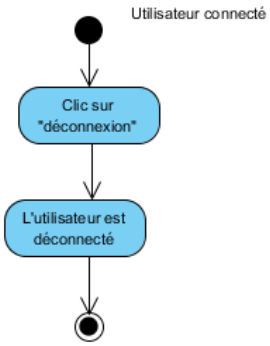
Acheter un article



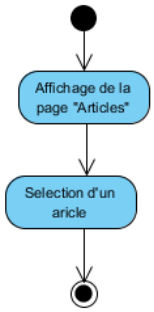
S'abonner


Clic sur "Nos offres"
Affichage de la page "Offres ...
Selection d'une offre d'abonnement
[Oui]
[Non]
Utilisateur identifié ?
Affichage de la page de connexion
[Oui]
Client Existant ?
[Non]
Connexion du client
Inscription du client
Affichage du résumé de l'opération
Le client est débité
L'arbonnement est effectif

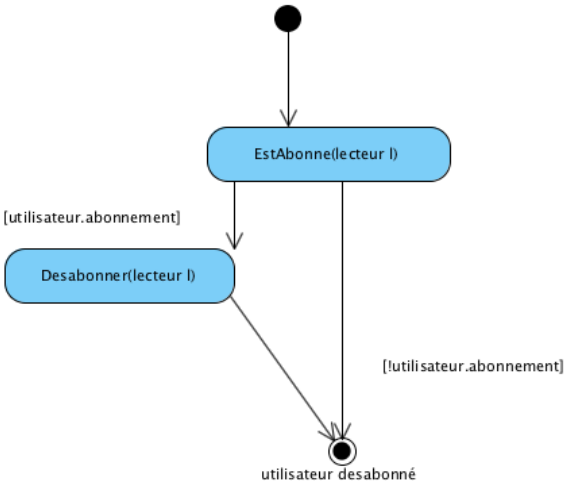
Se déconnecter



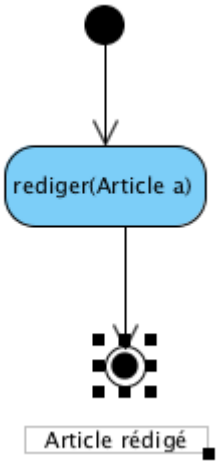
Consulter un article



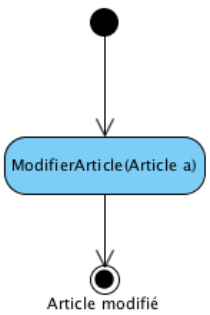
Se désabonner



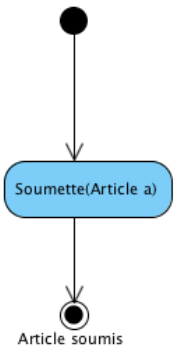
Écrire un article



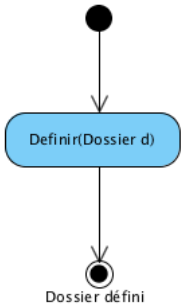
Modifier un article



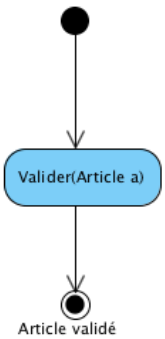
Soumettre un article



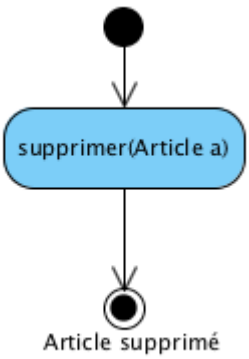
Définir un dossier



Valider un article

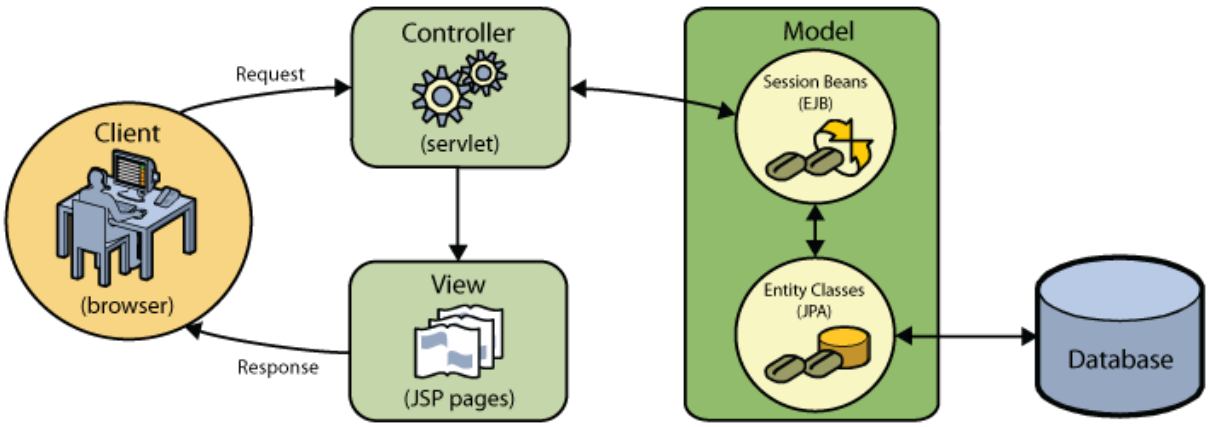


Supprimer un article



7. Architecture technique

7.1. Architecture applicative



L'application est structurée de manière à être plus facilement maintenue. Pour cela nous utiliserons les design pattern MVC (modèle - vue - contrôleur) et DAO. Elle se découpe en quatre modules distincts :

- module entity
- module DB
- module control
- module view

Module entity

Ce module regroupe **les classes métier de l'application** qui serons présentes sous formes **d'entity bean**. Cinq classes sont au coeur de notre application :

- utilisateur : permet de représenter les différents utilisateurs de l'application (visiteurs, utilisateurs inscrits, journalistes, rédacteurs en chef et administrateurs du site)

- article : représente les différents articles rédigés par les journalistes.

- message : représente les différents messages liés aux articles qui sont laissés par les rédacteurs en chef lors de refus ou de suppressions d'articles.

- dossier : représente les différents dossiers qui sont créés par les rédacteurs en chef et qui regroupent certains articles.

- achat : représente les achats que peut effectuer un utilisateur sur la plate-forme.

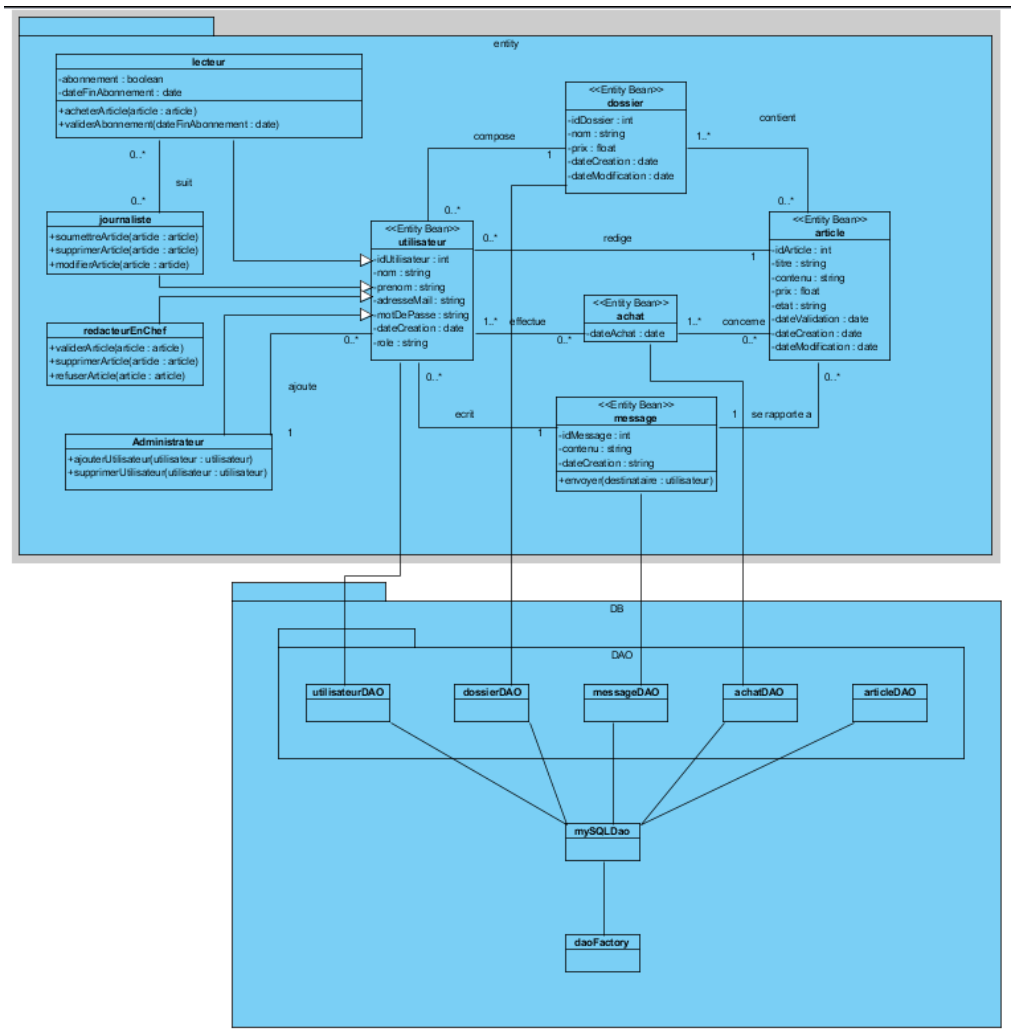
Module DB (database)

Ce module est chargé du d**ialogue entre les classes métier et la base de données**. Il met en place le design pattern DAO afin de permettre une plus grande modularité.

Il se comporte donc de classes DAO représentant chacune des classes métiers à préserver en base :

- utilisateur : utilisateurDAO
- article : articleDAO
- message : messageDAO
- dossier : dossierDAO
- achat : achatDAO

Le schéma suivant permet de montrer les différentes classes des modules entity et db et la manière dont ces classes interagissent.



Module control

Le module de contrôle va permettre de faire **le lien entre les différentes vues et les classes** métiers. On y retrouve un certain nombre de classes de contrôle.

Module view

Le module view se compose de toutes les interfaces de notre application.

VUES	SERVLET	EJB SESSION / METHODES	ENTITY BEAN
inscription connexion modifierProfil preference	UtilisateurServlet	EJB STATELESS creer() modifier() supprimer() getByMailAndPassword()	utilisateur
afficherTousArticles afficherUnArticle afficherTousArticlesAchetes acheterArticle	ArticleServlet	EJB STATELESS getAllArticles() getArticleById() getAllArticlesAchetes()	article achat
ecrireArticle modifierArticle afficherTousMesArticles soumettreArticle	JournalisteServlet	EJB STATELESS creerArticle() modifierArticle() getAllMyArticle()	article
afficherTousArticlesAValider validerArticle refuserArticle supprimerArticle envoyerMessage afficherTousDossiers creerDossier modifierDossier supprimerDossier	RedacteurServlet	EJB STATELESS getAllArticlesAValider() modifierArticle() supprimerArticle() envoyerMessage() getAllDossiers() creerDossier() modifierDossier() supprimerDossier()	article message dossier
afficherTousUtilisateurs creerUtilisateur supprimerUtilisateur	AdministrateurServlet	EJB STATELESS getAllUtilisateurs() creerUtilisateur() supprimerUtilisateur()	utilisateur

7.2. Architecture logicielle

Le système d'exploitation de notre serveur est :

Linux entreprise server

Sur le serveur sera installé un serveur d'application :

GlassFish 3.1.1

Enfin sur ce serveur sera installé une base de données :

MySQL 5.6
(connecteur JDBC)

7.3. Architecture matérielle

Un seul serveur contiendra l'application web et hébergera la base de données.

L'application sera accessible depuis n'importe quel poste équipé d'un des navigateurs supportés:

- Firefox 4+
- Chrome 20+
- Safari 4+
- Iinternet Explorer 9+

7.4. Politique de sécurité

Les informations relatives au projet ne sont divulguées qu'aux équipes participantes à ce projet. Les informations sensibles liées à l'application sont cryptées avant leur insertion en base afin de renforcer la sécurité des informations au sein de notre application.

8. Implémentation, règles de codage et conventions de nommage

La réalisation de ce projet devant se faire en équipe (par groupe de quatre), il est nécessaire pour nous d'utiliser un outil nous permettant de travailler tous ensemble de manière efficace (travail en parallèle) sur les différentes fonctionnalités de l'application.

Nous avons choisi d'utiliser un logiciel de gestion de versions à savoir GIT associé à un service web d'hébergement et de gestion de développement de logiciels Bitbucket (https://bitbucket.org/) . Notre choix s'est porté sur GIT, car d'une part, c'est un outil que nous maîtrisons tous les quatre et d'autre part, il nous permet d'avoir un bon suivi de l'avancement du projet, du travail des autres membres du groupe et de pouvoir récupérer facilement le code des autres développeurs.

Afin de garantir une certaine qualité et cohérence dans notre code, nous avons établi un ensemble de règles de programmation et nous nous sommes engagés à respecter des conventions de nommage et d'indentations communes. Nous utiliserons notamment la notation "CamelCase" pour le nommage de variables ou de classes... ainsi que l'usage uniquement de minuscule (et d'underscore) pour le nommage de tables ou de colonnes dans la base de données.

Nous avons également décidé de mettre en place les différents Design Pattern (MVC, Factory, Singleton, ...) afin d'avoir un code de qualité et plus facilement maintenable.