**XML – Mini projet**

ProjectArchiver – Rapport final

Grégory Brissonnette

Grégory Moinat

Thomas Jordan

5 juin 2009

**Résumé**

Le projet ProjectArchiver permet aux professeurs d’archiver des projets (code source, exécutable, description, etc…) en ligne et aux élèves de consulter ces archives.

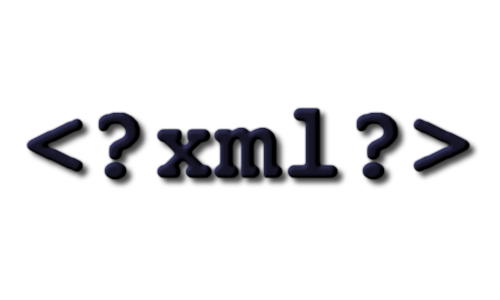


Table des matières

[1 Cahier des charges 3](#_Toc231920323)

[1.1 Objectifs 3](#_Toc231920324)

[1.2 XML, Schéma des fichiers XML 3](#_Toc231920325)

[1.3 Cas d’utilisation 4](#_Toc231920326)

[1.4 Planification et répartition des tâches 4](#_Toc231920327)

[2 Introduction 5](#_Toc231920328)

[3 Fonctionnalités 5](#_Toc231920329)

[4 Objectifs 5](#_Toc231920330)

[5 Structure des fichiers XML 6](#_Toc231920331)

[5.1 Description des fichiers XML 6](#_Toc231920332)

[5.2 Stockage des cours 6](#_Toc231920333)

[5.3 Schéma XSD des cours 6](#_Toc231920334)

[5.4 Stockage des projets 7](#_Toc231920335)

[5.5 Schéma XSD d’un projet 7](#_Toc231920336)

[6 Décrire les flux de transformation XML au travers de l'applicatif 8](#_Toc231920337)

[7 Données de test 8](#_Toc231920338)

[7.1 Cours - Données initiales 8](#_Toc231920339)

[7.2 Cours - Données résultantes 8](#_Toc231920340)

[7.3 Projets - Données initiales 9](#_Toc231920341)

[7.4 Projets - Données résultantes 11](#_Toc231920342)

[8 Définition du XSL 13](#_Toc231920343)

[9 Interrogations XQuery et résultats 14](#_Toc231920344)

[9.1 Sélectionner les projets qui ont été réalisé dans le cadre d'un cours donné soit en toute lettre, soit par un id. 14](#_Toc231920345)

[9.2 Sélectionner les projets qui ont été données par un prof durant une année donnée. 14](#_Toc231920346)

[9.3 Sélectionner tous les projets auxquels un auteur à participé. 15](#_Toc231920347)

[9.4 Sélectionner les projets qui contiennent un mot clé dans la liste des mots clé. 15](#_Toc231920348)

[9.5 Lister chaque prof avec toutes les branches qu'ils ont données 16](#_Toc231920349)

[10 Description des fragments de code Dom 17](#_Toc231920350)

[10.1 Ajout d’un projet : 17](#_Toc231920351)

[10.2 Suppression d’un projet : 19](#_Toc231920352)

[10.3 Modification d’un projet : 19](#_Toc231920353)

[10.4 Ajout d’un cours 20](#_Toc231920354)

[11 Tests 21](#_Toc231920355)

[12 \*Reste à faire 22](#_Toc231920356)

[12 Reste à faire 23](#_Toc231920357)

[13 Problèmes rencontrés 23](#_Toc231920358)

[14 Conclusion 23](#_Toc231920359)

[15 Annexes 23](#_Toc231920360)

# Cahier des charges

## Objectifs

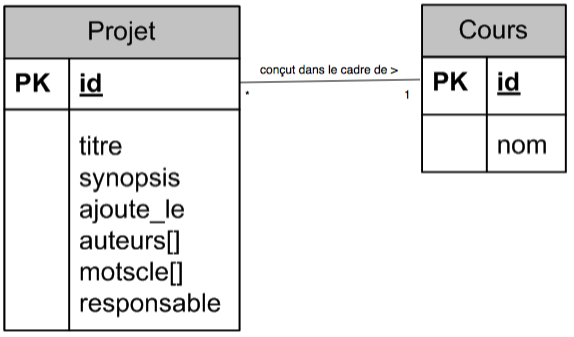
Durant leurs études, les étudiants en informatique de la HEIG-VD réalisent un nombre important de projets. Cependant, dans la plupart des cas, ces travaux sont oubliés dès qu’ils sont terminés alors que certains mériteraient d’être repris ou pourraient être des sources d’informations importantes pour le développement d’autres logiciels.

Le but de ce projet est de créer un outil qui répond à cette problématique. Ceci se concrétiserait par un site web sur lequel les professeurs peuvent archiver les projets (code source, exécutable, auteurs, commentaires, mots-clés, …). Les étudiants de leur côté pourraient se connecter au site sans authentification pour consulter ces archives. Afin de faciliter l’utilisation du site, le LDAP de l’école serait utilisé pour authentifier les professeurs. Ainsi, chaque professeur aurait automatiquement un compte.

Le cadre de ce projet étant l’apprentissage de la technologie XML, la base de données reposerait sur ce dernier. D’autre part, afin de mieux appréhender les technologies XML, XSLT sera utilisé dans une vue dans laquelle tous les projets apparaissent.

## XML, Schéma des fichiers XML

Deux fichiers XML seront utilisés dans le programme. Le premier représente les projets alors que le deuxième représente les cours.



## Cas d’utilisation

Completer avec gerer les cours.

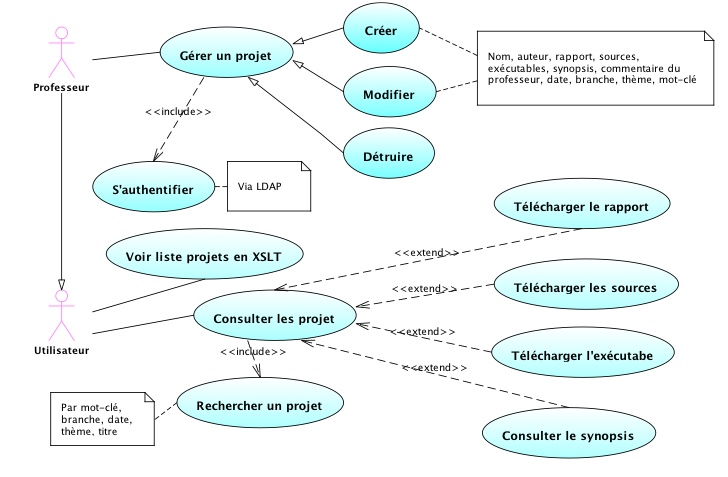


Figure - Cas d'utilisation

## Planification et répartition des tâches

|  |  |  |
| --- | --- | --- |
| **Description** | **Responsable** | **Date d’échéance** |
| XSchema | Thomas | 13 mai 2009 |
| Module d’authentification | Grégory M. | 13 mai 2009 |
| Ajouter des branches | Grégory B. | 13 mai 2009 |
| Création de projet (Front end) | Grégory B. | 20 mai 2009 |
| Création de projet (Back end) | Grégory M | 20 mai 2009 |
| Gestion de projet (Front end) | Grégory M. | 27 mai 2009 |
| Gestion de projet (Back end) | Grégory B. | 27 mai 2009 |
| Recherche de projet | Thomas | 27 mai 2009 |
| Vue XSLT | Thomas | 20 mai 2009 |
| Rapport | Thomas | 6 juin 2009 |
| Test + Réserve | Grégory B. + M. | 6 juin 2009 |

# Introduction

Dans le cadre des mini-projets du cours XML, nous avons choisi de réaliser une application avec GWT pour le coté client, PHP pour le coté serveur et XML pour le stockage des données (imposé). L’application permet de gérer les projets que les étudiants de l’école réalisent au cours de leur formation. Elle permet de mettre les travaux des élèves à disposition des étudiants des années suivantes pour pouvoir les reprendre ou simplement les utiliser. En plus de pouvoir consulter la liste des projets, il y aura la possibilité de télécharger une archive contenant le rapport, les sources et l’exécutable.

# Fonctionnalités

1. S’identifier si on est un professeur
2. Ajouter un projet
3. Modifier un projet
4. Supprimer un projet
5. Ajouter des cours
6. Rechercher des projets pour pouvoir les consulter
7. Visionner la liste des projets à travers une page HTML générée par un fichier XSL

# Objectifs

* Réaliser des interfaces web avec Google Web Toolkit (GWT)
* Implémenter les méthodes permettant d’ajoute, modifie et supprime les données stockées dans des fichiers XML avec DOM.
* Définir la structure des fichiers XML avec XML Schéma.
* Implémenter la gestion des données via des fichiers XML
* Fournir une vue au format XSL
* Formuler des requêtes XQuerry

# Structure des fichiers XML

## Description des fichiers XML

Pour gérer les données de notre site nous avons besoin de deux fichiers XML. L’un permet de stocker la liste des branches et le deuxième permet de stocker les informations sur les projets. Un projet est lié à un cours par une clé étrangère. L’intégrité n’est pas assurée, mais l’unicité des id des branches et des projets l’est.

Pour décrire la structure XSD, nous avons utilisé l’éditeur graphique Oxygen XML qui nous permit de gérer des fichiers de manière très simple.

## Stockage des cours

<?xml version="1.0" encoding="ISO-8859-1"?>

<**listeDesCours** xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="**listeDesCours.xsd**">

<cours id="1">

<nom>BDR</nom>

</cours>

<cours id="2">

<nom>PDG</nom>

</cours>

<cours id="3">

<nom>XML</nom>

</cours>

<cours id="4">

<nom>WEB</nom>

</cours>

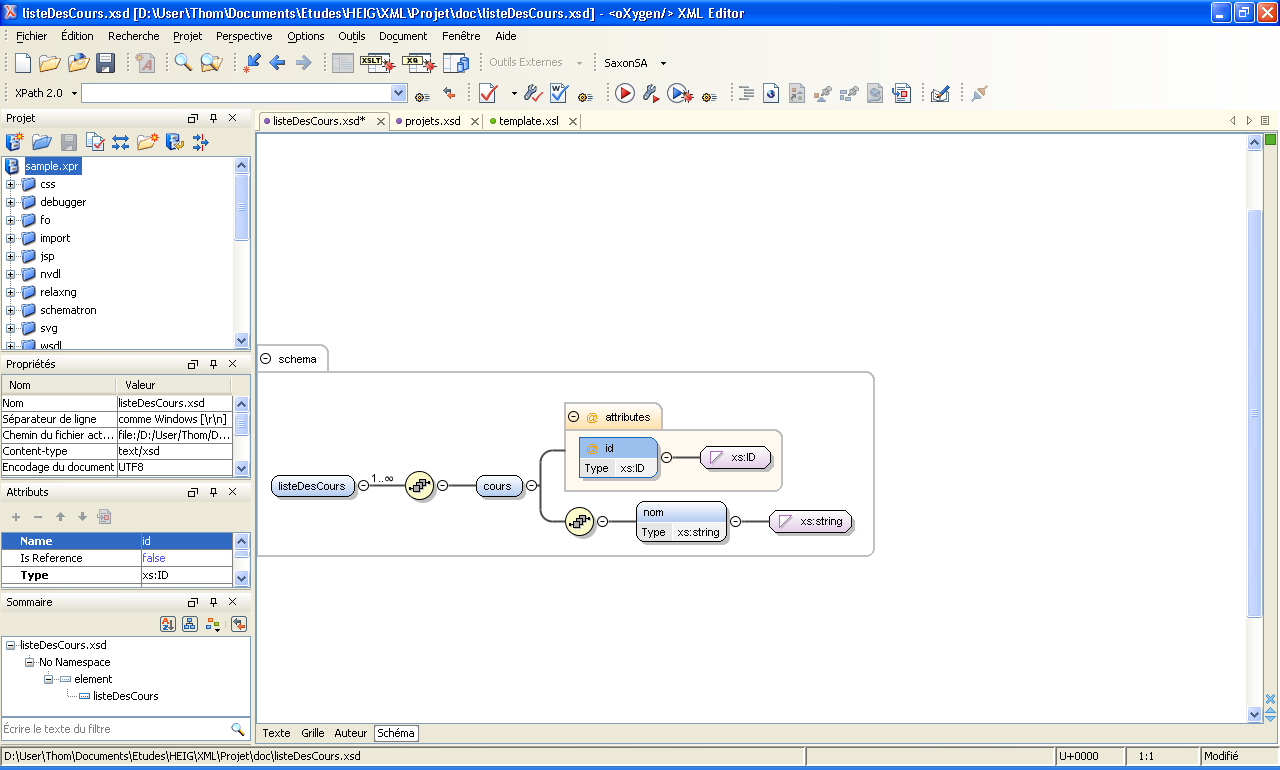
<cours id="5">

<nom>RIM</nom>

</cours>

</**listeDesCours**>

## Schéma XSD des cours



## Stockage des projets

<?xml version="1.0" encoding="ISO-8859-1" ?>

<?xml-stylesheet type="text/xsl" href="projets.xsl"?>

<**projets** xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="**projets.xsd**">

<projet id="1">

<titre>Project Archiver</titre>

<idBranche>1</idBranche>

<synopsis>C'est l'un des plus gand projet jamais réalisé à la HEIG</synopsis>

<ajouteLe>2008-05-14</ajouteLe>

<responsables>

<responsable>Eric LeFrancois</responsable>

</responsables>

<auteurs>

<auteur>Greg One</auteur>

<auteur>Greg Two</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>Idée géniale</motCle>

<motCle>Inovation</motCle>

<motCle>Greg</motCle>

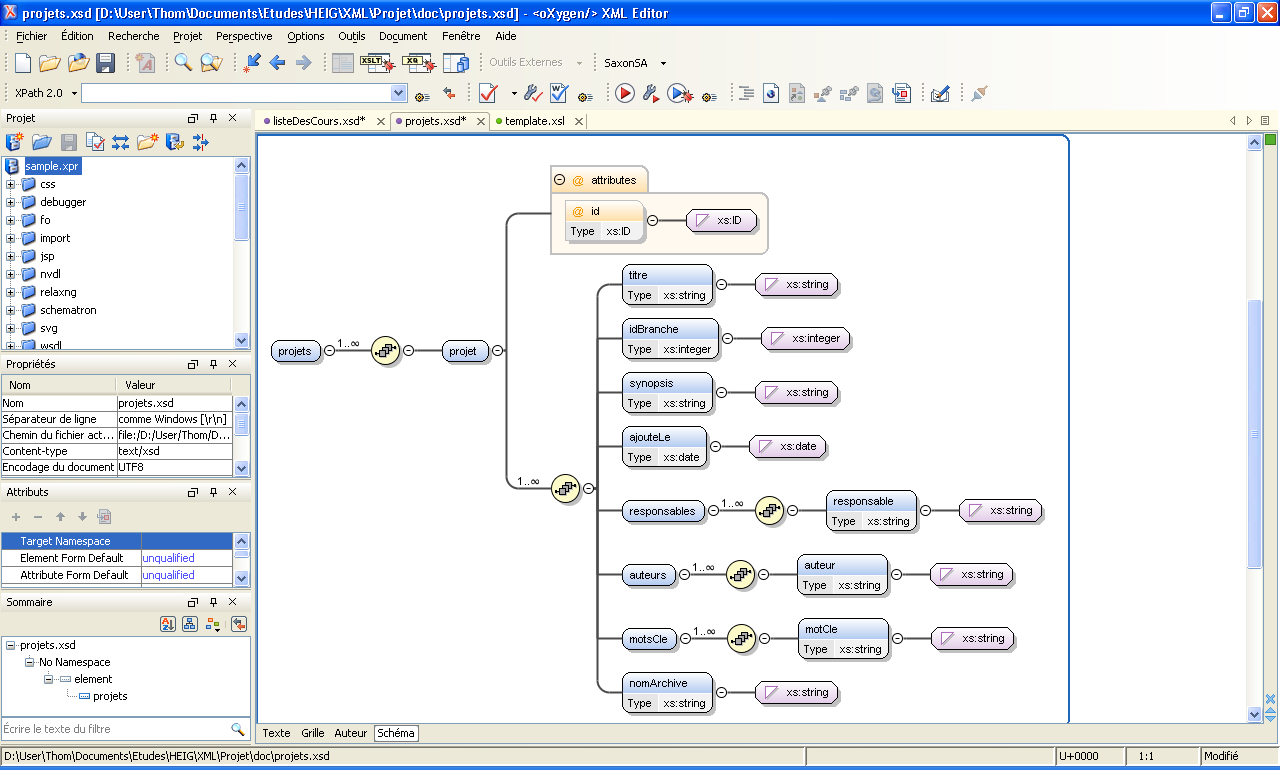
</motsCle>

<nomArchive>project\_Archiver\_megahit.tar.bz2</nomArchive>

</projet>

</**projets**>

## Schéma XSD d’un projet



# Décrire les flux de transformation XML au travers de l'applicatif

Dans l’application nous utilisons XML pour stocker les données, mais aussi pour transmettre les informations entre le serveur et le client. En effet, le client reçoit un flux XML et les widget de la GWT permettent directement de mettre en forme un flux XML dans des tables.

Nous utilisons aussi le XSLT pour afficher la liste de tous les projets enregistrés dans les fichiers XML. Il suffit de lancer une requête pour ouvrir le fichier XML qui est situé sur le serveur permettant d’afficher le fichier et la magie du XSL fait que le fichier XML est mis en page automatiquement.

# Données de test

## Cours - Données initiales

<?xml version="1.0" encoding="UTF-8"?>

<listeDesCours xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="listeDesCours.xsd">

<cours id="1">

<nom>BDR</nom>

</cours>

<cours id="2">

<nom>XML</nom>

</cours>

<cours id="3">

<nom>WEB</nom>

</cours>

<cours id="4">

<nom>GEN</nom>

</cours>

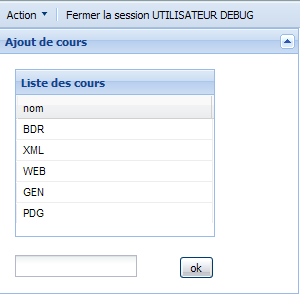
<cours id="5">

<nom>PDG</nom>

</cours>

</listeDesCours>

## Cours - Données résultantes



## Projets - Données initiales

<?xml version="1.0" encoding="ISO-8859-1" ?>

<?xml*-stylesheet* type="text/xsl" href="projets.xsl"?>

<projets xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="projets.xsd">

<projet id="1">

<titre>Project Archiver</titre>

<idBranche>2</idBranche>

<synopsis>C'est l'un des plus gand projet jamais réalisé à la HEIG</synopsis>

<ajouteLe>2009-06-02</ajouteLe>

<responsables>

<responsable>Eric LeFrancois</responsable>

</responsables>

<auteurs>

<auteur>Grégory One</auteur>

<auteur>Grégory Two</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>Idée géniale</motCle>

<motCle>Inovation</motCle>

<motCle>Archiver</motCle>

<motCle>Awsome</motCle>

</motsCle>

<nomArchive>project\_Archiver\_megahit.tar.bz2</nomArchive>

</projet>

<projet id="2">

<titre>Tron Cycle</titre>

<idBranche>4</idBranche>

<synopsis>Anotherone gregs'fabulous idea</synopsis>

<ajouteLe>2008-03-18</ajouteLe>

<responsables>

<responsable>Eric LeFrancois</responsable>

</responsables>

<auteurs>

<auteur>Greg M(agnifique)oinat</auteur>

<auteur>Lionel Erard</auteur>

<auteur>Julien Leglaunec</auteur>

<auteur>Nicolas Muller</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>Tricyle</motCle>

<motCle>Tron</motCle>

<motCle>Greg</motCle>

</motsCle>

<nomArchive>proj\_gen\_12.12.2008\_troncycle.zip</nomArchive>

</projet>

<projet id="3">

<titre>AlloBiblio</titre>

<idBranche>3</idBranche>

<synopsis>Site comme allociné(mais mieux). Au lieu de parler de filmes on y parle de livres.</synopsis>

<ajouteLe>2009-06-12</ajouteLe>

<responsables>

<responsable>Eric LeFrancois</responsable>

</responsables>

<auteurs>

<auteur>Christos Kornaros</auteur>

<auteur>Grégory Brissonnette</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>allociné</motCle>

<motCle>livre</motCle>

<motCle>rails</motCle>

<motCle>ror</motCle>

<motCle>ruby</motCle>

</motsCle>

<nomArchive>[web]alloBiblio1.0.zip</nomArchive>

</projet>

<projet id="4">

<titre>Chat virtuel 3D</titre>

<idBranche>5</idBranche>

<synopsis>Application réalisée avec XNA, qui permet de se déplacer dans un monde virtuel et de chatter avec 8 autres joueurs</synopsis>

<ajouteLe>2009-01-20</ajouteLe>

<responsables>

<responsable>Jean-Pierre Molliet</responsable>

<responsable>Abdelali Guerride</responsable>

</responsables>

<auteurs>

<auteur>Christophe Lugon</auteur>

<auteur>Tristan Rouiller</auteur>

<auteur>Lionel Erard</auteur>

<auteur>Julien LeGlaunec</auteur>

<auteur>Bertran Zang-Ondua</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>chat</motCle>

<motCle>second life</motCle>

<motCle>xna</motCle>

<motCle>3D</motCle>

</motsCle>

<nomArchive>ChatVirtuel - PDG.zip</nomArchive>

</projet>

<projet id="5">

<titre>Gestion d'une societé de gymn</titre>

<idBranche>1</idBranche>

<synopsis>8 Application web pour gérer une société de gymnastique permettant de gérer les membres, gérer les plannings d'entrainement et les absences aux entrainements. Permet aussi de noter les résultats des concours et de classer les gymnastes en interne</synopsis>

<ajouteLe>2008-04-15</ajouteLe>

<responsables>

<responsable>Nastaran Fatemi</responsable>

</responsables>

<auteurs>

<auteur>Grégory Brissonnette</auteur>

<auteur>Thomas Jordan</auteur>

</auteurs>

<motsCle>

<motCle>gymnastique</motCle>

<motCle>mySQL</motCle>

<motCle>requêtes</motCle>

<motCle>societé</motCle>

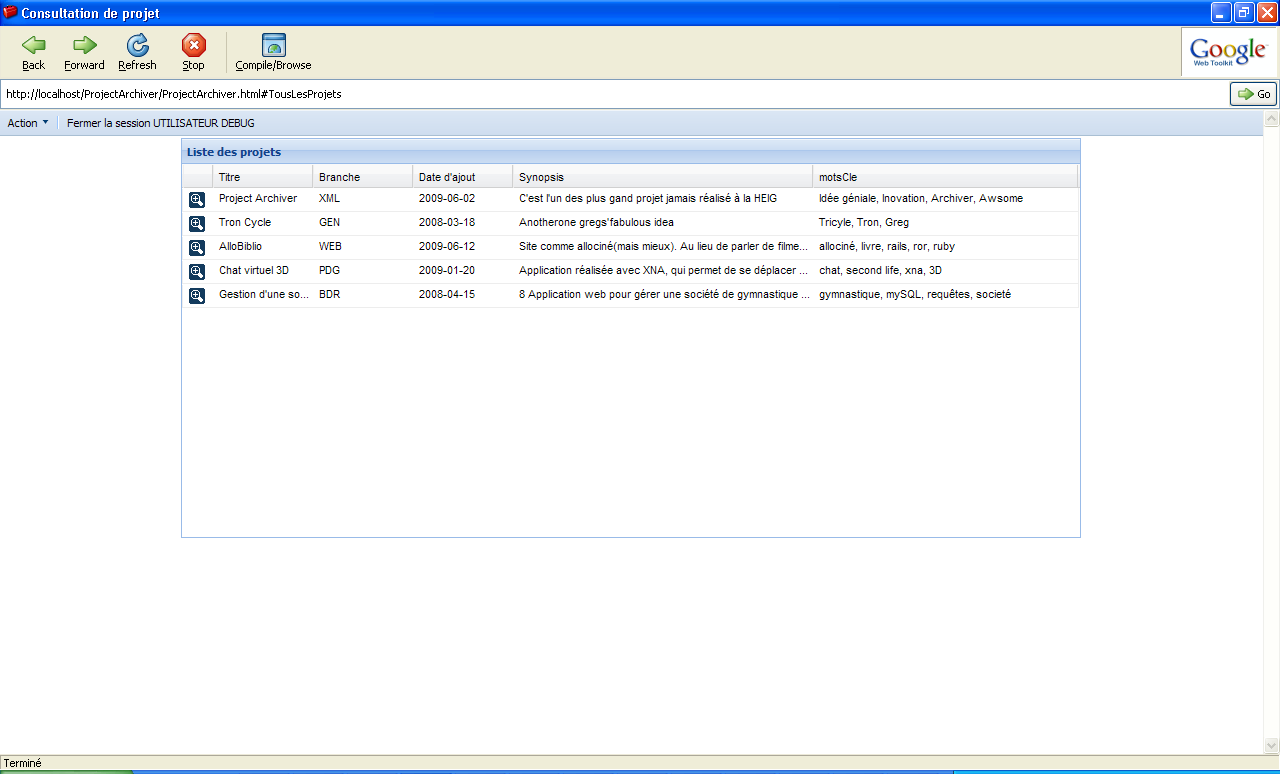
</motsCle>

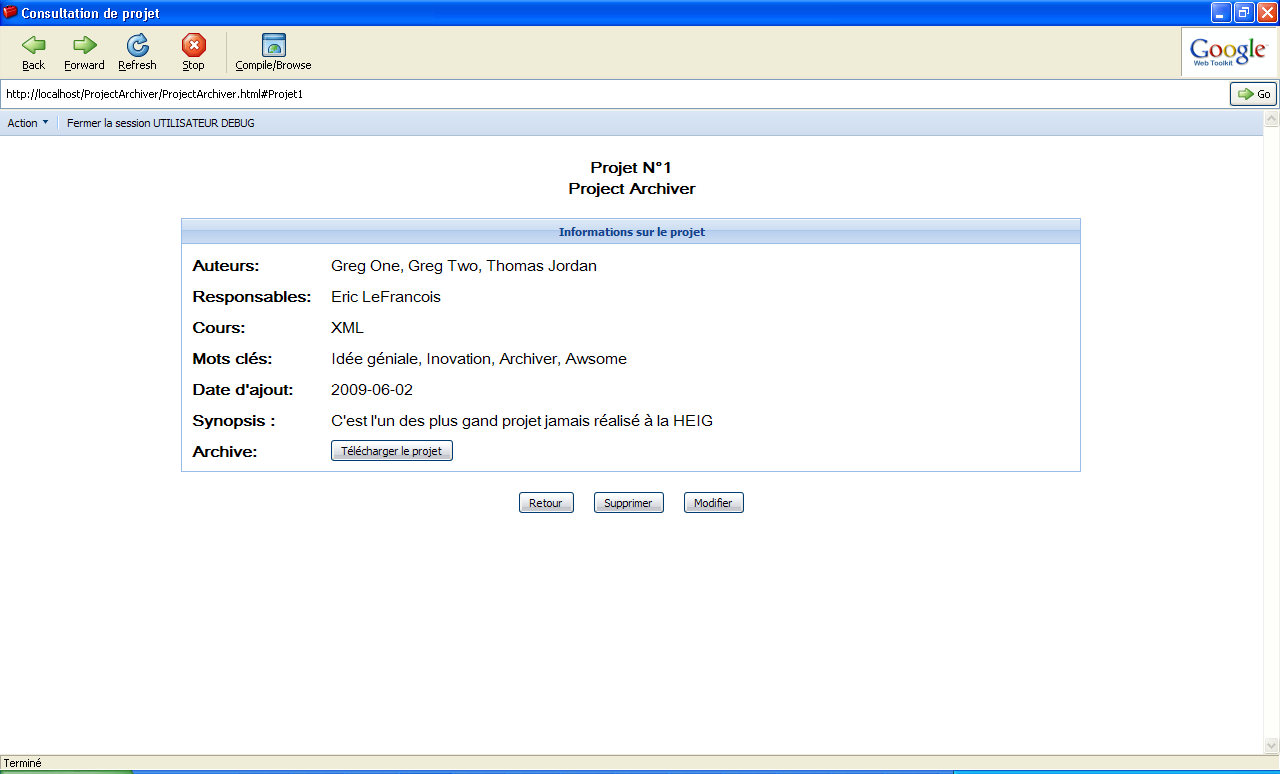
<nomArchive>[BDR] Projet\_Societé\_Gymnastique.rar</nomArchive>

</projet>

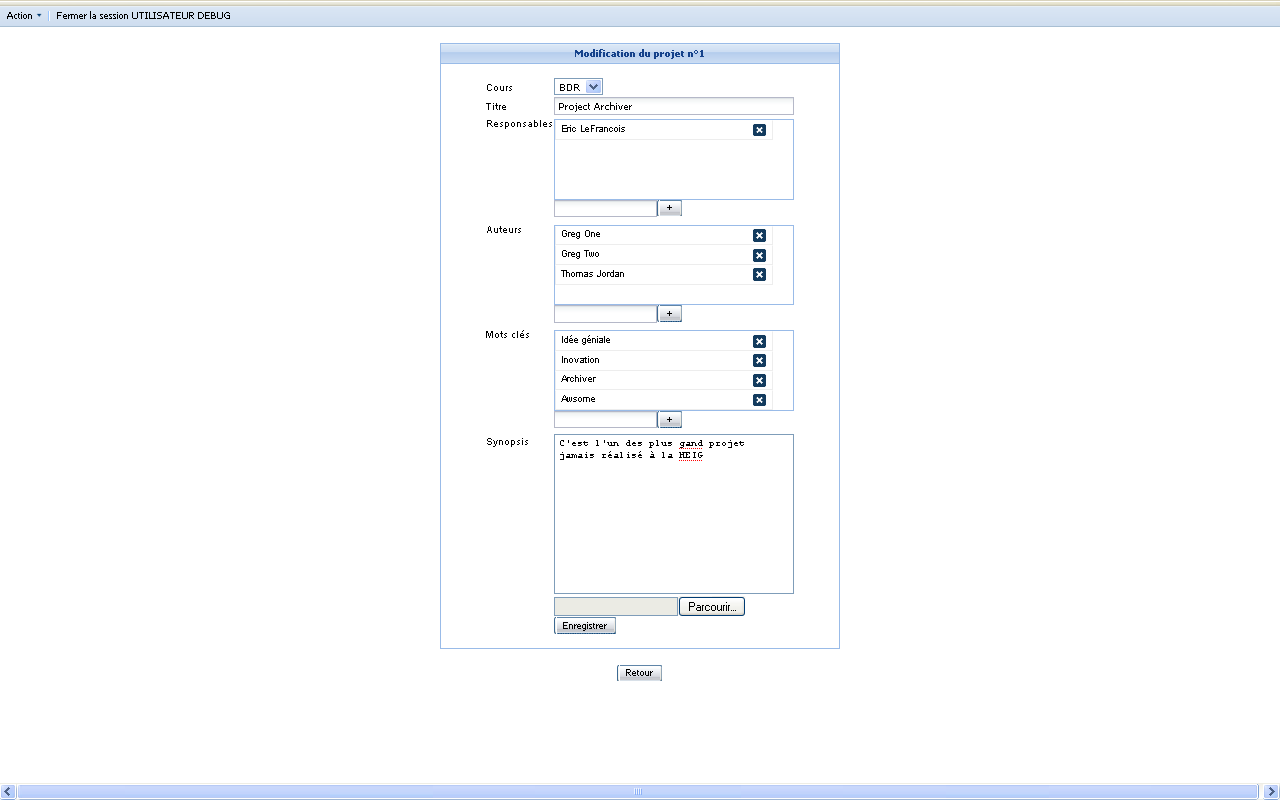
</projets>

## Projets - Données résultantes









# Définition du XSL

Notre un fichier xsl permet de transformer l’XML de la liste des projets en une page HTML listant tous les projets répertoriés. Cette page html affiche tous les détails du fichier xml et inclus des liens pour naviguer dans la page. Il y a aussi pour chaque projet le lien permettant de récupérer l’archive correspondante. La mise en page est faite en partie par un style CSS.

Cette vue ressemble à cela :



ZONE 2

…

ZONE 1

Le résultat est un tableau composé de deux parties principales. La zone 1 regroupe des liens vers les projets qui sont affichés sur la page. La zone 2 s’étend jusqu’au pied de la page, elle contient le listing complet avec le détail des projets sous forme de tableau. Un lien au bas de chaque projet permet de télécharger l’archive.

Il y a juste une spécialité par rapport à ce que l’on a vu dans les laboratoires, c’est qu’un projet est lié à une branche. Dans le projet il y a un champ numérique « idBranche » qui contient l’id du cours correspondant. Il faut faire une sorte de jointure pour récupérer le label de la branche pour l’afficher en toute lettre.

Il suffit de récupérer l’id dans une variable, et de sélectionner dans le deuxième fichier, le nom du cours dont l’id est égal à la variable.

<xsl:variable name="idCoursCourant" select="idBranche"/>

<xsl:value-of select="document('listeDesCours.xml')

//cours[@id=$idCoursCourant]/nom"/>

# Interrogations XQuery et résultats

N’ayant pas eu le temps d’implémenter ces requêtes xQuery au projet, nous avons comme convenu, écrit les requêtes et les avons testées à l’aide du logiciel eXist. Dans les requêtes ci-dessous les parties en gras représentent les paramètres qui sont normalement fourni par l’utilisateur.

## Sélectionner les projets qui ont été réalisé dans le cadre d'un cours donné soit en toute lettre, soit par un id.

for $p in doc("projets.xml") //projet,

$c in doc("listeDesCours.xml")//cours

where $p/idBranche = $c/@id

and ($c/nom ="BDR" -- L’un ou l’autre

or $p/idBranche= "1") -- 1 est l’id de BDR

return

<resultat>

{ $p/titre }

</resultat>

Résultat :

<resultat>

<titre>Gestion d'une societé de gymn</titre>

</resultat>

## Sélectionner les projets qui ont été données par un prof durant une année donnée.

for $p in doc("projets.xml") //projet

where contains($p//responsable, "**Eric**")

and contains($p/ajouteLe,"**2009**")

return

<resultat>

{ $p/titre }

</resultat>

Résultat :

<resultat>

<titre>Project Archiver</titre>

</resultat>

<resultat>

<titre>AlloBiblio</titre>

</resultat>

## Sélectionner tous les projets auxquels un auteur à participé.

for $p in doc("projets.xml") //projet

where contains($p//auteur, "Grégor")

return

<resultat>

{ $p/titre }

</resultat>

Résultat :

<resultat>

<titre>Project Archiver</titre>

</resultat>

<resultat>

<titre>Tron Cycle</titre>

</resultat>

<resultat>

<titre>AlloBiblio</titre>

</resultat>

<resultat>

<titre>Gestion d'une societé de gymn</titre>

</resultat>

## Sélectionner les projets qui contiennent un mot clé dans la liste des mots clé.

for $p in doc("projets.xml") //projet

where contains($p//motCle, "Inovation")

return

<resultat>

{ $p/titre }

</resultat>

Résultat :

<resultat>

<titre>Project Archiver</titre>

</resultat>

## Lister chaque prof avec toutes les branches qu'ils ont données

for $res in distinct-values(document("projets.xml")//responsable)

return

<resultat>

<prof>

{$res}

</prof>

{

for $id in document("projets.xml")

//idBranche[..//responsable=$res],

$c in doc("listeDesCours.xml")//cours

where $id = $c/@id

return

<cours>

{$c/nom}

</cours>

}

</resultat>

Résultat :

<resultat>

<prof>Eric LeFrancois</prof>

<cours>

<nom>XML</nom>

</cours>

<cours>

<nom>GEN</nom>

</cours>

<cours>

<nom>WEB</nom>

</cours>

</resultat>

<resultat>

<prof>Jean-Pierre Molliet</prof>

<cours>

<nom>PDG</nom>

</cours>

</resultat>

<resultat>

<prof>Abdelali Guerride</prof>

<cours>

<nom>PDG</nom>

</cours>

</resultat>

<resultat>

<prof>Nastaran Fatemi</prof>

<cours>

<nom>BDR</nom>

</cours>

</resultat>

# Description des fragments de code Dom

Pour ajouter/modifier/supprimer des projets du fichier XML nous utilisons DOM depuis PHp. Voici ci-dessous le code qui permet d’ajouter un projet. Pour la modification, nous supprimons l’ancien projet et le rajoutons.

## Ajout d’un projet :

$document = **new** DOMDocument();

// Supprime l'indentation avant la lecture

$document->preserveWhiteSpace = **false**;

$document->load("../xml/projets.xml");

// Réindente le fichier

$document->formatOutput = **true**;

**if** ($id == "") {

// Récupération de l'id le plus grand (pour générer l'id de ce projet)

$xpath = **new** DOMXPath($document);

// Récupération des ids des projets

$ids = $xpath->evaluate("/projets/projet/@id");

// Sélection du plus grand id

$id = 0;

**for** ($i = 0; $i < $ids->length; $i++)

**if** ((int) $ids->item($i)->nodeValue > $id)

$id = (int) $ids->item($i)->nodeValue;

// On ajoute 1 pour créer le nouvel id

$id++;

}

// Création de éléments composants le projet

$nouveauProjet = $document->createElement("projet");

$nouveauProjet->setAttribute("id", $id);

// Titre

$elemTitre = $document->createElement("titre");

$valeurTitre = $document->createTextNode($titre);

$elemTitre->appendChild($valeurTitre);

$nouveauProjet->appendChild($elemTitre);

// Branche

$elemIdBranche = $document->createElement("idBranche");

$valeurIdBranche = $document->createTextNode($idBranche);

$elemIdBranche->appendChild($valeurIdBranche);

$nouveauProjet->appendChild($elemIdBranche);

// Synopsis

$elemSynopsis = $document->createElement("synopsis");

$valeurSynopsis = $document->createTextNode($synopsis);

$elemSynopsis->appendChild($valeurSynopsis);

$nouveauProjet->appendChild($elemSynopsis);

// Date

$date = **new** DateTime("now", **new** DateTimeZone("Europe/Berlin"));

$elemDate = $document->createElement("ajouteLe");

$valeurDate = $document->createTextNode($date->format("d/m/Y"));

$elemDate->appendChild($valeurDate);

$nouveauProjet->appendChild($elemDate);

// Responsables

$elemResponsables = $document->createElement("responsables");

$responsablesSepares = split(";", $responsables);

**for** ($i = 0; $i < count($responsablesSepares); $i++) {

$tabElemResponsable[$i] = $document->createElement("responsable");

$tabValeursResponsable[$i] = $document-

>createTextNode($responsablesSepares[$i]);

$tabElemResponsable[$i]->appendChild($tabValeursResponsable[$i]);

$elemResponsables->appendChild($tabElemResponsable[$i]);

}

$nouveauProjet->appendChild($elemResponsables);

// Auteurs

$elemAuteurs = $document->createElement("auteurs");

$auteursSepares = split(";", $auteurs);

**for** ($i = 0; $i < count($auteursSepares); $i++) {

$tabElemAuteur[$i] = $document->createElement("auteur");

$tabValeursAuteur[$i] = $document->createTextNode($auteursSepares[$i]);

$tabElemAuteur[$i]->appendChild($tabValeursAuteur[$i]);

$elemAuteurs->appendChild($tabElemAuteur[$i]);

}

$nouveauProjet->appendChild($elemAuteurs);

// Mots-clés

$elemMotsCle = $document->createElement("motsCle");

$motsCleSepares = split(";", $motsCle);

**for** ($i = 0; $i < count($motsCleSepares); $i++) {

$tabElemMotCle[$i] = $document->createElement("motCle");

$tabValeursMotCle[$i] = $document->createTextNode($motsCleSepares[$i]);

$tabElemMotCle[$i]->appendChild($tabValeursMotCle[$i]);

$elemMotsCle->appendChild($tabElemMotCle[$i]);

}

$nouveauProjet->appendChild($elemMotsCle);

// Nom de l'archive

$elemArchive = $document->createElement("nomArchive");

$valeurArchive = $document->createTextNode($nomArchive);

$elemArchive->appendChild($valeurArchive);

$nouveauProjet->appendChild($elemArchive);

// Récupération de la racine du document pour y ajouter le nouveau projet

$racine = $document->getElementsByTagName("projets")->item(0);

$racine->appendChild($nouveauProjet);

// On enregistre le document

$document->save("../xml/projets.xml");

Le principe est exactement le même avec PHPDom qu’avec JDom. Tout d’abord on test si $id contient une valeur. Si c’est le cas, on continue, sinon il faut rechercher quelle sera la valeur de l’id en fonction des id déjà présents dans le fichier XML. Après cela on créer un élément projet, auquel on ajoute l’attribut ID, et ensuite pour les fils de projet, on construits ces fils, et on les ajoute en tant qu’enfant du projet. A la fin il suffit d’enregistrer avec l’instruction save.

## Suppression d’un projet :

// Suppression éventuelle de l'archive

**if** ($aussiArchive) {

**if** (!unlink("../projectFiles/$id/". getNomArchive($id)))

**return** "!error";

**if** (!rmdir("../projectFiles/$id/"))

**return** "!error";

}

$document = **new** DOMDocument();

// Supprime l'indentation avant la lecture

$document->preserveWhiteSpace = **false**;

$document->load("../xml/projets.xml");

// Réindente le fichier

$document->formatOutput = **true**;

$projets = $document->documentElement;

// Récupération du projet ayant l'id concerne

$xpath = **new** DOMXPath($document);

$projet = $xpath->query("/projets/projet[@id='" . $id . "']");

$projetSupprime = **null**;

**foreach** ($projet **as** $resultat)

$projetSupprime = $projets->removeChild($resultat);

**if** ($projetSupprime == **null**)

**return** "!not found";

// On enregistre le document

$document->save("../xml/projets.xml");

Lorsque l’on supprime un projet, on donne en paramètre en plus de l’id un booléen permettant de dire si l’on veut supprimer l’archive contenant le code source. Ensuite on recherche le nœud projet possédant l’id « $id ». Le foreach permet de retirer les nœuds correspondant à la requête xPath. A la fin il faut aussi sauver avec save.

## Modification d’un projet :

// On regarde si l'archive a été modifiée

**if** ($nomArchive == "") {

$nom = getNomArchive($id);

supprimerProjet($id, **false**);

ajouterProjet($id, $titre, $idBranche, $synopsis, $responsables,

$auteurs, $motsCle, $nom);

} **else** {

// On supprime l'ancien fichier

**if** (!unlink("../projectFiles/$id/". getNomArchive($id)))

**return** "!error";

supprimerProjet($id, **false**);

ajouterProjet($id, $titre, $idBranche, $synopsis, $responsables,

$auteurs, $motsCle, $nomArchive);

}

Pour modifier un projet, on va dans tous les cas nous supprimons le projet et le recréons. La valeur booléenne de supprimerProjet est dans les deux cas à false. On s’occupe localement de la suppression de l’archive existante.

## Ajout d’un cours

Pour l’ajout d’un cours, nous utilisons le même principe que pour l’ajout d’un projet. Voici le code :

$document = **new** DOMDocument();

// Supprime l'indentation avant la lecture

$document->preserveWhiteSpace = **false**;

$document->load("../xml/listeDesCours.xml");

// Réindente le fichier

$document->formatOutput = **true**;

// Récupération de l'id le plus grand (pour générer l'id de ce cours)

$xpath = **new** DOMXPath($document);

// Récupération des ids des projets

$ids = $xpath->evaluate("/listeDesCours/cours/@id");

// Sélection du plus grand id

$id = 0;

**for** ($i = 0; $i < $ids->length; $i++)

**if** ((int)$ids->item($i)->nodeValue > $id)

$id = (int)$ids->item($i)->nodeValue;

// On ajoute 1 pour créer le nouvel id

$id++;

// Création de éléments composants le cours

$nouveauCours = $document->createElement("cours");

$nouveauCours->setAttribute("id", $id);

// nom

$elemNom = $document->createElement("nom");

$valeurNom = $document->createTextNode($cours);

$elemNom->appendChild($valeurNom);

$nouveauCours->appendChild($elemNom);

// Récupération de la racine du document pour y ajouter le nouveau projet

$racine = $document->getElementsByTagName("listeDesCours")->item(0);

$racine->appendChild($nouveauCours);

// On enregistre le document

$document->normalizeDocument();

$document->save("../xml/listeDesCours.xml");

# Tests

SP se réfère au numéro de fonctionnalité décrit au chapitre 3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CT** | **SP** | **Procédure** | **Résultat attendu** | **OK** |
| **Entrer sur le site** | | | | |
| 1 | 1 | Se loguer avec un username et un mot de passe connu du LDAP de l’école, appartenant à un professeur | On arrive sur la page d’administration permettant de gérer les projets | **OK** |
| 2 | 1 | Entrer dans le site sans se loguer | On arrive dans la partie visiteur du site et on a la possibilité de rechercher et visionner les projets | **KO** |
| **Actions accessibles uniquement par un professeur** | | | | |
| 3 | 2 | Ajouter un projet | Une page s’ouvre et demande, le cours concerné, le titre, la liste des professeurs responsables, les auteurs, les mots clés, un résumé et l’archive à uploader. Le projet se voit attribuer un ID et ce dernier est stocké dans le fichier projets.xml | **OK** |
| 4 | 2 | Ajouter le premier projet dans le fichier xml | Le projet doit être créé normalement | **KO** |
| 5 | 3 | Modifier un projet existant | Le même formulaire que pour l’ajout s’ouvre, ayant les champs remplis avec les valeurs à modifier. Nous n’avons pas eu le temps de s’occuper de la fonctionnalité pour supprimer l’archive uploadée dans le but de la remplacer. | **OK** |
| 6 | 4 | Détruire un projet | Le projet est supprimé et l’archive présente sur le serveur l’est aussi. | **KO** |
| 7 | 5 | Ajouter un cours à la liste des cours | Le cours se voit attribuer un nouvel id, et il est ajouté au fichier listeDesCours.xml.  Fonctionne même si le fichier XML est vide | **OK** |
| 8 |  | Modifier le nom d’un cours | Pas implémenté | **OK** |
| 9 |  | Supprimer un cours | Pas implémenté | **OK** |
| **Actions accessibles par un utilisateur non connecté** | | | | |
| 10 | 6 | Consulter un projet | Le site afficher une page présentant les différents attributs du projet et un lien pour télécharger l’archive. | **OK** |
| 11 | 6 | Télécharger l’archive qui contient le code source, et le rapport | Le téléchargement de l’archive commence. | **KO** |
| 12 | 6 | Rechercher un projet par mot clé | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK\*** |
| 13 | 6 | Rechercher un projet par auteur | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK\*** |
| 14 | 6 | Rechercher un projet par id de cours | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK** |
| 15 | 6 | Rechercher un projet par responsable | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK\*** |
| 16 | 6 | Rechercher un projet par année | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK** |
| 17 | 6 | Rechercher un projet par titre | La page affiche le listing des projets répondant aux critères donnés par l’utilisateur. | **OK** |
| 18 | 7 | Visualiser l’ensemble des projets via une vue XSLT | Le site nous redirige sur une page affichant un tableau avec en haut la liste des projets et des liens pour les atteindre, et en dessous le descriptif complet de chaque projets, avec un lien pour télécharger l’archive et un autre pour revenir en haut de la page. *Attention : Cette page fonctionne de manière optimale sous IE. Download KO* | **OK** |

\*La fonction de recherche ne fonctionne que si la chaine recherchée apparait dans le premier sous élément du fichier XML, exemple « premier auteur de la liste », « premier mot clé de la liste »

# \*Reste à faire

* Espace pour les utilisateurs non connectés à l’application
* Ajouter les fonctions modification/suppression sur les cours
* Corriger la fonction de recherche pour que tous les champs soient pris en compte
* Ajouter une liste des branches pour la recherche
* Rendre les liens vers les archives depuis l’XSL fonctionnel

# Problèmes rencontrés

Dans un premier temps nous avons du nous familiariser avec Google Web Toolkit, afin de maitriser suffisamment l’outil pour le manier correctement. Après la période d’acclimatation nous avons enfin pu nous lancer dans le développement des interfaces WEB. Du coté PHP il y a eu moins de problèmes car le travail était plus classique et déjà connu des développeurs. Concernant les opérations concernant les fichiers XML, nous avons pu rapidement démarrer car nous avions toutes les clés en mains pour faire le travail.

[A COMPLETER avec vos problèmes ou à corriger si ce que j’ai mi n’est pas arrivé]

# Conclusion

Ce projet à été une excellente occasion pour nous de réaliser quelque chose de concret avec les fichiers XML. Nous avons aussi pu découvrir la GWT (Google WebToolkit) qui permet très rapidement de réaliser de très belles interfaces graphiques, tout en gardant le même style de programmation qu’en Java.

Certes nous aurions pu aller plus loin dans ce projet, mais le temps a été l’ingrédient principal qui nous manquait.

[A COMPLETER si tu veux ^^]

# Annexes

Fichier de test projet.xml + XSD associé

Fichier de test listeDesCours.xml + XSD associé

Fichier de transformation XSL