

# LIVRET DE MODULES

PRÉPARÉ PAR : BOULEALF SOUFIANE

ENCADRÉ PAR : M. ABDELALI ED-DBALI

UNIVERSITÉ D'ORLÉANS

JUIN 2013

## Remerciement

Je souhaite tout d'abord remercier particulièrement :

- Mon maitre de Stage M. AbdelAli Ed-dbali pour m'avoir encadré et avoir été à l'écoute pendant toute la durée du stage.
- L'ingénieur technicien M. Naly Raliravaka et Mm. la directrice du département Catherine Juilé-Bonnet , de nous avoir fournit les outils et un environnement de travail.

# SOMMAIRE

INTRODUCTION	6
Présentation de l'Entreprise	6
Positionnement du Stage	7
Cahier des Charges	7
Livret de Module	8
<i>L'introduction du Livret de module</i>	9
<i>Détails du module</i>	9
Matière Première	10
CONCEPTION DE LA BASE & MODIFICATION DE MATIÈRE PREMIÈRE	11
contraintes B.D.D	11
Modèle Conceptuel de Données	12
Structure du projet «T.X.E.B»	12
Tests préliminaires	13
Modification du projet «T.X.E.B»	13
<i>Transformation TEX =&gt; Xml</i>	13
<i>Ajout De La Couche Composante Et Filière</i>	14
Expressions Régulières	14
Exploitation Du Code Apogee	14
XSL	15
<i>Exécution De Requêtes</i>	15
Divers	16
<i>Rapport de Stage</i>	3

<b>CONCEPTION DE L'INTERFACE WEB</b>	<b>16</b>
<b>Outils Informatique Utilisés</b>	<b>16</b>
<i>Programmation orienté objet PHP</i>	16
<i>Framework JQuery</i>	17
<i>Kit CSS : Framework Bootstrap Twitter</i>	17
<b>Conception Outils de Gestion B.D.D</b>	<b>17</b>
<i>Connexion à la B.D.D</i>	17
<i>Gestion des Tables</i>	18
Classes Instanciable	18
Classes Outils	18
<i>Exception et Gestion des Erreurs</i>	19
<b>Pages de Gestion &amp; de Navigation</b>	<b>19</b>
<i>Structure basique des pages</i>	20
<i>Interaction Avec Le Tableau</i>	21
Double Clique	21
Simple Clique	21
La Page de Gestion de Module	23
Sous Menu	23
<i>Traitement des Formulaires</i>	24
Messages de Confirmation et d'erreurs	24
<b>Génération de Latex</b>	<b>25</b>
<i>Interface utilisateur</i>	25
<i>Détails Modules</i>	25
<i>Maquette des Semestres</i>	26
<b>Avancement et Problèmes Rencontrés</b>	<b>27</b>
<b>Alimentation de la base de données</b>	<b>27</b>
<b>interface</b>	<b>27</b>

Conclusion	28
Annexe	29
<i>Structure d'un module en Latex</i>	29
<i>Structure du tableau de semestre</i>	29
<i>Fonction JQuery Permettant de récupérer les valeurs de Variables Get</i>	31

# INTRODUCTION

Étudiant en dernière année de Licence Informatique à Orléans, et afin de valider celle-ci, j'ai effectué un stage, d'une durée de 3mois, au sein du département informatique de l'université d'Orléans.

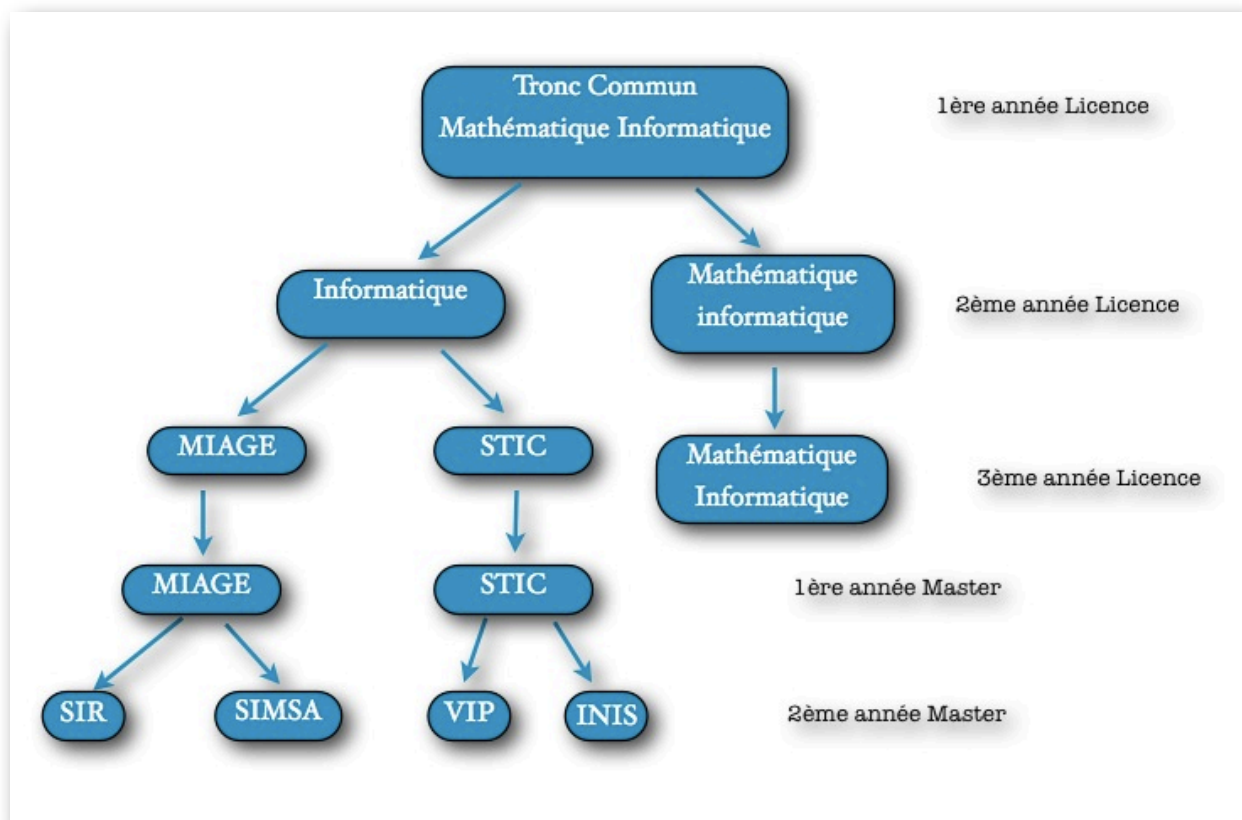
## PRÉSENTATION DE L'ENTREPRISE

Le département informatique d'Orléans compte 500 à 550 étudiants, avec 25 enseignants-chercheurs permanents, 40 vacataires du monde socio-économique et une dizaine de contractuels doctorants et ATER (Attachés Temporaires d'Enseignement et de Recherche).

Directrice : Catherine Julié-Bonnet.

Il fait partie de l'UFR sciences, qui vient de fusionner cette année avec l'UFR STAPS en un UFR Collegium Sciences et Techniques.

Les différentes formations proposé par le département sont :



## **P O S I T I O N N E M E N T   D U   S T A G E**

Le stage a pour but d'aider les étudiants, les employés des différents départements de la faculté à avoir un support. Ce support fournit un plan ou une maquette de modules en fonction de leurs filière. C'est le livret des modules.

Ce livret pourra être personnalisé selon les souhaits de chacun, ainsi nous pourrons avoir un livret de modules de la composante de Science par exemple, ou de plusieurs filières etc..

Pour cela une base de données qui gèrera les modules avec leurs composantes, filières et promotions, est nécessaire.

Ce projet est en réalisation depuis quelques années sous la direction de M. Ed-dbali. En effet une structure du livret a été défini avec des fichiers latex.

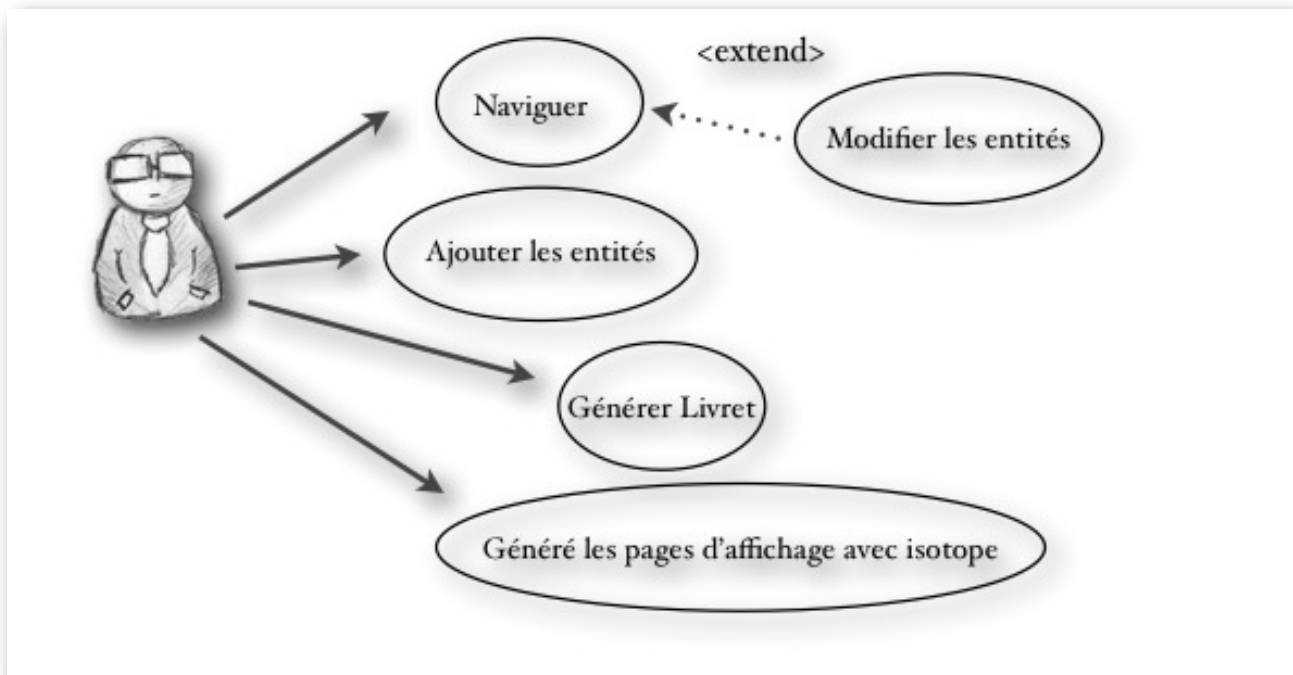
Le but donc est d'exploiter ces données, permettre de les gérer et de générer les livrets grâce à une interface web.

## **C A H I E R   D E S   C H A R G E S**

La mission qui m'a été confié, consiste à créer une interface permettant de gérer une base de données. Celle-ci contiendra les modules et unités d'enseignements des différentes facultés avec leurs détails. Elle permettra aussi de :

- Générer le livret des modules en fonction de leurs promotion, filière ou composante. (base du projet)
- Générer des pages html utilisant la Bibliothèque Isotope de JQuery, pour afficher des modules en fonction de leurs filière ou promotion.

Ces fichiers ainsi générés, auront pour but d'ajouter plus d'informations et de ressources au site internet de l'Université d'Orléans.



## LIVRET DE MODULE

Le livret de module est construit à partir de fichiers latex. Ces derniers sont structurés de manière à avoir la page de garde, le préambule et les détails des modules séparés dans des sous-fichiers, la page de garde et le préambule étant l'introduction du livret.

L'intérêt de procéder ainsi est de faciliter la modification de la structure du livret (ajouter préambule par promotion ou par semestre par exemple).

Pour plus de détails de la structure de ces fichiers latex voir annexe.



## 1. L'INTRODUCTION DU LIVRET DE MODULE



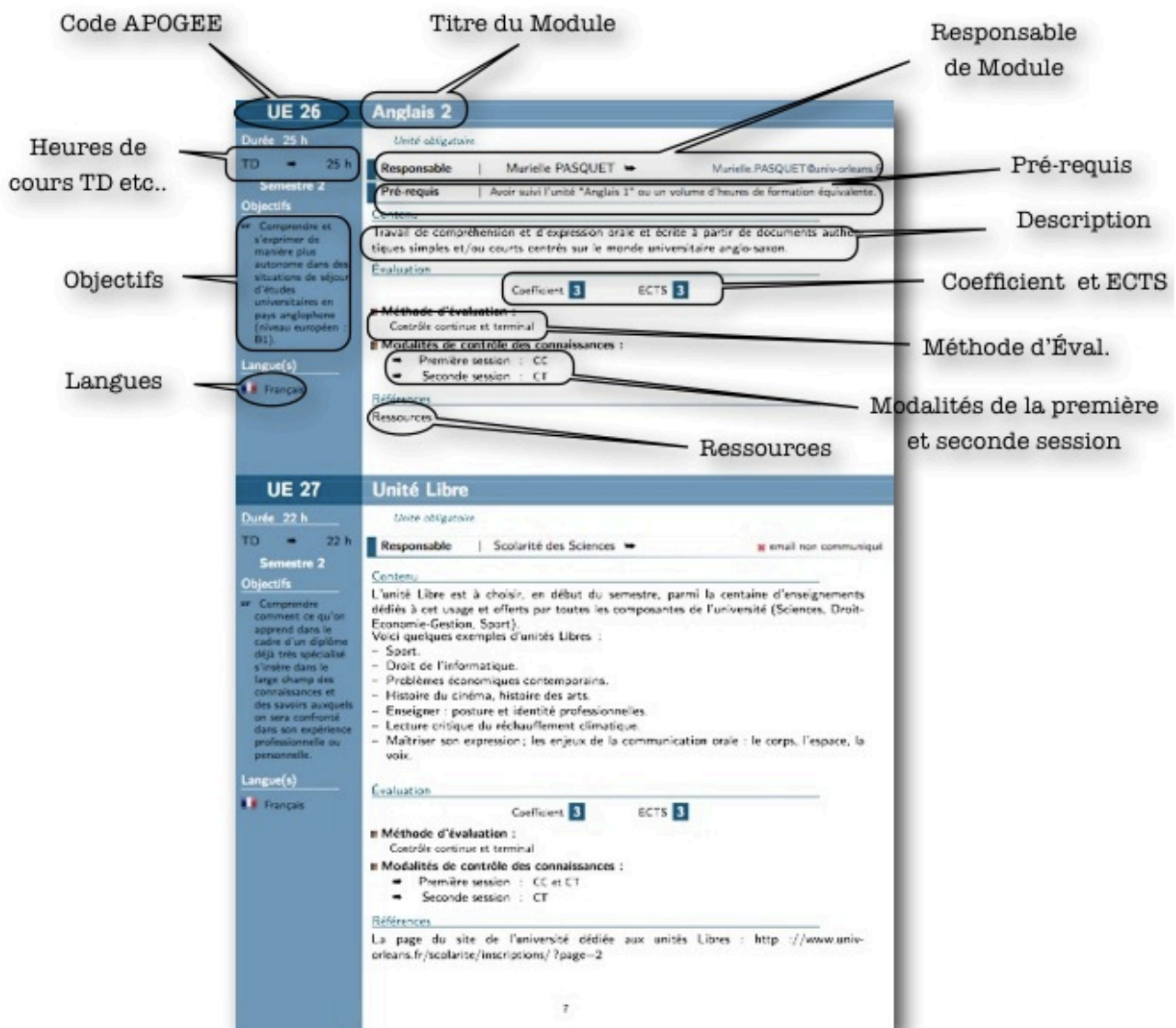
Celle-ci est divisé en deux parties (cf. l'aperçu) :

- Page de garde : Titre de la filière
- Préambule de la Filière : Un texte d'introdutif avec les objectifs de la filière, s'ajoute deux sous-parties : la fiche du responsable (carré jeune avec photo) et la maquette ou tableau des différents semestres.

## 2. DÉTAILS DU MODULE

Ensuite arrive le détail des modules, on les affiche par block, chaque block affiche les informations disponible (on affiche pas les sections vides).

Les différentes caractéristiques d'un module sont illustrées et mises en évidence sur l'exemple ci-dessous.



aperçu des détails des modules

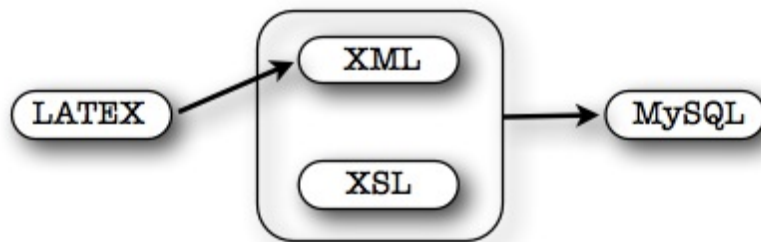
Cette partie est la plus importante, puisqu'elle permettra d'alimenter la B.D.D

## MATIERE PREMIERE

Ce projet a été entamé l'an dernier par un étudiant. Son travail consistait à créer un programme Java qui permet d'alimenter une base de données à partir de fichiers en format Latex. La structure de celle-ci (b.d.d) n'ayant pas encore été fixée. Il a commencé aussi à développer un programme java qui permet de générer des pages html (cette partie ne sera pas abordée).

Nous appellerons, tout au long de ce rapport, le programme qui alimente la base de données «TXEB» pour signifier «Transformer XML Écriture B.d.d». Celui-ci traite les fichiers de type «détails de module» (cf. au-dessus).

Pour alimenter la B.D.D, le programme «T.X.E.B» doit appliquer plusieurs transformations au fichiers Latex. Elle sont illustré par le schéma suivant :



Les principales étapes exécutées par «T.X.E.B» sont :

- Lecture des fichiers de détails des modules par block de module
- Transformation de latex en Xml
- Génération des requêtes avec un fichier XSL

Ma mission commence alors par valider ce programme. et appliquer les changements en fonction de la base de données qui sera définit.

## CONCEPTION DE LA BASE & MODIFICATION DE MATIÈRE PREMIÈRE

### CONTRAINTES B . D . D

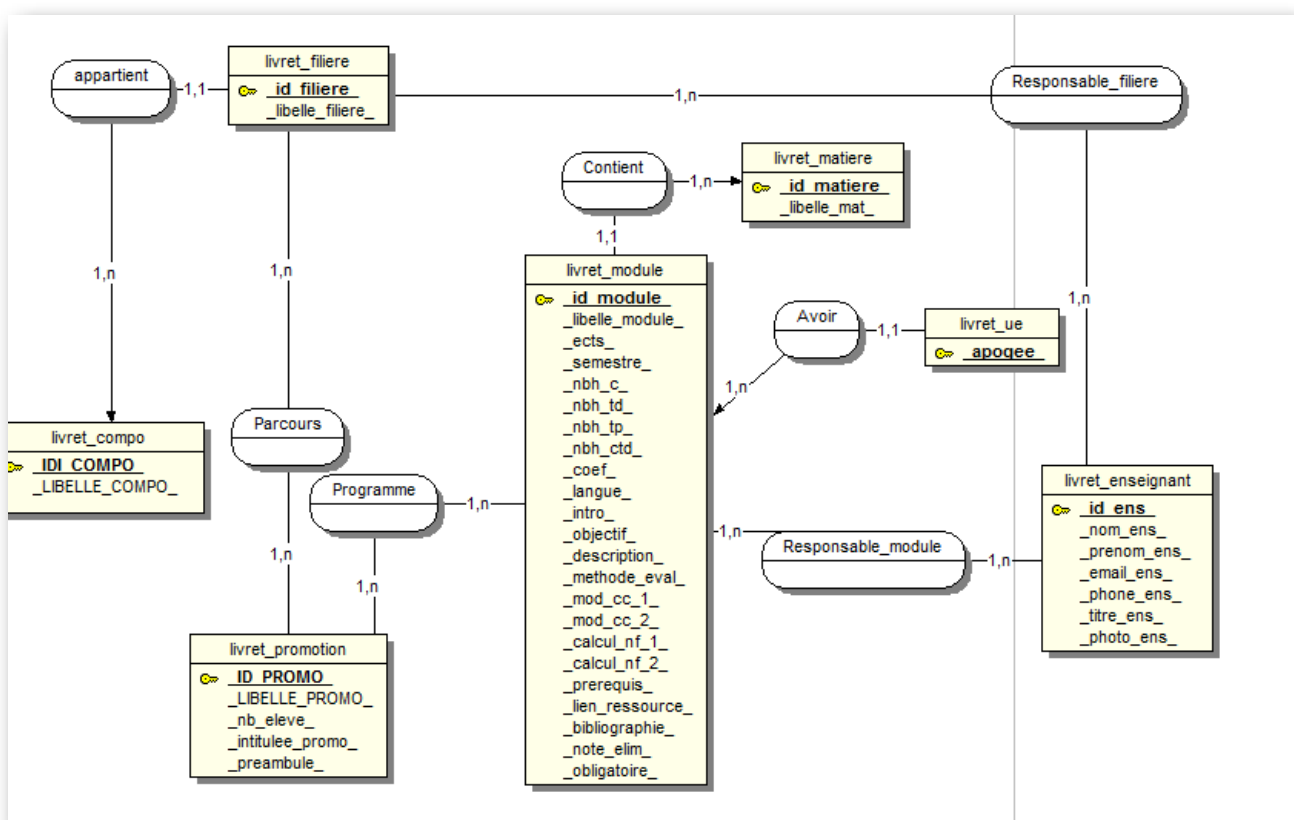
Notre base de données doit répondre à plusieurs critères :

- Une filière appartient à une composante.
- Une promotion peut être à cheval sur plusieurs filières. (Table Parcours)
- Un module peut appartenir à plusieurs promotions. (Table Programme)
- Une matière appartient à un module.
- Un module peut appartenir à plusieurs unités d'enseignements.
- Un module peut avoir plusieurs responsables.

- Une filière peut avoir plusieurs responsables.
- Un responsable est un enseignant
- Les informations ne doivent pas être dupliquées sur la base

## MODÈLE CONCEPTUEL DE DONNÉES

Nous avons donc abouti à un MCD comme celui-ci :



## STRUCTURE DU PROJET « T . X . E . B »

Le projet du programme «T.X.E.B» est constitué de trois différents packages :

- IOFichiers : contenant les classes «LectureFichier» et «EcritureFichier» permettant la lecture, transformation Xml, et écriture dans un fichier.
- ACCES\_BD : contenant la classe «Requete», cette partie sera entièrement modifiée, en raison de la reconstruction de la base de données.

- Principal : contenant la classe «Menu» qui permet de transformer XSL en MySQL et ensuite exécuter dans la base, cette classe contient le «main» du projet.

## TESTS PRÉLIMINAIRES

Les tests du programme se sont faits par deux différentes manières :

- Grâce à JUnit : La classe «LectureFichier» du projet a été testé grâce à la classe «LectureFichierTest». La fonction «lireModule» est la plus importante dans cette classe, elle permet d'associer à un fichier de détails des modules un tableau de String contenant dans chaque case un module.
- Test visuel : La classe «EcritureFichier» offre des fonctions qui transforme, à partir d'un String contenant les informations du module, celui-ci en Xml. Toutes ces fonctions ont été testé avec les fichiers des modules de notre département. les transformations semblait fonctionner correctement.
- La classe «Requete» n' a pas été testé, puisque le procédé d'exécution vers la base sera entièrement modifié : la transformation de Xml vers MySQL génèrait que des requêtes simples enregistré dans des fichiers texte, ensuite exécuter vers la base de données (cf. Modification XSL).

## MODIFICATION DU PROJET « T . X . E . B »

Après avoir trouvé une B.D.D adéquate à nos besoin, la modification commence par le changement du fichier XSL permettant de transformer Xml en requêtes MySQL.

Mais avant de parler XSL, je vais d'abord expliquer les modifications faites sur la partie de transformation Latex en Xml , la classe «EcritureFichier».

### 1. TRANSFORMATION TEX => XML

La structure d'un module en latex (voir annexe), inclut différents types d'attributs, séparés par une virgule et dont la valeur est situé entre deux accolades, écrit d'une manière spécifique en sortie xml :

- Les attributs en majuscule : ils sont transformés en attributs de la balise <module> en xml.
- Les attributs composée : on parle ici de l'attribut langue par exemple, qui peut contenir plusieurs langues séparées par une virgule, cet attribut est transformé en sous balise de module <Languages> composé de balise <langue>.
- Les attributs normaux : ils sont écrit en minuscule, ces attributs sont transformé en sous balise de module d'une manière normal.
- Les attributs de type texte : ces derniers sont situé entre accolades et ne portent pas d'étiquette, ils sont ordonnées respectivement : description Courte, description Longue, texte Pré-requis, objectifs, ressources et bibliographie.
- Les attributs qui contiennent des valeurs booléennes permettent de savoir si les attributs texte sont vide ou non, ils seront ignoré dans notre modification à la demande de mon maitre de stage.

Pour chaque type d'attributs est attribué une fonction dans la classe «EcritureFichier».

## **2. AJOUT DE LA COUCHE COMPOSANTE ET FILIÈRE**

Le nom des fichiers .tex de détails de modules sera de la forme suivante :

Details\_[nom de la composante]\_[nom de la filière]\_[nom de la promotion].tex

ces informations seront transmises en paramètre au fichier xsl.

### **EXPRESSIONS RÉGULIÈRES**

Après avoir reçu les informations des autres départements, l'ajout de commentaires (entre attributs texte par exemple) et d'espaces ou caractère blanc (entre accolade, entre les attributs) générerait des erreurs. La mise à jour des Patterns a permis de contourner ces problèmes.

Les attributs de type texte peuvent contenir des enumeration, itemize ou description qui permettent de lister le contenu, afin de garder cette mise en page, nous passons toutes les valeurs de ces attributs dans une fonction qui joue le rôle d'un filtre de la manière suivante :

- enumeration : on place un chiffre en début de chaque énumération suivis du caractère « - » ensuite la ligne correspondante
- itemize : pour chaque \item qui déclare le début d'un nouveau point ou ligne, on associe une étoile \* .
- description : une description est un itemize avec un titre en début de chaque point, ce dernier placé entre crochets juste après un \itemize. on lui associe donc le titre suivis de « : ».

Le principe de cette fonction consiste à récupérer grâce à la fonction «split», trois grande parties; avant le début de la structure de liste, au milieu (entre \begin{enumeration} et \end{enumeration} ), et après la liste.

la partie du milieu est encore une fois découpé, en prenant comme délimiter le signe associé à chaque type de structure de liste.

Une amélioration du temps d'exécution a été faite, en utilisant la méthode «contains» lors du test de présence d'éléments de listage à la place matches, la différence est assez importante quand il s'agit de long texte.

Pour plus de détails voir la méthode «checkFormat» de la classe «EcritureFichier».

### **EXPLOITATION DU CODE APOGEE**

le code apogee est structuré de manière à contenir des lettres significative :

- 1<sup>ère</sup> caractère : composante (S pour Science).
- 2<sup>ème</sup> caractère : ville (O pour Orléans).
- 3<sup>ème</sup> caractère : niveau LMD (L pour Licence et M pour Master)

- 4ème caractère : semestre entre 1 et 6 en Licence et 1 et 4 en Master.
- 5 et 6ème caractère : département ou grande orientation de la matière (CH pour chimie, BO pour biologie, BH pour biochimie, IF pour informatique, MA pour mathématique ...)
- 7 et 8ème caractère : numéro d'ordre.

cette réglementation est la plus récente, il existe des codes plus anciens qui sont écrit sur les 4 derniers caractères expliqués au dessus.

Grâce à la fonction «matière» de la classe «EcritureFichier», nous pouvons extraire la matière du module grâce au caractère du code apogee faisant référence à la grande orientation et ajouter celle-ci comme attribut de la balise module en xml.

Un module peut avoir plusieurs code apogee, j'ai donc développé la fonction «apogee» qui me permet d'extraire chaque code apogee et de l'ajouter comme nouvel attributs à la balise module.

### 3. X S L

Le fichier XSL permettra de transformer le contenu brut xml par des requêtes qui permettent d'insérer dans toutes les tables les lignes dont les champs obligatoires (comme le libelle ou l'adresse e-mail d'un responsable ...) qui n'existe pas déjà, l'interface web permettra de compléter les autres champs.

Le nom de la composante, filières et promotion seront fournis en paramètre au fichier XSL.

Afin de s'assurer de ne pas avoir des doublons dans la base de données, j'ai utilisé des variables Mysql pour vérifier si une composante, ou une filière par exemple existe déjà dans la base et ignorer son ajout par la suite.

Le document est divisé en deux parties :

- `<xsl:template match="/">` : Le corps de la transformation en requête, son contenu est appliqué à l'ensemble du document.

Le fichier .tex de détails de module est sensé contenir tous les modules de même promotion donc de même filière et composante. On essaie donc d'insérer dans les trois tables et leurs tables relationnelles (promotion composante filière parcours) qu'une seule fois par fichier, donc une seule transformation. Ensuite on applique le template de transformation d'un module (point suivant) pour tous les modules du fichier.

- `<xsl:template match="module">` : Contient les différentes requêtes d'insertions dans les tables susceptibles d'avoir de nouvelles informations en fonction du module : les tables module, programme, enseignant, responsable de module et unités d'enseignements (u.e).

Pour plus de détails voir le fichier transf\_requete.xsl.

### 4. EXECUTION DE REQUÊTES

Cette partie a été entièrement modifiée (la classe Requete), afin d'éviter les doublons (cf. au dessus) sur les tables de la base de données.

Cette classe aura pour but, l'exécution d'un script shell qui prend en argument un répertoire contenant des fichiers SQL, et les exécute sur la base de donnée local, notre classe contient donc deux méthode permettant ce fonctionnement :

- `execShellScript` qui permet d'exécuter un processus bash externe à partir du chemin du script et un autre paramètre, le répertoire contenant les fichiers de requêtes . Cette fonction redirige la sortie d'erreur et la sortie standard du processus vers le programme java.
- `convertStreamToString` : comme son nom l'indique, permet la conversion de Stream en String, ceci pour pouvoir afficher les résultats des différentes sorties du processus lancé sur la sortie standard du programme.

On peut lancer le script (disponible en annexe) directement sur le terminal en lui fournissant le répertoire des fichiers de requête, en cas de non fonctionnement de cette partie.

Les requêtes anciennement enregistré dans un fichier texte, sont maintenant enregistré dans un fichier .sql par fichier de détails.

## 5. D I V E R S

Divers modifications ont été appliqué afin d'optimiser le fonctionnement de l'application java :

- Amélioration de la gestion d'erreur.
- Lecture et recherche de fichiers de détails module à partir d'un dossier, et recherche récursive dans les sous dossiers.
- Amélioration du format de sortie XML grâce à la fonction «`prettyFormat`» de la classe «Menu». Celle-ci utilise la librairie Jdom, qui offre des outils de manipulation de xml.  
Cette amélioration se caractérise par l'enlèvement des caractères blanc inutiles, l'encodage en UTF-8 de tous l'ensemble du document. Tous ceci afin d'éviter les erreurs lors de la transformation en requête et l'exécution sur la base de données.

# CONCEPTION DE L'INTERFACE WEB

## OUTILS INFORMATIQUE UTILISÉS

### 1. PROGRAMMATION ORIENTÉ OBJET PHP

La programmation orienté objet ou P.O.O offre des avantages qui permettent de faciliter la programmation en php, et la rendre plus intuitive.

En effet la facilité de l'organisation, réutilisation du code et l'utilisation des ensembles cohérents appelé



objet, rend la modélisation des problèmes plus intuitive. Les programmes en POO sont facilement concevables car ils décrivent des entités comme elles existent dans le monde réel.

Du point de vue de la programmation l'orienté objet permet d'écrire des programmes facilement lisibles avec un minimum d'expérience, de taille minimale et la correction aisée. Ces programmes sont, de plus, souvent très stables. Sans oublier la gestion d'erreur plus efficace grâce aux exceptions.

Du point de vue de sécurité, il est facile de sécuriser le programme en interdisant ou autorisant l'accès à ces objets aux autres parties du programme.

## **2 . F R A M E W O R K   J Q U E R Y**

Le framework javascript JQuery nous permettra de parcourir et manipuler le DOM (l'arbre des éléments HTML), gérer les événements et interactions de l'utilisateur avec l'interface, ajouter des effets et animations visuels, gérer les Styles CSS et attributs des balises HTML.

Jquery est un framework multi-navigateurs :   IE6+      FF2+      Safari3.0+      Opera 9.0+  
Chrome

L'avantage par rapport aux autres frameworks de javascript est que JQuery peut être utilisé avec d'autres frameworks, il n'entre pas en conflit.

## **3 . K I T   C S S   :   F R A M E W O R K   B O O T S T R A P   T W I T T E R**

Afin d'avoir un affichage adéquat, le framework Bootstrap de Twitter est une puissante boîte à outils qui offre multiples avantages :

- La facilité de l'utilisation.
- Un framework bien documenté.
- Offre des plugins JQuery de qualité.
- Prise en compte du responsive.

## **C O N C E P T I O N   O U T I L S   D E   G E S T I O N   B . D . D**

Tous les fichiers de gestion de base de données sont regroupés dans le dossier «Tools» du site web.

### **1 . C O N N E X I O N   À   L A   B . D . D**

Pour pouvoir se connecter à la base de données, j'ai créé une classe «Connexion» (voir fichier connexion.php) qui contient comme attributs :

- Les différents champs nécessaires à la connexion vers la base de données, notamment «user», «password», «host» et le nom de la base de données.
- Une instance PDO construite à partir des attributs cités plus haut.

Cette classe permet de protéger l'accès à ces attributs à travers des «Getters» et des «Setters».

## 2. GESTION DES TABLES

Pour chaque table de la base de données j'ai associé un fichier de gestion, ces fichiers sont divisés en deux parties distinctes :

RM : Toutes les méthodes (d'instance ou de classes) qui requièrent l'accès à la base de données, prennent un paramètre de type PDO.

### CLASSES INSTANCIABLE

Ces classes incarnent ou modélisent la ligne en fonction de la table, les champs de la table deviennent ses attributs.

On a pas besoin de spécifier les attributs de ces classes instanciables, car la fonction «fetchAll» de «PDO» avec comme paramètre «PDO::FETCH\_CLASS» et le nom de la classe dans laquelle on souhaite charger la ligne, permet de créer les attributs dans la classe (visibilité public).

Elles offrent des fonctions d'instances, de classes, qui permettent de :

- Générer différents formulaires de modification ou d'ajouts. Ces derniers sont compactés dans un modal qui sera affiché grâce au JQuery.
- Tableau de la table avec pagination pour les tables qui peuvent contenir beaucoup de lignes (modules, enseignants par ex...). Les lignes du tableau sont générées par une autre fonction «generateLine».

Dans le cas de table relationnel comme la table de parcours ou programme ou encore responsable, elle permet en plus de faire la liaison avec les autres tables et récupérer les détails des champs des tables en relation avec celle-ci sous forme d'objet encore une fois.

### CLASSES OUTILS

Ces classes sont marquées par le mot «Tools» à la fin du nom. Elles contiennent que des méthodes de classes permettant d'exécuter un certain nombre de requêtes indispensables. L'exécution de ces requêtes se fait grâce à la fonction «prepare» de «PDO». La requête préparée contient des points d'interrogations aux

endroits nécessitant les valeurs concrètes des champs. Pour fournir ces dernières, on utilise la fonction «bindParam» ou «bindValue».

L'intérêt d'utiliser ces fonctions est de donner plus de lisibilité au code, et de permettre d'ajouter des restrictions en fonction du type et la taille de la valeur.

Les fonctions sont divisé en trois catégories :

- Les méthodes GET : permettent en fonction du second paramètre (le premier étant toujours PDO) qui correspond à l'id, le libelle ou encore l'e-mail de l'enseignant par exemple, de récupérer les lignes correspondantes dans la table, de transformer le résultat en objet, et placer tous les résultats dans un tableau.
- Les méthode EXIST : permettent de savoir si le paramètre existe ou non dans la table (id, libelle, e-mail etc ...).
- Les méthodes d'insertion, modification et de suppression.

### **3. EXCEPTION ET GESTION DES ERREURS**

Afin d'éviter les erreurs au niveau de la base de données, Une Exception est généré dans les cas suivants :

- Invalidité de type de paramètres.
- La fonction «execute» de «PDO» retourne un booléen permettant de confirmer l'exécution ou non de la requête, dans le cas échéant une exception est levé.
- Pour les clés étrangères, nous vérifions avant insertion ou mise à jour, que l'élément existe déjà dans la table de référence, on lève une exception dans le cas échéant.

Chaque méthode inscrit sont nom et le nom de sa classe sur le message d'erreur.

Pour les exceptions captées :

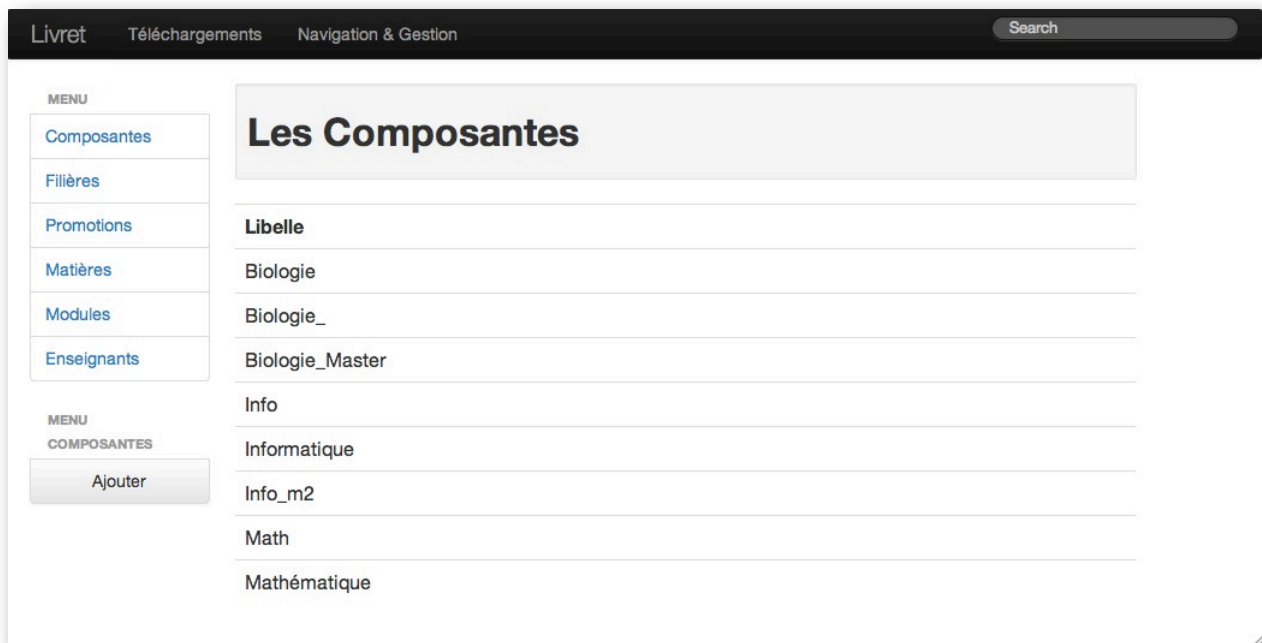
- La méthode ayant capter cette exception, récupère le message d'erreur.
- Ajoute sa signature.
- Lève une exception.

Tout ceci afin d'aboutir à un système similaire ou presque au «PrintStackTrace» de java.

## **PAGES DE GESTION & DE NAVIGATION**

L'interface de gestion devra répondre à des critères de simplicité. On a donc choisi d'afficher les informations correspondantes à une table dans un tableau interactif; il génère des formulaires en fonction de l'action de l'utilisateur.

## 1. STRUCTURE BASIQUE DES PAGES



Pour chaque ligne du menu de la barre de navigation latérale, est associé un fichier se terminant par le mot «Manager», et tous ces fichiers sont enregistrés dans le dossier «Manager».

Les pages de gestion sont de la forme suivante :

- Parties statiques : Menu principal (en haut) et la barre de navigation latérale.
- Parties dynamiques : Le tableau et son titre généré par la fonction static «genereTable» disponible dans toutes les classes instanciables (cf. partie classes instanciable au-dessus).  
Le menu personnalisé (juste en dessous de la barre de navigation latéral) change en fonction de chaque page de gestion.

Chaque ligne du tableau est en réalité une instance d'objet, les lignes du tableau contiennent un id caché, celui-ci nous permettra de faire la liaison entre les lignes et leurs formulaires de modifications.

Pour chaque instance on affiche les formulaire de modifications et d'ajouts cachés «modal».

## 2. INTERACTION AVEC LE TABLEAU

On peut interagir avec le tableau par différentes manières. Le double clique est disponible sur tous les tableaux des différentes pages de gestion et pour toutes les lignes. Le clique simple est réservé pour afficher un «popover».

### DOUBLE CLIQUE

Il est disponible sur tous les tableaux, ce clique permet d'afficher le formulaire de modification de la ligne sélectionnée. Il contient en plus les champs non affichés dans les colonnes de la table.



Ces formulaires donnent la possibilité de changer directement les champs de la table, ou de supprimer la ligne qui a été sélectionné.

Grâce à JQuery, nous pouvons détecter les cliques sur les lignes du tableau. Nous pouvons donc afficher par ce conséquence le modal (formulaire) correspondant.

L'appel du bon formulaire de la ligne, se fait par le biais de l'ajout de l'id au nom du formulaire,

### SIMPLE CLIQUE

Il permet d'afficher ou cacher un popover contenant des information selon les pages suivantes :

- Promotions : Filière associé à la promotion.
- Filières : Responsable de la filière.
- Modules : Responsable du module.

Il est disponible que pour ces pages.

## Les Filières

Libelle	Composante
m2	Informatique
Licence-biobio	Biologie
Master-BBMB	Biologie

Responsable(s)



## Les Promotions

Libelle	Nombre d'élèves	Intitule	Préambule	Epilogue
INFO	0			
m1	0			
MIAGE	0			

Filières

[Licence3](#)



Pour chaque ligne (donc pour chaque objet) est associé un popover. Celui-ci est généré pour chaque instance par la fonction «generePop».

Le popover contient les éléments en relation avec la ligne, et un bouton d'ajout. Tous des balise <a>. Pour modifier ces éléments, il suffit de cliquer pour avoir un formulaire de modification.

Pour faire la liaison entre le contenu du «popover» et leurs formulaires, nous utilisons la méthode GET. Grâce à la fonction de JQuery fournit en annexe, nous pouvons extraire les valeurs de ces variables. Ainsi nous pouvons détecter le formulaire correspondant.

## Les Modules

Libelle	Matière	Semestre	Cours	TD	TP	Cours-TD	ECTS	Coef	U.E	Promo
Pratiques des contraintes	UNDEFINED	0	20	15	0	0	4	4	Aucun +	MoCaHP +
Problèmes et méthodes de la science économique	Science Economique	1	24	0	0	0	3	3	1SE02 +	L1-MASE +
Lois fondamentales d'électricité linéaire	Physique	1	0	0	12	24	3	3	1PY03 +	L1-MP +
Anglais scientifique 1	Anglais	1	0	24	0	0	3	3	SOM1AG41 +	M1-S1 +
Atomistique et thermodynamique	Chimie	1	17	16	15	0	5	5	SOL1CH02 SSL1CH02 +	L1 +

Un module est en relation avec plusieurs tables :

- Enseignant : responsable du module
- Unité d'enseignement
- Promotion : un module fait partie du programme d'une promotion ou plusieurs.

Dans cette page nous réservons le simple clique pour afficher un «popover» avec le responsable du module. il reste à gérer les deux autres relation avec les tables, sauf que des «popover» ne pourrions pas être réutilisé.

Pour se faire nous avons ajouté deux colonnes dans le tableau de module, ayant le même fonctionnement que le «popover» et de la même façon.

### S O U S M E N U

Le sous-menu contient un bouton d'ajout. celui-ci est géré par JQuery et permet d'afficher le formulaire caché d'ajout.

La page promotion a un bouton d'ajout de parcours, générons un formulaire d'ajout de parcours quelconque.

### 3. TRAITEMENT DES FORMULAIRES

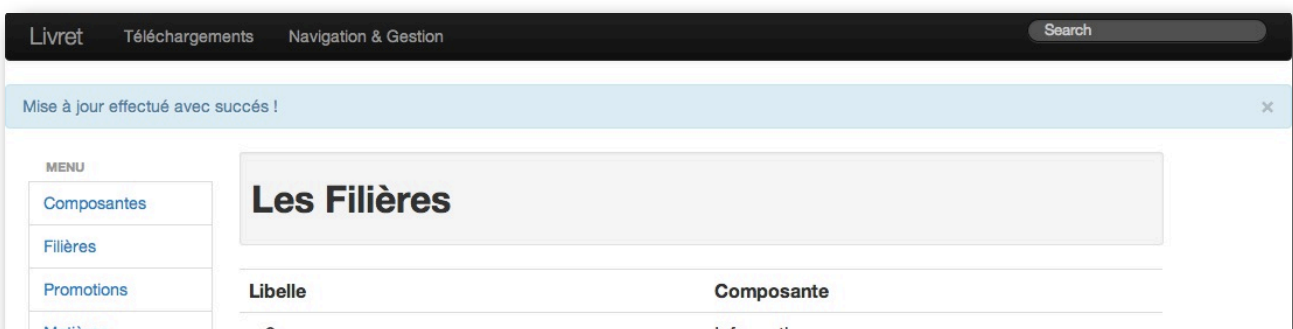
Les champs des formulaires permettent de faire des contrôles basiques afin de s'assurer de la validité des informations, ceci grâce à html5 :

- Utilisation de l'attribut pattern afin de contrôler les champs qui devrait contenir uniquement des chiffres.
- Utilisation de balise input de type «color» : La base de données contient des couleurs en format RVB séparé par une virgule. La balise de type «color» permet d'avoir les couleurs en HEX . le fichier «color.php», disponible dans le répertoire «Tools», contient deux fonctions qui permettent de faire la conversion entre les deux.
- Utilisation de balise type e-mail, phone ..
- Pour les champs faisant référence à d'autres tables (clés étrangères), nous utilisons la balise «select»
- Les photos des enseignants sont copiées vers le répertoire «Pictures» du serveur.
- Les champs indispensables à l'envoi du formulaire sont marqués grâce au mot clé required.

Les formulaires de modifications et d'ajouts sont traités dans les pages de gestions elles mêmes, au début du fichier. Ceci pour pouvoir communiquer un message situé en haut de page et permettant de confirmer l'action ou de transmettre le message d'exception.

### MESSAGES DE CONFIRMATION ET D'ERREURS

Le fichier «alert.php» situé dans le répertoire «Tools» permet d'offrir plusieurs différentes méthodes qui génère des alertes avec le style de Bootstrap, avec le message fournit en paramètre.



Il est nécessaire d'initialiser les alertes avec JQuery, dans toutes les pages.

Pour plus d'informations voir la documentation du Bootstrap.

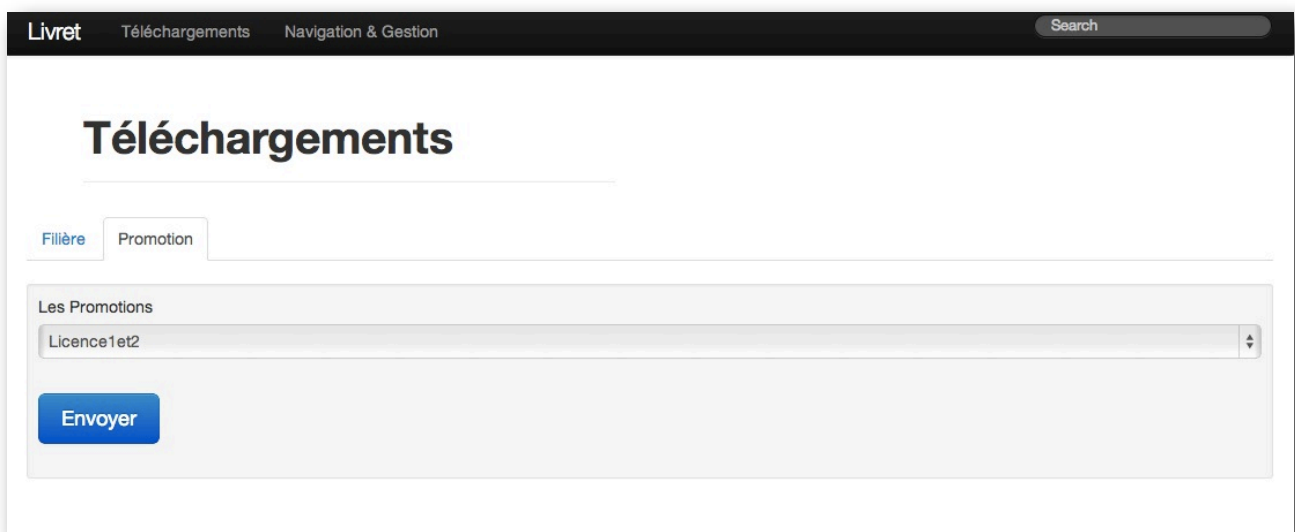


## GÉNÉRATION DE LATEX

Après avoir fini de programmer la partie de gestion de la base de donnée place à la génération de Latex, cette partie a été commencée, et est encore en construction.

Nous allons donner possibilité de générer les fichiers de modules, à partir de la promotion et du semestre.

### 1. INTERFACE UTILISATEUR



Cette interface permet de choisir la promotion pour laquelle l'utilisateur souhaite générer le fichier de détails de modules, et la maquette des semestres. En effet la validation du «select» permet de valider l'id de la promotion souhaité, pour construire à partir de celle-ci l'objet de promotion et exécuter ses fonction permettant de construire nos fichiers.

### 2. DÉTAILS MODULES

Pour respecter la principe de programmation orienté objet, chaque objet doit offrir des fonctions de génération de latex, à commencé par la classe de plus bas niveau : la classe «module».

En effet la promotion, et par l'intermédiaire de la classe «programme» permet d'avoir tout le programme de la promotion. Ainsi nous pouvons appeler la fonction qui permet de transformer un module en format latex (voir annexe pour le format), pour tous les modules de la promotion. Il s'agit de la méthode «genereLatex»

La génération de détails de modules dans la classe «promotion» se fait de la manière suivante :

- La fonction «*genereTexDtlF*» permet de récupérer le programme de la promotion ainsi que sa composante et filière. Ces informations nous permettrons de nommer et créer le fichier de détails vierge. Ce fichier est rempli avec la fonction qui suit.
- la fonction «*genereLatex*» permet d'écrire dans le fichier créé en fonction des semestres sélectionnés, les modules en format latex. On ajoute les couleurs de la promotion en début de page (nous rappelons que celle-ci est stocké dans la base de données).

### 3. MAQUETTE DES SEMESTRES

Pour générer le fichier de semestre nous utilisons le même procédé, cependant la forme n'est pas la même, voyons voir de plus près :

	Intitulé	ECTS	CM	TD	TP
SEMESTRE 3	Algorithmique 3 (programmation orientée objet) et programmation	6	24	36	
	Bases et Internet de données	5	12		24
	Atelier 2 de l'informaticien	4	12		24
	Architecture des ordinateurs	4	12	12	6
	Applications de l'algèbre	5		48	
	Anglais 3	3		25	
	Unité d'ouverture	3		22	
	Projet professionnel personnel et	2	12		
SEMESTRE 4	Programmation fonctionnelle	6	24	36	
	Algorithmique des structures discrètes et combinatoire	6	24	36	
	Projet (Conception et projet) informatique 2	5	12		24
	Probabilités	5		48	30
	Anglais 4	3		24	
	Bases comptable du système	5		30	
	Programmation impérative	5	12	20	

Nous avons deux niveaux de lignes dans ce tableau :

- Premier niveau (3 lignes) : qui définit la limite des semestres, qui est de 6 modules par semestre. La ligne des titres des colonnes est du même niveau.
- Second niveau : les lignes définissant les différents modules du semestres

La première colonne (de gauche à droite) subit une rotation de 90°

Une ligne sur deux est en blanc et avec une couleur de police sombre.

La structure du document Latex de cet image est fournit en annexe avec quelques explications.

La génération du tableau de semestre en latex se fait d'une manière similaire à celle des détails de modules :

- La fonction «`genereTexTabF`» : permet de récupérer le programme de la promotion ainsi que sa composante et filière. Ces informations nous permettrons de nommer et créer le fichier de tableau de semestre vierge.
- La fonction «`genereTabSemTex`» : permet de générer le tableau de semestres. En fonction du nombre de semestres passés en paramètre, elle écrit sur le fichier fournit en paramètre, les modules du semestre sélectionné.

## AVANCEMENT ET PROBLÈMES RENCONTRÉS

### ALIMENTATION DE LA BASE DE DONNÉES

La modification du projet a pris plus de temps que prévus. Les commentaires et le rapport de stage manquants, ont eu leur effet, sans parler du temps d'adaptation et d'exploration des outils XSL, XML et Latex que j'ai utilisé pour la première fois.

Des erreurs de saisis dans les fichiers de détails de modules, ont été trouvé, cela a en quelque sorte rendu le debuggage du programme java «T.E.X.B» plus difficile.

À présent le programme fonctionne correctement, les requêtes sont correctement généré.

Voici quelques exemple d'améliorations faites :

- Ajout des enseignants en premier lieu, afin de permettre d'ajouter les responsables par la suite.
- Suppression des doublons lors de l'ajout des informations.
- Amélioration de lisibilité du code.

### INTERFACE

Ce projet a été réalisé avec des framework utilisé pour la première fois. En effet la documentation et l'assimilation des fonctionnalités a ralentie le projet. L'intérêt étant de fournir un travail qui pourra être poursuivis dans les années à venir.

L'assimilation du fonctionnement du système de module de notre faculté n'a pas été évidente. En effet les documentations, et les bases étaient quelques fois intrigante, parfois contradictoire.

Donc pour trouver une base de donnée adéquate, plusieurs changements séquentiel ont été effectués.

La taille des fichiers de base, qui gère l'accès à la base de données (tous les fichiers du dossier «Tools») a rajouté plus de difficultés, malgré l'utilisation de la POO qui permet de catégoriser les fonctions de la manière la plus méthodologique et la plus proche de la réalité.

## C O N C L U S I O N

La partie de base du projet a été faite. Nous pouvons désormais alimenter et gérer la base de donnée graphiquement sans aucune difficulté.

Faute de temps la partie de génération de livret et des pages html isotope reste en cours de développement.

L'interface pourra intégrer d'autres fonctionnalités :

- Marquer les lignes des tables de la base de donnée, qui ne sont pas entièrement ou partiellement complétées. Ceci afin d'aider l'utilisateur à remplir les tables de manière efficace.
- Ajouter une fonctionnalité de recherche permettant d'afficher les résultats sous forme de tableau interactif encore une fois.
- Ajouter des filtres sur les formulaires ayant un «input» de type «select» avec un choix de sélection très grand. Ceci afin d'optimiser la recherche dans cette sélection.

Ce stage m'a permis d'avoir une idée concrète du monde professionnel : la durée de 3 mois à temps plein, se révèle être une expérience très bénéfique dans le monde du travail, ceci est ma plus longue insertion, j'ai pu dépasser le stade de formation, et concrétiser mes acquis par un vrai projet. Ce stage a été une vraie expérience professionnelle puisque j'ai été amené à travailler de manière autonome, tout en gardant une communication avec le responsable du sujet.

Ce stage a répondu à mes attentes aussi bien au niveau professionnel qu'en terme de compétences et savoir-faire : malgré le fait que je n'ai pas pu arriver au but final de l'application. J'ai pu voir différents aspects de programmation, de la mise à niveau d'application (première partie modification de matière première) à la conception entière d'une application.

# ANNEXE

## STRUCTURE D'UN MODULE EN LATEX

```
\module[codeApogee={UE 23},
titre={Modélisation},
CODEUE={1},
COURS={},
TD={},
TP={},
CTD={24},
TOTAL={24},
SEMESTRE={Semestre 2},
COEFF={3},
ECTS={3},
MethodeEval={Contrôle continue et terminal},
ModalitesCCSemestreUn={CC et CT},
ModalitesCCSemestreDeux={CT},
%CalculNFSessionUne={\frac{(CC+2*CT)}{3}$},
%CalculNFSessionDeux={CT},
NoteEliminatoire={},
nomPremierResp={Frédéric LOULERGUE},
emailPremierResp={Frédéric.LOULERGUE@univ-orleans.fr},
nomSecondResp={},
emailSecondResp={},
langue={Français},
nbPrerequis={0},
descriptionCourte={true},
descriptionLongue={true},
objectifs={true},
ressources={true},
bibliographie={false}}
{
Unité obligatoire.
}
{
Présentation simplifiée de techniques de modélisation et de leur mise en \oe uvre.
}
{}
{\begin{itemize}
\ObjItem Première approche des principes de modélisation des problèmes informatique.
\end{itemize}
}
{Ressources}
{Biblio}

\vfill
```

## STRUCTURE DU TABLEAU DE SEMESTRE

```

%-----%% % Personnalisation des couleurs %% ----- BLEU -----
\definecolor{couleurFonce}{RGB}{0,92,133} % Couleur du Code APOGEE
\definecolor{couleurClaire}{RGB}{100,151,186} % Couleur du fond de la bande
\definecolor{couleurTexte}{RGB}{255,255,255} % Couleur du texte de la bande
%-----

\arrayrulecolor{couleurFonce} % Couleur des lignes séparatrices du tableau
\renewcommand{\arraystretch}{1.2} % Coeff appliqué à la hauteur des cellules
%\rowcolors[1]{\hline}{\ligneDébut}{couleurPaire}{couleurImpaire} % Alternance de couleur (need package xcolor)
\begin{tabular}[c][m{6cm}][cm{1cm}][cm{1cm}][cm{1cm}][cm{1cm}][cm{1cm}]{
\cline{2-6}

&
\cellcolor{couleurFonce} \color{white}\bfseries Intitulé \e & \cellcolor{couleurFonce} \color{white}\bfseries ECTS & \cellcolor{couleurFonce} \color{white}\bfseries CM & \cellcolor{couleurFonce} \color{white}\bfseries TD & \cellcolor{couleurFonce} \color{white}\bfseries TP \\ \cline{2-6}

\hline \multirow{6}{*}{\rotatebox{90}{\color{couleurFonce}\bfseries SEMESTRE 2}}
& \color{black} \mbox{Algorithmique} \mbox{2} \mbox{et} \mbox{programmation} & \color{black} 6 & \color{black} 60 & \color{black} & \color{black} \\ \cline{2-6}
& \cellcolor{couleurClaire} \color{couleurTexte} \mbox{Outils} \mbox{de} \mbox{l'informatique} \mbox{mathématiques} \mbox{pour} & \color{black} 4 & \color{black} 48 & \color{black} & \color{black} \\ \cline{2-6}
& \cellcolor{couleurClaire} \color{couleurTexte} 48 & \color{black} 3 & \color{black} 24 & \color{black} & \color{black} \\ \cline{2-6}
& \color{black} \mbox{Modélisation} & \color{black} 3 & \color{black} 24 & \color{black} & \color{black} \\ \cline{2-6}
& \cellcolor{couleurClaire} \color{couleurTexte} \mbox{Projet} \mbox{informatique} \mbox{1} & \color{black} 3 & \color{black} 24 & \color{black} & \color{black} \\ \cline{2-6}
& \cellcolor{couleurClaire} \color{couleurTexte} \mbox{Mathématiques} & \color{black} 5 & \color{black} 60 & \color{black} & \color{black} \\ \cline{2-6}
& \color{black} \mbox{Anglais} \mbox{2} & \color{black} 3 & \color{black} 24 & \color{black} & \color{black} \\ \cline{2-6}
& \cellcolor{couleurClaire} \color{couleurTexte} \mbox{Unité} \mbox{d'ouverture} & \color{black} 3 & \color{black} 24 & \color{black} & \color{black} \\ \cline{2-6}
& \color{black} \mbox{Projet} \mbox{professionnel} \mbox{personnel} \mbox{et} & \color{black} 3 & \color{black} 12 & \color{black} & \color{black} \\ \cline{2-6}
\hline
\end{tabular}

```

- Les cellules sont séparées par le caractère : &
- Pour déclarer une nouvelle ligne : \hline
- les mots placés à l'intérieur d'une cellule sont mis dans des \mbox
- Une cellule sur deux est colorée en bleu
- Pour ajouter les séparateurs de ligne : \cline
- Pour ajouter les séparateurs de colonne : \ccline

L'affichage correspondant au code ci-dessus est le suivant :

	Intitulé	ECTS	CM	TD	TP
SEMESTRE 2	Algorithmique 2 et programmation	6	60		
	Outils de l'informatique mathématiques pour	4	48		
	Modélisation	3	24		
	Projet informatique 1	3			
	Mathématiques	5	60		
	Anglais 2	3	24		
	Unité d'ouverture	3		24	
	Projet professionnel personnel et	3	12		

## FONCTION JQUERY PERMETTANT DE RÉCUPÉRER LES VALEURS DE VARIABLES GET

```
//source : http://stackoverflow.com/questions/6001839/check-whether-a-url-variable-is-set-using-jquery
//permet d'avoir les valeur d'une variable GET
$.extend({
  getUrlVars: function(){
    var vars = [], hash;
    var hashes = window.location.href.slice(window.location.href.indexOf('?') + 1).split('&');
    for(var i = 0; i < hashes.length; i++)
    {
      hash = hashes[i].split('=');
      vars.push(hash[0]);
      vars[hash[0]] = hash[1];
    }
    return vars;
  },
  getUrlVar: function(name){
    return $.getUrlVars()[name];
  }
});
```