Extranet v.2

Cahier des charges et documentation technique

Version 0.1

1er février 2017

Sommaire

Contexte

L'extranet est l'outil utilisé par les programmes 5D, IT4Net et Syni dans la gestion et l'organisation de mesures et des activités des programmes auprès des participants qu'ils accueillent. Des outils comme le gestionnaire des interventions d'IT4Net est utilisé par les usagers d'autres programmes ETSL s'y connectent quotidiennement. Les secrétaires provenant du programme CAPTA en fonction à 5D et IT4Net en font leur outil de travail principal.

La conception de l'extranet a démarré en 2007 avec une première version qui s'est amélioré d'années en années et au fil des implémentations. Il a comme objectif de faciliter les opérations de suivi des participants par l'encadrement. Se sont ajouté des outils de gestion pour les ateliers, les projets, l'emprunt de matériels notamment. Les droits d'accès ont également évolués avec le temps permettant aux participants des mesures d'être eux-mêmes acteur dans le suivi par le journal d'activité, les emprunts ou encore l'évaluation des ateliers suivis.

Les outils développés répondent principalement aux besoins du programme 5D qui est à l'origine des développements. L'extranet est également un outil qui permet de proposer un projet de développement aux participants du programme.

Les méthodes de développement utilisées depuis l'origine du projet ont quelques peu évoluées mais s'avèrent désuètes 9 ans plus tard. Sans compter que l'interface utilisateur n'a pas été suivi l'évolution des supports notamment mobile.

Par le nombre d'outils et modules mis en fonction et par son architecture, le développement de nouvelles fonctionnalités s'avèrent long et complexe. Il est dorénavant complexe de proposer à des participants d'effectuer de nouveaux développement, d'une part, parce que cela ne leur permet pas d'actualiser leur pratique et, d'autre part, par la complexité du code qui implique période d'adaptation déraisonnablement longue.

Pour ces raisons et afin de garder possible le développement et l'évolution de l'outil, une nouvelle version de l'extranet s'engage. Les outils et méthodes proposés sont ceux présentés par l'encadrement de 5D dans les ateliers qui concernent le développement Web et le Webdesign. Il s'avère que le projet peut ainsi redevenir un moyen pour les participants de développer des compétences en fonction des exigences du marché du travail dans ces domaines.

Par la complexité de la base de données et le volume des contenus qu'elle contient, il a été décidé de conserver la même structure et d'effectuer les développements en fonction. Le temps que la migration de la version une à la version deux soit complète et opérationnelle.

La transition se fera au fil des développements permettant d'utiliser les outils de la précédente version tout au long de la migration. Cette opération sera complète une fois que tous les modules définis comme encore utiles auront été intégrés dans la seconde version de l'extranet.

Références techniques

La seconde version de l'extranet utilise la même architecture de la base de données MySQL que la version actuelle de l'outil. Ceci afin de faciliter la migration et effectuer une implémentation itérative du nouvel outil.

Par conséquent, les langages sont les mêmes que pour la première version. L'architecture de développement, l'organisation des répertoires et les ressources utilisés sont revues afin d'offrir une meilleure maintenance et de proposer un outil actuel disposant de fonctionnalités adaptées notamment pour les mobiles.

Langages

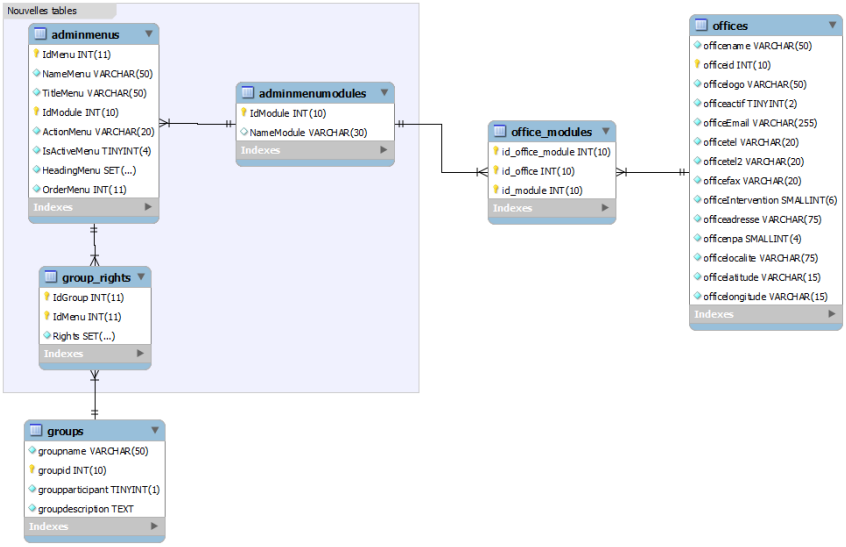
- Langage client : HTML 5, CSS 3 et Javascript
- Langage serveur : PHP 5.5 (MVC et POO), MySQL 5.6

Ressources

- Framework CSS Bootstrap 3.0
- jQuery 2.2.0
- Outils JS (datatables, dataranpicker, moment, fullcalendar)

Architecture de la base de données

La transition de la première version de l'extranet à la seconde n'implique aucun changement dans la base de données puisqu'il est question d'utiliser l'architecture de l'actuelle base de données tout au long du processus. Les tables **« adminmenus »** et **« adminmenumodules »** qui gère la composition du menu de l'administration et se greffent à la gestion des droits ont été ajoutées. La table **« group_rights »** a également été ajoutée.

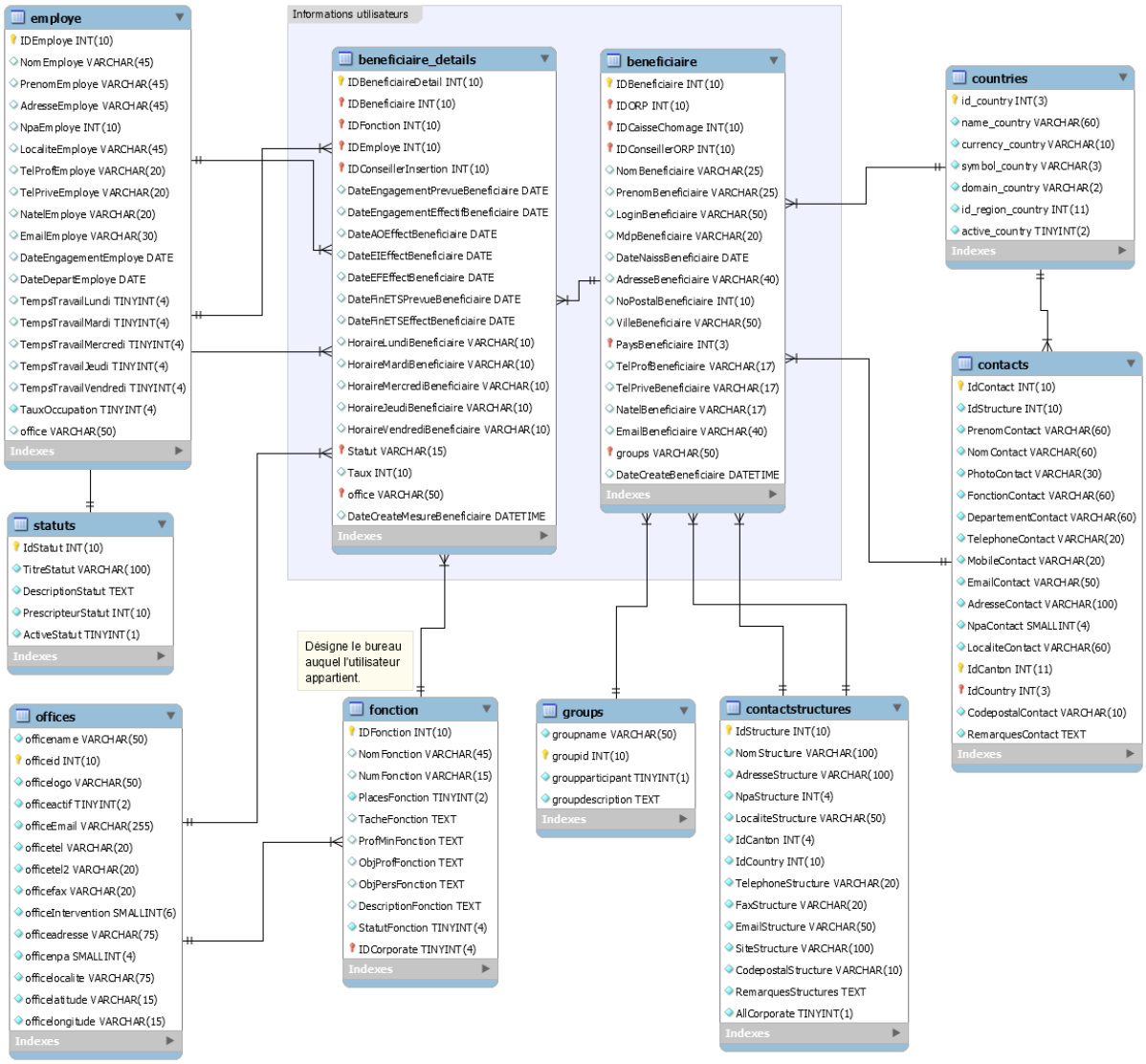


Les développements éparses de l'extranet a comme conséquence un manque de constance dans la construction des tables et champs. Certaines incohérences sont à noter en ce qui concerne la nomenclature des champs et de plusieurs tables. L'harmonisation et le respect des indications données dans la rubrique « nomenclature » de ce document ne pourra se faire qu'après que la migration de la nouvelle mouture soit complète puisqu'il est question d'utiliser l'architecture de l'actuelle base de données tout au long du processus. Il s'agira d'un futur développement tel que cela est proposé dans la rubrique « perspectives et évolutions ».

Les tables principales de la base de données sont **« beneficiaire »** et **« beneficiaire_details »** qui regroupent les informations pour chaque utilisateur. D'autres tables provenant notamment d'autres modules y sont liées pour compléter les données de l'utilisateur, particulièrement utiles pour le statut des participants.

La table **« beneficiaire »** contient les informations essentiels et communs d'un utilisateur, telles que ses coordonnées, accès au système ou groupe d'appartenance (qui définit les droits d'accès qu'il dispose dans le système).

La table **« beneficiaire_details »** recense les informations pour chaque période d'activité d'un utilisateur. Ces périodes regroupent des informations concernant notamment les dates, statuts, fonctions de l'activité. La particularité de cette table est qu'elle est autant utilisée pour tous les profils qui ont accès au système des administrateurs aux participants. Par conséquent, certaines informations ne sont pas utiles pour certains profils. C'est le cas par exemple, de l'encadrant référent, identifié dans la table lié **« employe »** que ne concerne que les participants.



Structure et organisation des fichiers

L'organisation du projet est définie selon ce qui est présenté dans les ateliers sur le langage PHP. Il s'agit d'une architecture simplifiée qui s'apparente à celles utilisées dans les « frameworks » PHP courants, tels que « Symfony » ou « Laravel ». Elle respecte une modélisation MVC (Model-View-Controller) et permet une implémentation modulaire des vues en séparant les aspects prévus pour l'interface utilisateur.

applications	Modèle et le contrôleur des modules
...	Modules...
caches	Fichiers temporaires
sessions	Sessions des utilisateurs
includes	Noyau du système
components	Modèles et contrôleurs communs
tools	Classes et outils et communs
config.ini	Fichier de configuration
...	Fichiers de gestion du système
public	Contenus visibles du site
languages	Fichiers de traduction
theme	Composants graphiques
css	Feuilles de styles : principal et outils
email	Gabarit des e-mails
fonts	Polices Web
images	Images d'interfaces
js	Scripts : principal et outils
upload	Fichiers chargés depuis l'administration
views	Interfaces communs et celles des modules
components	Interfaces communes utiles aux modules
...	Modules...
...	Interfaces communes (page, login, home, menu, ...)
.htaccess	Paramètres serveur
index.php	

Nomenclature

L'un des aspects qui a particulièrement fait défaut dans la première version de l'extranet est l'établissement d'une nomenclature qui permet de limiter la disparité de cohérence.

L'anglais est la langue à privilégier pour le nom des éléments définis dans le projet. Ceci permet de rester cohérent avec les outils tiers utilisés et correspondre avec ce qui s'emploie dans la programmation.

Les commentaires et documentations peuvent être en français pour faciliter l'accès à l'information auprès de programmeurs et designers francophones. Pour cette raison, elles permettent une meilleure compréhension de concepts et informations complexes ou abstraites.

Les références ou sources d'informations complémentaires en lien avec un outil tiers par exemple, sont dans la langue disponible quelle qu'elle soit.

Tableau avec exemple de la nomenclature à respecter dans ce projet.

Variables	Inscrites en minuscule. Une majuscule au début de chaque mot suivant afin de faciliter la lecture de la variable. Le nom d'une variable peut commencer par une majuscule pourvu qu'il s'agisse d'une correspondance avec le nom d'un autre élément facilitant son identification (nom du champ d'une base de données, par exemple).
	
	Un tiret bas simple (_) est employé devant une variable servant d'attribut privé ou protégé d'une classe.
		`// Standard`
		`$nbArticles = 6;`
		
		`// Correspond au nom du champ d'une table de BD`
		`$IdArticle = $_POST['IdArticle'];`
		
		`// Attribut public d'une classe`
		`public $article;`
		
		`// Attribut privé ou protégé d'une classe`
		`private $_article;`

Constantes	Inscrites en majuscule.
	
	Un tiret bas simple (_) sépare les mots d'une même constante.
		`define('SITE_PATH',realpath(dirname(__FILE__)));`
		
		`include SITE_PATH . '/includes/Db.php';`

Fonctions	Inscrites en minuscule. Une majuscule est ajoutée au début de chaque mot suivant composant le nom de la fonction.
	
	Un tiret bas simple (_) est employé devant une fonction servant de méthode privée ou protégée d'une classe.
		`// Standard`
		`function mailSend(){ … }`
		
		`// Méthode public d'une classe`
		`public function mailSend(){ … }`
		
		`// Méthode privée ou protégée d'une classe`
		`private function _mailSend(){ … }`

Classes	Une majuscule en début de chaque mot composant le nom de la classe.	`class Articles()`
		`{`
		`}`

MySQL	Le langage MySQL inscrit dans le PHP est en majuscule.	`$sql = 'SELECT * FROM menus;`

Tables (BD)	Le nom d'une table est en minuscule et au pluriel. Une table associative s'inscrit au singulier séparé par un tiret bas simple (_).
	
	Un tiret bas simple (_) sépare les mots d'un même nom de table.
		`// table standard`
		`$sql = 'SELECT * FROM menus;`
		
		`// table associative`
		`$sql = 'SELECT * FROM menus_motcle';`

Champ (BD)	Une majuscule est ajoutée au début de chaque mot composant le nom du champ. Un rappel du nom de la table au singulier est introduit.	`$sql = 'SELECT IdMenu FROM menus';`

Fichiers, répertoires et espaces de nom	Le nom de fichiers et répertoires est en minuscule.
	
	La première lettre du nom de fichiers et de répertoires est tolérée pourvu qu'il s'agisse de cohérence avec le nom d'une classe du fichier.
		`namespace application\menus;`
		
		`include SITE_PATH . '/includes/Db.class.php';`

Gestion des accès

Connexion

Le processus de connexion d'un utilisateur est géré par la classe « includes/Login.class.php ». Cette même classe permet l'obtention d'un nouveau mot de passe.

Une fois l'utilisateur connecté la session dispose des données suivantes :

```
$_SESSION['adminOK']          Boolean | True lorsque connecté
$_SESSION['adminLogin']       String  | Pseudo (e-mail) de l'utilisateur
$_SESSION['adminId']          Integer | Id de l'utilisateur
$_SESSION['adminRight']       Integer | Id du groupe de l'utilisateur
$_SESSION['adminLastname']    String  | Nom de l'utilisateur
$_SESSION['adminFirstname']   String  | Prénom de l'utilisateur
$_SESSION['adminOffice']      Integer | Id du bureau d'appartenance de l'utilisateur
```

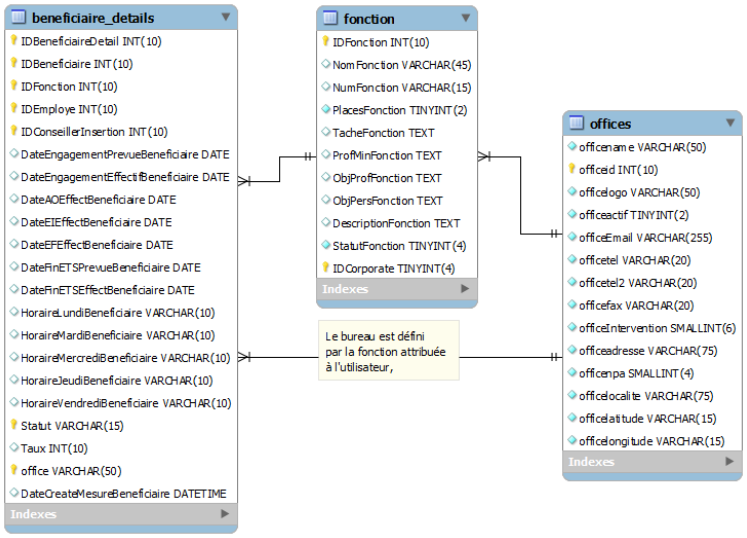
Droits

Les droits d'accès d'un utilisateur sont définis selon les paramètres suivants :

- **Le groupe d'utilisateur auquel il appartient :** Chaque groupe peut accéder différemment aux modules et pages du système. Ceci conditionne la composition des rubriques apparaissant dans le menu principal en plus des éléments accessibles dans les différentes interfaces.

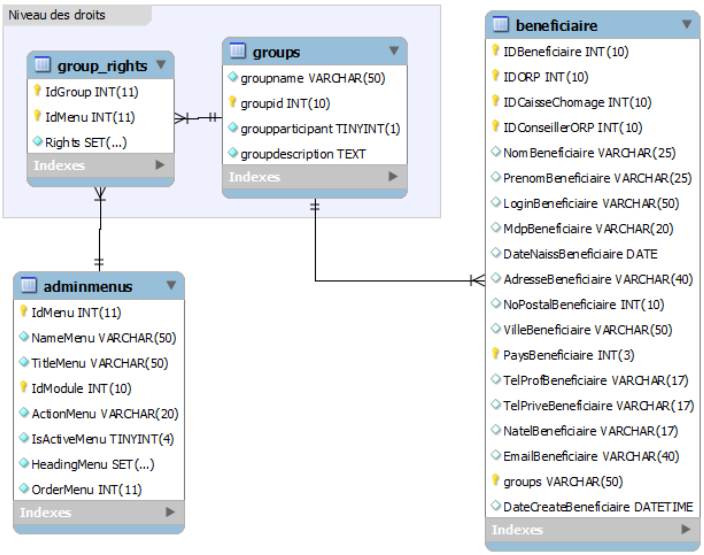
- **Le bureau auquel il appartient** : Les modules ne sont pas les mêmes pour chaque bureau. Ceci conditionne la composition des rubriques apparaissant dans le menu principal.

Lors de la connexion au système de l'utilisateur, ce dernier accède au bureau qui correspond à la plus récente période d'activité comme indiqué dans la table « beneficiaire_details ». Le champ « office » indique l'identifiant correspondant au bureau auquel l'utilisateur appartient pour cette période. Celui-ci est défini lorsque la fonction a été attribuée à l'utilisateur car chaque fonction est elle-même associée à un bureau.



Groupe d'utilisateurs

Les groupes d'utilisateurs indiquent précisément ce qui est accessible ou possible de faire pour un utilisateur. Le groupe auquel un utilisateur fait parti est précisé dans la table **« beneficiaire »**, les droits de ce groupe sont eux spécifiés dans la table **« group_rights »** qui est liés à la table gérant les données du menu principal **« admimenus »**.



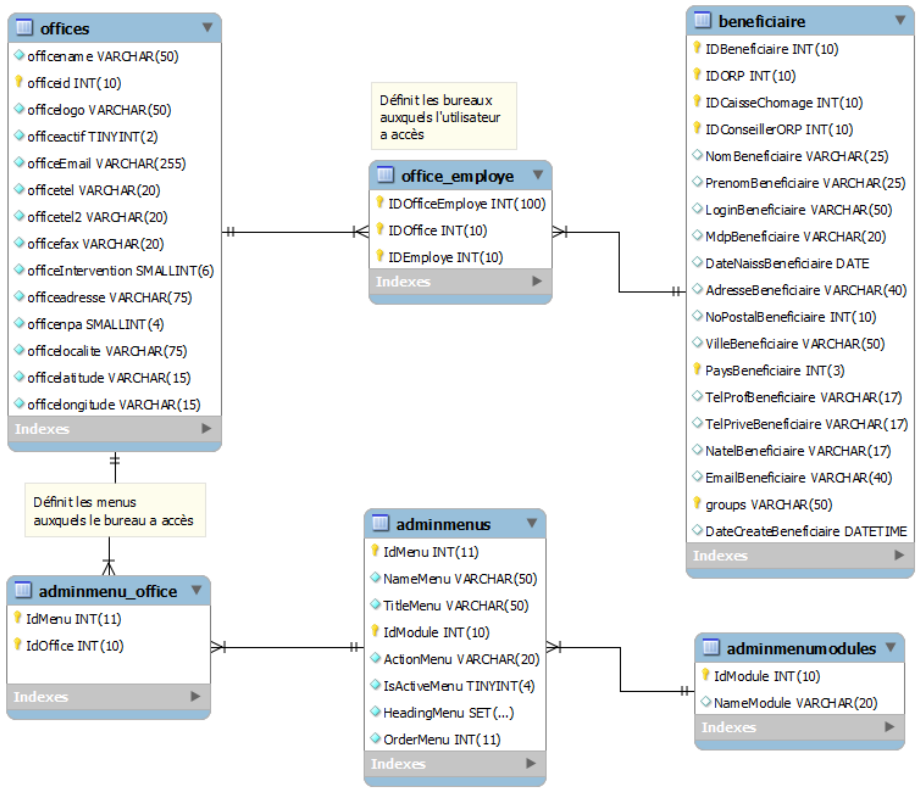
Les niveaux droits des groupes d'utilisateurs sont établis dans la table **« group_rights »** comme présentés dans le tableau qui suit.

Niveau de droits	Lettre utilisée dans la table « group_rights »
Lecture	r
Ajout	w
Modification	m
Suppression	d
Validation	v

La gestion des droits des groupes d'utilisateurs est établie dans la classe **« includes/Adm.class.php »**. Toutefois des fonctions spécifiques suivantes permettent de vérifier le niveau de droit d'accès d'un groupe d'utilisateur en fonction d'un module et accessoirement d'une action. Toutes renvoient **« true »** ou **« false »** en fonction du verdict de la vérification.

```
Lecture      autorise_read( 'module', 'action' );   Booléen | True lorsque droit donnée
Ajout        autorise_add(  'module', 'action' );   Booléen | True lorsque droit donnée
Modification autorise_mod(  'module', 'action' );   Booléen | True lorsque droit donnée
Suppression  autorise_del(  'module', 'action' );   Booléen | True lorsque droit donnée
Validation   autorise_valid('module', 'action' );   Booléen | True lorsque droit donnée
```

Bureau
Les bureaux disposent de droits d'accès à certains modules. L'utilisateur quant à lui peut accéder à un ou plusieurs bureaux.



Utilitaires

Créateur de modules

La mise en place d'un nouveau module est facilitée par un outil qui génère automatiquement à partir de tables de la base de données les fichiers, les interfaces et les fonctionnalités. Les interfaces utilisent les outils communs déjà fonctionnels. Tandis que les fonctionnalités correspondent à l'ajout, modification et suppression de données qui utilisent également des ressources déjà fonctionnelles telles que des appels en AJAX.

L'intérêt d'un tel outil est multiple :

- Gain de temps pour le développeur et obtention rapide d'un résultat.
- Faciliter la familiarisation avec des composantes spécifiques propres à l'outil.
- Eviter, ou du moins limiter, le codage à double de certaines fonctionnalités redondantes
- Faciliter la maintenance en imposant partiellement une méthode de codage.

Sur la base du code généré, le développeur aura à mettre en place les interfaces et fonctionnalités complémentaires.

Audit

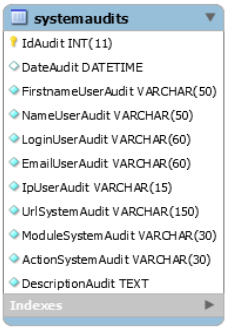
Les manipulations des utilisateurs du système peuvent engendrer de fausses interprétations de ce que le système a réellement effectué comme opération.

Pour cette raison, un système d'audition de certaines manipulations des utilisateurs avait été mis en place dans la première version de l'extranet et aurait intérêt à s'y retrouver dans la nouvelle version de l'outil.

Les manipulations sont recensées dans la table **« systemaudits »** et contiennent les informations suivantes :

- Date et heure de la manipulation
- Nom, prénom, nom d'utilisateur et adresse e-mail de l'utilisateur

- Adresse IP du poste de l'utilisateur
- Adresse de la page sur laquelle le problème est survenu
- Module et action menés par le système
- Description de l'action tentée



Compte tenu de la densité et la quantité des informations recensées ainsi que de la faible pérennité et utilité dans le temps qu'elles ont, seules les 5'000 dernières données sont conservées.

Modules et framework PHP

Plusieurs outils ont été mis en place pour faciliter la création et la maintenance des modules. Voici une description des ressources disponibles.

Application

L'application dispose des ressources de traitement des données. Les modules dispose tous d'un « Controller », de « Models » et d'une « Interface ». Accessoirement certains dispose de « Builders ».

« Controller » (Class)
Le « Controller » a comme rôle de :

- Transmettre des demande de traitement des données aux « Models » (il peut s'agir de modèles provenant d'autres modules).
- Récupérer les informations utiles à l'affichage par l' « Interface ».
- Indiquer la vue « View » à afficher.

« Models » (Class)
Les « Models » se divisent en fonction des ensembles de données pouvant stratégiquement être regroupées par la relation qui existe entre leurs données. Ils ont comme rôle de :

- Établir les requêtes vers la base de données.
- Traiter et formater les données afin qu'elles correspondent au format d'affichage prévu dans les vues.

« Interface » (Class)
L' « Interface » est automatiquement disponible dans le « Controller » du module auquel il appartient (par l'intermédiaire de l'attribut ($this->_interface). Il a le rôle de :

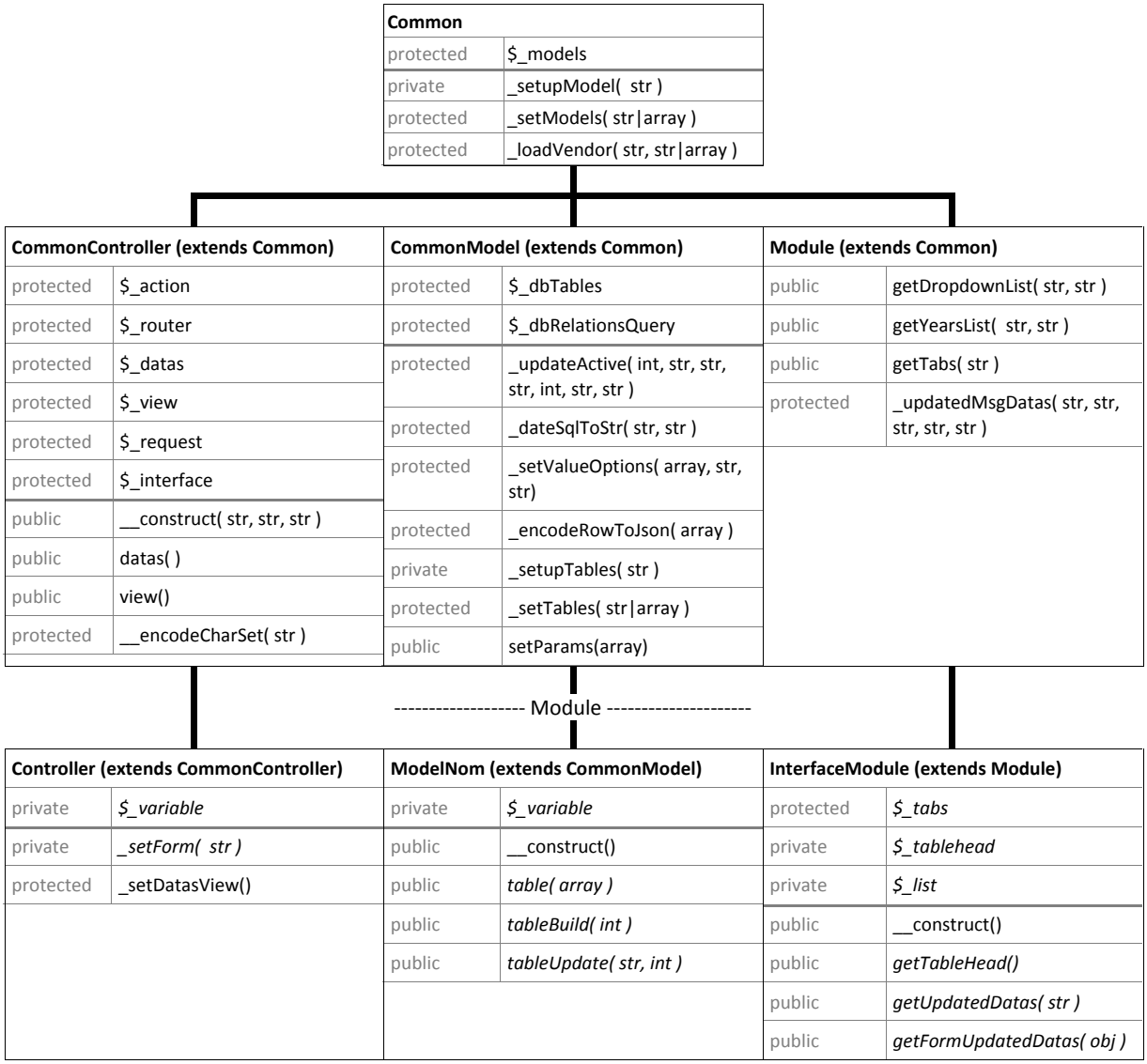
- Disposer des données fixes utiles à l'affichage (onglets d'interfaces, entêtes de tables, listes déroulantes)
- Composer les messages de réponse prévus pour les interfaces suite à un traitement.
- Traiter et formater des données réservées au module prévues pour un affichage dans les vues (en complément au « Model »).

« Builders » (Array)
Un « Builder » est créé pour chaque « Model » existant dans le module. Il dispose des informations décrivant chaque table de la base de données utiles aux requêtes. Il a le rôle :

- Décrire chaque table et leur champs selon les indications prévues par l'ORM (voir la classe 'includes/tools/Orm.class.php').
- Indiquer les relations qui existent entre les clés étrangères des tables.
- Préciser les conditions de suppression des données en indiquant les relations qui ne peuvent pas être supprimées.

Composants des modules

Les modules ont a disposition des ressources qui facilitent certaines opérations récurrentes. Ces opérations sont disponibles à travers une série d'héritage provenant des composants disponibles dans les classes de 'includes/components'.



ORM (Mapping Object Relationnel)

Un ORM permet de reproduire en orienté objet les relations et traitement des données d'une base de données. Cette technique a pour but d'offrir une meilleures visibilités de l'organisation des données et de faciliter les opérations courantes (requêtes, traitement et transmission des données à la base de données ou aux formulaires).

Mapping

Le mapping est une reproduction de la table de la base de données sous la forme d'un tableau. Il dispose des informations concernant le traitement réservé aux données de chaque champs. Ces informations sont utilisées lors du traitement. Elles indiquent notamment le type du champ, les valeurs par défaut à insérer, l'obligation de renseigner le champ, le format des données.

Les *mappings* sont déclarées dans le dossier 'builders' pour chaque module.

```php
'tablename' = [
  'Id'          => [ 'type' => 'INT',      'primary' => true,     'autoincrement' => true,
                                           'dependencies' => ['table'=>'IdField'] ],
  'Name'        => [ 'type' => 'STR',      'mandatory' => true ],
  'Infos'       => [ 'type' => 'TEXT',     'mandatory' => true ],
  'Date'        => [ 'type' => 'DATE',     'default' => 'NOW',    'dateformat' => 'DD.MM.YYYY' ],
  'Dateandtime' => [ 'type' => 'DATETIME', 'default' => 'NOW' ],
  'File'        => [ 'type' => 'STR',      'file' => true ],
  'Active'      => [ 'type' => 'INT',      'mandatory' => true,   'default' => 0 ], // Checkbox
]
```

Builder

Les informations indiquées dans le *mapping* sont liées au *builder*. Celui-ci s'occupe du traitement des données insérées dans les formulaires. Le *builder* permet d'utiliser le même formulaire pour l'ajout ou la modification de données. Il transmettra automatiquement au formulaire les données correspondantes. Dans le cas de l'ajout ces données seront vides ou contiendront les valeurs indiquées par défaut dans le *mapping*.

Autre intérêt du *builder* est qu'il transmettra les erreurs en fonction des indications indiquées quant aux champs obligatoires dans le *mapping*. Il permet ainsi de limiter les actions de vérification.

Class Orm()			
Type	Méthodes	Retour	Description
	__construct(str, [array] , [array])		Récupère le nom de la table à traiter et charge le *mapping* de la base de données (array) et les relations (array).
Sélection	**select([array])**	this	Initie une sélection. Peuvent être indiqué comme paramètre les champs à récupérer.
	join([array], [str])	this	Indique la jointure et les champs liés des deux tables.
	joins(array)	this	Utilise les jointures déclarées lors de l'initialisation de la classe en indiquant par le paramètre lesquelles effectuées.
	where([array])	this	Indique une condition équivalente.
	wherenot([array])	this	Indique une condition différente.
	whereor([array])	this	Indique une condition alternative.
	whereoror([array])	this	Indique une condition alternative pouvant disposer de plusieurs alternatives.
	whereorand([array])	this	Indique une condition alternative pouvant disposer de plusieurs conditions équivalentes.
	whereandor([array])	this	Indique une condition équivalente pouvant disposer de plusieurs alternatives.
	wheregreater([array])	this	Indique une condition de taille plus grande.
	wheregreaterandequal([array])	this	Indique une condition de taille plus grande et équivalente.
	wheregreaterorequal([array])	this	Indique une condition de taille plus grande ou équivalente.
	wherelower([array])	this	Indique une condition de taille plus petite.
	wherelowerandequal([array])	this	Indique une condition de taille plus petite et équivalente.
	wherelowerorequal([array])	this	Indique une condition de taille plus petite ou équivalente.
	wherecustom(str)	this	Indique une condition sur mesure (en SQL).
	wherelike([array])	this	Indique une condition de comparaison.
	group([array])	this	Groupe les résultats
	havinggreater([array])	this	Résultat doit avoir un nombre minimum d'entrées
	havinglower([array])	this	Résultat doit avoir un nombre maximum d'entrées

	havinggreaterorequal([array])	this	Résultat doit avoir un nombre minimum ou équivalent d'entrées
	havinglowerorequal([array])	this	Résultat doit avoir un nombre maximum ou équivalent d'entrées
	order([array])	this	Ordonner les résultats
	limit([array])	this	Indique une limite de résultats
	execute([boolean | *default=false*] **)**	array	Fin d'une requête indiquant de retourner tous les résultats sous la forme d'un tableau (d'objets). Indique également si des informations complémentaires sont à transmettre.
	first([boolean | *default=false*] **)**	object	Fin d'une requête indiquant de retourner un seul résultat sous la forme d'un objet. Indique également si des informations complémentaires sont à transmettre.
Requêtes	**getQuery()**	string	Renvoi la dernière requête effectuée
	numrows()	integer	Nombre de lignes dans le résultat
	count([array])	integer	Nombre de résultats correspondant au paramètre
	exist([array])	boolean	Si résultats existant selon le paramètre
	getNbResult([array])	integer	Nombre de résultats correspondant au paramètre
Préparation des données *Méthodes utilisées pour le traiter des données provenant de formulaires et prévues pour la base de données.*	**checkUniqueData(str, str, [array])**		Force l'ajout de données qui n'existe pas dans le *mapping* de la base de données mais utile au traitement au *builder* ou aux opérations d'insertion (*insert*) ou de mise à jour (*update*).
	prepareGlobalDatas([array], [array])	array | null	Récupère les données provenant des variables GET, POST ou FILE, les traite, initialise les erreurs et prépare ces données pour le builder, l'insertion ou la mise à jour. Dans le cas du FILE des informations complémentaires seront nécessaires pour définir les limites de tailles, poids et le format du fichier.
	setErrors([array])	(void)	Ajoute une erreur. Utile lors de traitements externes à L'*ORM* dont les erreurs doivent être répertoriées
	getErrors()	array	Permet d'afficher les erreurs identifiées par l'*ORM*
	issetErrors()	boolean	Indique qu'une erreur existe lors du traitement
Build *Transmet les données de la base de données prévues pour un affichage dans un formulaire.*	***build(array, [str])***	object	Renvoi les données correspondant au *mapping* destinées à un formulaire : - provenant d'une ligne de la table désignée. - vides dans le cas d'un ajout. - les données insérées par l'utilisateur dans le formulaire dans le cas d'erreurs.
	builds(array, [str])	array	Renvoi une liste de données correspondant au *mapping* destinées à un formulaire selon les mêmes conditions que pour la méthode *build()*.
Insertion (insert) et mise à jour (update)	**prepareDatas([array], [array])**	boolean	Permet d'ajouter une valeur destinée à un champ. Par exemple, dans le cas où un formulaire ne renseigne pas un élément du *mapping*.
	prepareDatasArray(array $datas, int $key)	boolean	Permet d'ajouter une valeur destinée aux champs du mapping à partir de données groupées dans un tableau (array). Cette méthode est à utiliser dans une boucle du type foreach($datas as $k => $data) contenant également l'appel des méthodes d'insertion (insert()) ou (update()). Ce tableau doit disposer des noms clés correspondant aux valeurs du mapping (champs de la table). Les données sont elles disposées de la même position dans le tableau (ie: ['Id'=>[0=>23, 1=>24], 'Name'=>[0=>'Name 1', 1=>'Name 2']]). Ce format de tableau de données est automatiquement défini dela sorte par la méthode prepareDatasGlobal().
	insert()	object | false	Effectue une insertion en fonction des données récupérées

			par la méthodes *prepareDatasGlobals()*, *prepareDatas()* et les informations provenant du *mapping*.
	update(array)	object | false	Effectue une mise à jour en fonction des données récupérées par la méthodes *prepareDatasGlobals()*, *prepareDatas()* et les informations provenant du *mapping*.
Suppression (delete)	**delete(array , [boolean** | *default=false*])	boolean	Effectue une suppression de données. Indique également s'il est question de faire une suppression récursive.
	isDependent(array)	boolean	Indique s'il existe une dépendance avec une donnée de la table et les tables en relation avec cette donnée.
	deleteRecursive(array)	boolean	Effectue une suppression de données dans les tables liées.

Modules et tables supprimés

De la première version à la seconde de l'extranet, plusieurs modules ne sont plus supportés car s'avèrent désuets ou inutilisés. Par conséquent, les tables de la base de données correspondantes ne sont plus supportées.

Modules	**Tables**
Accord d'objectif	accordobjectif
Absence	absence
Clients	clients
	clients_contacts
Documentation	documents
	documentcategories
	document_group
Exercices	exercices
	exercices_beneficiaires
	exercices_questions
	exercices_reponses
Logiciels	logiciels
News	news
	news_categories
	news_categories_relations
Planification	planifications
	planifications_activites
	planifications_commentaires
	planifications_horaires
Postes	postes
	postes_logiciels
	logiciels
Profils	profil
	profils_experiences
	profils_formations
	profils_liens_professionnels
	profils_liens_professionnels_relations
	profils_logiciels
	profils_logiciels_categories
	profils_logiciels_relations
	profils_metiers
	profils_metiers_categories
	profils_metiers_relations
	profilweb

Modules	**Tables**
Projets	beneficiaireprojet
	projet
	projets_etats
	projets_type
	projet_type_categories
	projet_realise
	public_cible
	journalprojet
Sites	sites
	sites_categories
Taches	taches
Wiki	wiki_commentaire
	wiki_libelle
	wiki_rubrique
	wiki_theme
FAQ	faq_commentaire
	faq_question
	faq_rubrique
	faq_theme
Autres	countries_regions
	element_specifique
	prescripteurs
	menus
	menu_pages
	messagerie_group
	messagerie_group_membre
	messagerie_group_office
	pages_groups
	taches_projets
	coaching_feuilles
	coaching_feuille_inscrits
	acti

Modules

Module « Emprunt de matériel - *inventory* »

Ce module recense le matériel susceptible d'être emprunté par n'importe quel utilisateur du système pour une période définit. Ce module tient compte d'un classement par bureau. Le matériel disponible provient d'un bureau et ne peut être emprunté que par les utilisateurs ayant accès à ce bureau.

Le matériel pouvant être emprunté peut être de différent type. Il s'agit d'ouvrages mais également d'équipements.

L'emprunt peut se faire pour un ou plusieurs matériels à la fois. Un document établissant une liste du matériel emprunté par un utilisateur pour une période donnée devrait pouvoir être exporté du système.

Objectifs

- Permettre un suivi et un historique des emprunts du matériel
- Faciliter la consultation et l'emprunt du matériel à disposition
- Faciliter l'inventaire et la localisation du matériel

Améliorations à apporter

Dans cette nouvelle version du module, il serait nécessaire d'améliorer les éléments suivants :

- Faire une demande d'emprunt de plusieurs articles en une fois
- Disposer d'une liste des articles empruntés par un utilisateur pour une période définie

Interfaces

Les interfaces du module concernent la liste du matériel, les formulaires de gestion des données, celui des emprunts.

Liste du matériel

L'interface de la liste du matériel sert à la consultation et la recherche de matériel. La liste est classée par catégories d'article.

Un outil de filtre est proposé à l'utilisateur pour l'aider dans sa recherche. Le filtre propose les champs suivants :

Champs	Objet dans le formulaire	Commentaires
Titre/Code	Champ texte	
Catégorie	Liste déroulante (*Première option vide)*	
Type d'article	Liste déroulante (*Première option vide)*	
Etat	Liste déroulante (*Première option vide)*	
Emprunté	Case à cocher	

Il est possible depuis la liste :

- De consulter l'historique d'emprunt d'un matériel.
- D'avoir un signalement pour chaque article qu'il soit disponible, emprunté ou en retard.
- De constater qui est en possession d'un article.
- D'engager pour un utilisateur la demande d'emprunt pour un ou plusieurs articles.
- D'accéder au formulaire pour modifier les informations concernant un article le supprimer.

Formulaire d'emprunt

Ce formulaire dispose des informations permettant l'emprunt d'un article.

Champs	Objet dans le formulaire	Commentaires
Emprunteur	Liste déroulante	*Obligatoire* *[Utilisateur en période d'activité (lors de la modification : récupérer l'utilisateur bien qu'il ne soit plus en activité)]*
	Champ (type : caché)	*Invisible pour un utilisateur du type participant*
Liste des articles	Cases à cocher	*Obligatoire (au moins un article)*
Début de l'emprunt	Champ (type : date)	*Obligatoire*
Fin de l'emprunt	Champ (type : date)	*Obligatoire* *Futur à la date de début de l'emprunt*
Statut de la demande	Boutons radio	*Obligatoire* *[1 =>demande, 2=>emprunt, 3=>rendu]* *[Suppression possible d'une demande d'emprunt pour les ayants droits]*
	Champ (type : caché)	*Invisible à l'initiation d'une demande*
Identifiant de l'article	Champ (type : caché)	*Invisible*
Date de la demande d'emprunt	Champ (type : caché)	*Invisible*

Table(s) (BD) correspondante(s) : « librairie_emprunts »

Formulaire de gestion des informations d'un article

Ce formulaire permet d'informer des informations concernant un article.

Champs	Objet dans le formulaire	Commentaires
Nom	Liste déroulante	*Obligatoire*
Type	Liste déroulante	*[Associée au type de matériel (voir formulaire correspondant)]*
Catégorie	Liste déroulante	*Obligatoire* *[Associée aux catégories de matériel (voir formulaire correspondant)]*
Editeur	Liste déroulante (*Première option vide)*	*[Associée aux éditeurs (voir formulaire correspondant)]*
Prénom de l'auteur	Champ (type : texte)	
Nom de l'auteur	Champ (type : texte)	
Code (ex. ISBN)	Champ (type : texte)	
Numéro d'inventaire	Champ (type : texte)	
Etat	Liste déroulante	*[1 =>en fonction, 2=>désuet, 3=>hors d'usage, 4=>perdu]*
Responsable	Liste déroulante	*[Associée à la table « beneficiaire » des utilisateurs du bureau courant mais ne faisant pas parti d'un groupe associé aux participants de la table « groups ».]*

Table(s) (BD) correspondante(s) : « librairie_articles »

Formulaire de gestion des catégories

Ce formulaire permet de donner les informations concernant une catégorie d'article.

Champs	Objet dans le formulaire	Commentaires
Nom de la catégorie	Champ (type : texte)	*Obligatoire*
Identifiant du bureau	Champ (type : caché)	*Invisible*

Table(s) (BD) correspondante(s) : « librairie_categories »

Formulaire de gestion des informations d'un éditeur
Ce formulaire permet de donner les informations concernant un éditeur.

Champs	Objet dans le formulaire	Commentaires
Nom de l'éditeur	Champ (type : texte)	*Obligatoire*

Table(s) (BD) correspondante(s) : « librairie_editions »

Formulaire de gestion des informations du type de matériel
Ce formulaire permet de donner les informations concernant le type de matériel.

Champs	Objet dans le formulaire	Commentaires
Type de matériel	Champ (type : texte)	*Obligatoire*

Table(s) (BD) correspondante(s) : « librairie_types »

Fonctionnalités

1. Recenser les informations concernant un article
   Le formulaire de gestion des articles associe les données des tables **« librairie_categories »**, **« librairie_editions »** et **« librairie_types »**. L'emprunteur provient de la table **« beneficiaire »**.

   Le bureau associé à un article l'est par la catégorie définit pour cet article.

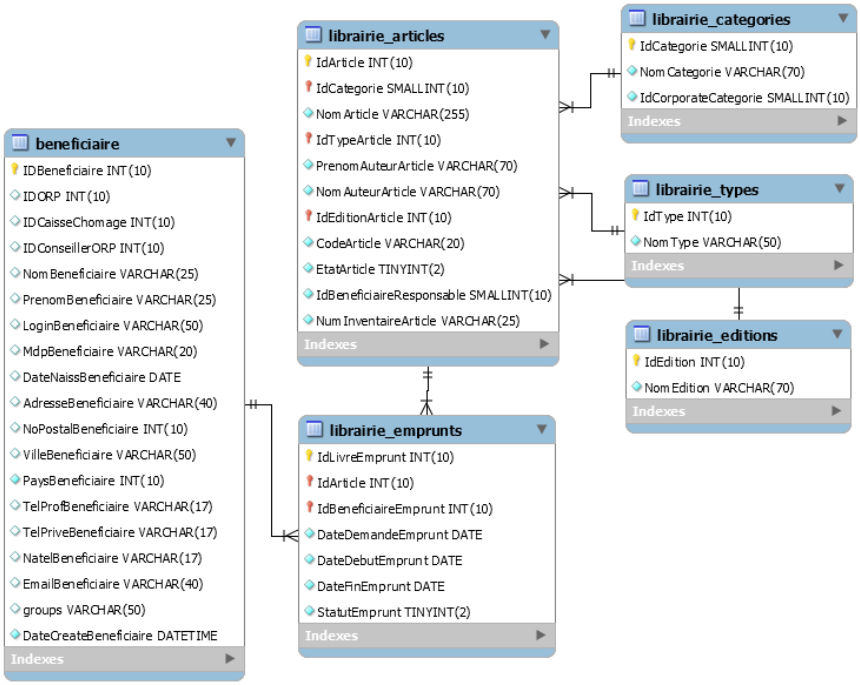
2. Recherche d'un article
   L'outil de filtrage des articles limite les résultats affiché dans la liste. Plus il y a de critères établis dans le filtre plus le résultat est limité.

3. Processus d'emprunt
   La demande d'emprunt se fait par le formulaire concerné pour un utilisateur. Dans le cas où l'utilisateur ne fait pas parti d'un groupe du type participant, il peut effectuer l'emprunt pour quelqu'un d'autre. Autrement l'emprunt est pour l'utilisateur courant.

   Lorsque la demande est initiée, un e-mail est transmis à l'adresse du bureau indiquant les informations concernant l'emprunt (nom de l'emprunteur, articles demandés, période (dates) prévue de l'emprunt).

   Au retour du matériel, un utilisateur ne faisant pas parti d'un groupe du type participant, change le statut de l'emprunt en l'indiquant comme « rendu ».

Table(s) de la base de données



Module « Ateliers - *workshops* »

Ce module permet la gestion des ateliers et formations pour chaque bureau. Cela concerne la création de formations, la planification de celles-ci et l'évaluation par les participants des apprentissages.

Objectifs

- Recenser les formations existantes
- Planification des ateliers par date et en fonction de participants intéressés.
- Permettre aux participants d'être au centre de la planification et l'évaluation de son apprentissage par des outils simples à utiliser.
- Permettre d'extraire aisément les feuilles de présence.

Améliorations à apporter

Dans cette nouvelle version du module, il serait nécessaire d'améliorer l'élément suivant :

- Permettre de planifier plus d'une date pour un futur atelier ou formation.

Interfaces

Les accès aux interfaces de ce module sont définit en fonction du groupe d'utilisateur et du bureau.

Liste des ateliers et formations

L'interface de la liste des ateliers sert à la consultation et planification des ateliers. La liste est classée par domaine. Il est possible de consulter par des onglets les ateliers actifs, les inactifs ou tous les ateliers.

Il est possible depuis la liste :

- De consulter l'historique de chaque ateliers (dates auxquelles il a eu lieu ainsi que la liste des inscrits
- D'avoir un signalement indiquant le nombre de participants en demande, inscrit ou qui ont suivis l'atelier.
- De planifier de nouvelles dates à l'atelier.
- D'effectuer l'inscription et le suivi des présences pour chaque participant pour chaque article.
- D'accéder au formulaire pour modifier les informations concernant un atelier ainsi que pourvoir le le supprimer.

Le formulaire des ateliers.

Champs	Objet dans le formulaire	Commentaires
Nom	Champ (type:texte)	*Obligatoire*
Domaine	Liste déroulante	*Associé aux domaines*
Formateur	*Liste déroulante*	*Associé aux formateurs*
Lieu	Champ (type:texte)	
Nombre de période	Liste déroulante	*De 1 à 12 périodes. (3 périodes représentes une demi-journée complète). Indiquer le nb de période et le nb de jour dans chaque énoncé de la liste.*
Description	Zone de texte	
Prérequis	Zone de texte	
Remarques	Zone de texte	
Statut	Champ caché (type : hidden)	*'archive' ou 'actif' [par défaut : 'actif']*
Type	Type de formation	*0=Théorique ; 1=Pratique ; 2=Séance ; 3=Cours*
Bureau	Champ caché (type : hidden)	*Invisible*

Table(s) (BD) correspondante(s) : « coaching »

Le formulaire d'inscription permet planifier les ateliers et d'indiquer l'état de l'inscription pour chaque participant.

Champs	Objet dans le formulaire	Commentaires
Date	Champ (type:date)	*Obligatoire*
Heure de début	Champ (type:time)	*Obligatoire*
Heure de fin	*Champ (type:time)*	*Obligatoire*
Statut	Bouton radio	*'demande', 'inscrit', 'suivi', 'absent'*
Motif absence	Champ (type:texte)	*En cas d'absence*
Absence justifiée	Case à cocher	*0=pas specifie; 1=autorise; 2=pas autorise;*
Message	Champ (type:texte)	
Responsable de l'inscription	Champ caché (type : hidden)	*Identifiant de la personne [table:beneficiaire] ayant initiée la planification de l'atelier*
Atelier	Champ caché (type : hidden)	*Identifiant de l'atelier [table:coaching]*
Participant	Champ caché (type : hidden)	*Identifiant du participant [table:beneficiaire]*

Table(s) (BD) correspondante(s) : « benenficiairecoaching »

L'inscription a un atelier peut être fait par un participant qui signale son intérêt pour un atelier.

Liste et formulaires des domaines

Les ateliers sont regroupés par domaines. Ces domaines permettent de plus aisément repérer les ateliers lors des inscriptions notamment. Ces domaines sont associés à un bureau et permettent de rendre accessible les ateliers au bureau concerné.

Une liste recense les domaines et sous-domaines. Deux formulaires permettent de mettre à jour leur contenu. Un premier permet de créer les domaines.

Champs	Objet dans le formulaire	Commentaires
Nom du domaine	Champ (type : texte)	*Obligatoire*
Domaine associé	Liste déroulante	

Table(s) (BD) correspondante(s) : « domaine »

Le second formulaire dispose des informations des sous-catégories.

Champs	Objet dans le formulaire	Commentaires
Nom du domaine	Champ (type : texte)	*Obligatoire*
Description	Liste déroulante	
Public cible	Zone de texte	
Type de projets concernés	Zone de texte	
Bureau	Champ (type : hidden)	*Invisible*

Table(s) (BD) correspondante(s) : « domaine_ateliers »

Liste et formulaire des formateurs

Les formateurs sont associés aux ateliers. Chaque atelier doit disposer d'un formateur qui lui est associé. La liste des formateurs permet de recenser leurs coordonnées et informations personnelles.

Champs	Objet dans le formulaire	Commentaires
Nom	Champ (type : texte)	*Obligatoire*
Prénom	Champ (type : texte)	*Obligatoire*
Téléphone	Champ (type : texte)	
Adresse E-mail	Champ (type : texte)	
Adresse	Champ (type : texte)	*Invisible*
Npa	Champ (type : texte)	
Localité	Champ (type : texte)	
Matières enseignées	Champ (type : texte)	
Statut	Champ caché (type : hidden)	*'archive' ou 'actif' [par défaut : 'actif']*
Bureau	Champ caché (type : hidden)	*Invisible*

Table(s) (BD) correspondante(s) : « formateur »

Questions et évaluation de ateliers

L'interface de la liste des domaines sert à organiser les ateliers.

Champs	Objet dans le formulaire	Commentaires
Question	Champ (type:texte)	*Obligatoire*
Destinataire	Bouton radio	*1=participant ; 2=formateur ;*
Statut	*Liste déroulante*	*0=archive ; 1=actif ;*
Bureau	Champ caché (type : hidden)	*Invisible*

Table(s) (BD) correspondante(s) : « coaching_evaluation_questions »

L'évaluation d'un atelier recense l'identifiant du participant, de la question et de l'atelier en plus de la date de l'atelier et la note attribuée (entre 1 à 5). Le formulaire d'évaluation affiche la liste des questions prévues pour l'évaluation et permet de noter chacune d'elle.

Champs	Objet dans le formulaire	Commentaires
Note (pour chaque question)	Bouton radio	*Obligatoire (1 à 5)*

Table(s) (BD) correspondante(s) : « coaching_evaluations »

Fonctionnalités

1. Recenser les informations concernant les ateliers, les domaines, les formateurs et les questions
   Le formulaire de gestion des ateliers associe les données des tables **« coaching »**, **« domaine », « formateur » et « coaching_evaluation_questions »**.

2. Suivi des inscriptions à un atelier
   La gestion des inscriptions aux ateliers se fait par la table **« beneficiairecoaching »**. L'inscription d'un participant à un atelier donne lieu a une inscription dans la base de donnée. En cas de modification de l'inscription, la donnée précédente en lien avec cet atelier est remplacée.

   Lors de l'inscription (statut « inscrit ») de participant(s) a un atelier, l'administrateur devrait avoir la possibilité d'envoyer une convocation par e-mail aux personnes inscrites.

   Lorsqu'un participant est absent d'un atelier, cette information peut être recenser lorsque le statut du participant pour cet atelier est définit comme tel (statut « absent »). Apparaît alors un champ qui permet d'indiquer le motif en plus d'une case à cocher dans le cas où ce motif justifie cette absence.

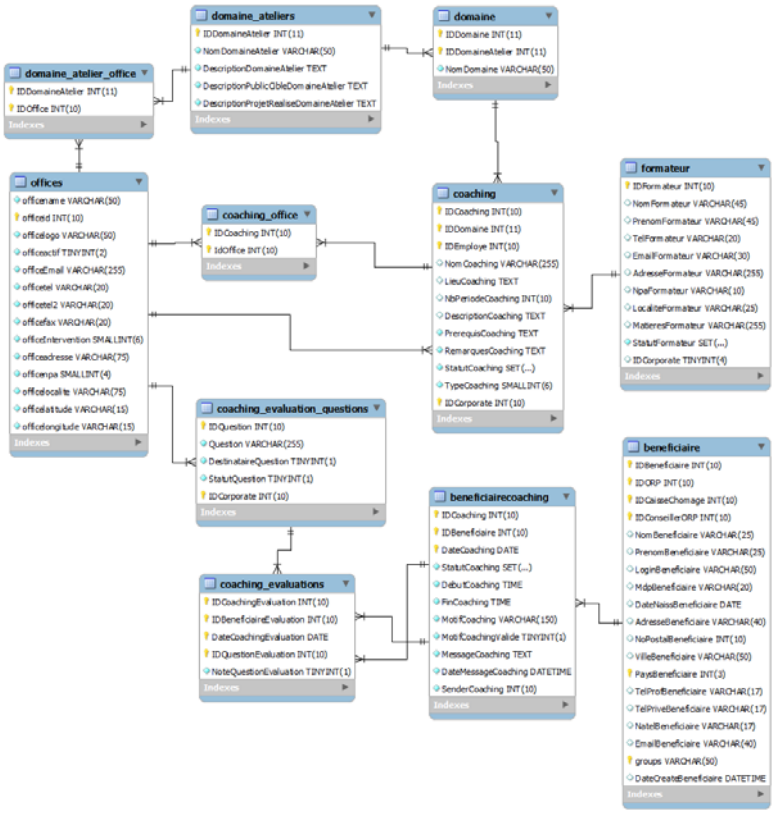
   Lorsqu'un participant a suivi un atelier, le statut de son inscription est définit comme tel (statut « suivi »). Au moment de la mise à jour de cette indication, un message est automatiquement transmis au participant l'invitant à remplir l'évaluation de l'atelier.

3. Génération de la feuille de présence
   Il est possible de générer une feuille de présence au format PDF pour chaque atelier planifié. Cette feuille de présence dispose des noms de chaque participant prévu en plus d'une colonne commentaire et une autre pour la signature du participant.

4. Évaluation d'un atelier
   Après avoir suivi un atelier le participant reçoit un message par e-mail l'invitant à évaluer l'atelier qu'il a suivi. Le formulaire qui lui est proposé expose les questions prévues pour cette évaluation provenant de la table **« coaching_evaluation_question »** invitant à noter chacune d'elle sur une échelle de 1 à 5. Chaque question évaluée est recensée dans la table « coaching_questions ».

   Le formulaire dispose également d'une zone de texte qui permet au participant de formuler une remarque concernant l'atelier. Cette information n'est toutefois par archivée dans la base de donnée mais figurera dans le message automatiquement transmis par e-mail à l'adresse du bureau tel qu'indiqué dans le champ « officeEmail » de la table « offices ».

Table(s) de la base de données



Module « Demandes d'intervention - *request* »

Présentation du module

Ce module permet de gérer les demandes d'interventions formulées auprès d'un bureau. Il permet a un utilisateur de faire une demande en remplissant un formulaire de demande. Cette demande peut être suivie et traitée depuis la zone d'administration. L'instigateur de la demande peut évaluer la prestation une fois le problème signalé a été résolu.

Une des particularités de ce module est que le questionnaire de la demande d'intervention peuvent être conçues sur mesure en fonction du type d'intervention. Chaque bureau peut disposer de son propre formulaire.

Objectifs

- Faciliter les demandes d'intervention des utilisateurs par une interface facile à employer.
- Permettre à l'utilisateur de suivre le déroulement de ses demandes.
- Établir une processus simple de gestion des demandes.
- Permettre aux utilisateurs d'évaluer les prestations
- Établir des statistiques sur la base des évaluations des utilisateurs.

Améliorations à apporter
Au niveau fonctionnel, peu de changements sont prévus. Il est question cependant d'améliorer et simplifier l'utilisation en révisant les interfaces de l'outil :

- Améliorer l'interface de la demande d'intervention qui permet à l'utilisateur de constater les étapes au fil de l'établissement de la demande.
- Améliorer l'interface de gestion et de suivi des demandes autant pour les administrateurs que pour les instigateurs des demandes.

Interfaces
Les interfaces de ce module permettent s'établissent en fonction du public concernés. Il y a, d'une part, les administrateurs qui ont à établir les questionnaires des demandes d'intervention, suivre les demandes et les évaluations. D'autre part, les instigateurs de demandes qui doivent pourvoir formuler des demandes aisément, suivre l'évolution de la demande et évaluer la prestation.

Liste et formulation des questions d'évaluation (pour administrateurs)
L'administrateur formule les questions que le demandeur doit répondre pour formuler une demande.
Sont prévus quatre (4) niveaux de questions :

- Deux (2) niveaux vont permettre d'établir les questions des étapes de l'établissement d'une demande.
- Un autre niveau génère les questions d'évaluation.
- Le dernier niveau sont les questions d'information auxquelles l'administrateur répond pendant le traitement de la demande.

L'ordre d'apparition des questions peut être établit et modifiée pour chaque niveau.

***Niveau 1** : Première étape - Questions préalables.*
Les questions créées pour cette étape sont celles considérées comme préalables à la demande. Il s'agit de vérifications que doit faire l'utilisateur avant de formuler sa demande.

Champs	Objet dans le formulaire	Commentaires
Étiquette de la question	Champ (type:text)	*Obligatoire*
Type de question	Champ (type:hidden)	*demande une confirmation à l'utilisateur [valeur unique : 0]*
Visible	Champ (type:hidden)	*0:invisible ; 1:visible ; [par défaut : 1]*
Bureau	Champ (type:hidden)	*Identifiant du bureau associé à la question*
Ordre	Champ (type:hidden)	*Dernier du lot de questions du niveau*
Nom du champ	Case à cocher	*[par défaut : activée]*

Table(s) (BD) correspondante(s) : « interventions_questions »

***Niveau 2** : Seconde étape - Questions de la demande d'intervention.*
Il s'agit des questions concernant la demande d'intervention. L'administrateur définit pour chaque question, le type de question en précisant s'il s'agit d'un champ texte, d'une zone de texte, d'un bouton radio, d'une case à cocher, d'une liste déroulante ou d'une remarque adressée à l'utilisateur.

Lorsqu'il s'agit d'un bouton radio, d'une case à cocher ou d'une liste déroulante, des libellés peuvent être ajoutés afin de proposer des options à l'utilisateur.

Champs	Objet dans le formulaire	Commentaires
Étiquette de la question	Champ (type:text)	*Obligatoire*
Type de question	Bouton radio	*demande une confirmation à l'utilisateur [par défaut : 1]* *(1:champ texte ; 2:Zone de texte ; 3 :Bouton radio ; 4:Case à cocher ; 5:Liste déroulante ; 7:Remarques;)*

Choix de réponses	Champs (type:text) - Multiples	*Dans le cas de boutons radio, de cases à cocher ou d'une liste déroulante.*
Visible	Champ (type:hidden)	*0:invisible ; 1:visible ; [par défaut : 1]*
Bureau	Champ (type:hidden)	*Identifiant du bureau associé à la question*
Ordre	Champ (type:hidden)	*Dernier du lot de questions du niveau*
Nom du champ	Case à cocher	*[par défaut : activée]*

Table(s) (BD) correspondante(s) : « interventions_questions, interventions_choix »

***Niveau 3** : Évaluation de l'intervention.*
Les questions générées à ce niveau serviront à l'évaluation de la prestation. Les types de questions sont les mêmes que ceux proposés au niveau précédent. S'ajoute toutefois la possibilité de formuler une question permettant de noter sur une échelle de 1 à 3 le niveau de la prestation.

Champs	Objet dans le formulaire	Commentaires
Étiquette de la question	Champ (type:text)	*Obligatoire*
Type de question	Bouton radio	*demande une confirmation à l'utilisateur [par défaut : 1]* *(1:champ texte ; 2:Zone de texte ; 3 :Bouton radio ; 4:Case à cocher ; 5:Liste déroulante ; 6:Évaluation (note); 7:Remarques;)*
Choix de réponses	Champs (type:text) - Multiples	*Dans le cas de boutons radio, de cases à cocher ou d'une liste déroulante.*
Visible	Champ (type:hidden)	*0:invisible ; 1:visible ; [par défaut : 1]*
Bureau	Champ (type:hidden)	*Identifiant du bureau associé à la question*
Ordre	Champ (type:hidden)	*Dernier du lot de questions du niveau*
Nom du champ	Case à cocher	*[par défaut : activée]*

Table(s) (BD) correspondante(s) : « interventions_questions, interventions_choix »

***Niveau 4** : Suivi de l'intervention.*
Les questions de ce dernier niveau servent à l'administrateur dans le processus de suivi de la prestation. Les types de questions sont les mêmes que ceux proposés au niveau précédent.

Champs	Objet dans le formulaire	Commentaires
Étiquette de la question	Champ (type:text)	*Obligatoire*
Type de question	Bouton radio	*demande une confirmation à l'utilisateur [par défaut : 1]* *(1:champ texte ; 2:Zone de texte ; 3 :Bouton radio ; 4:Case à cocher ; 5:Liste déroulante ; 7:Remarques;)*
Choix de réponses	Champs (type:text) - Multiples	*Dans le cas de boutons radio, de cases à cocher ou d'une liste déroulante.*
Visible	Champ (type:hidden)	*0:invisible ; 1:visible ; [par défaut : 1]*
Bureau	Champ (type:hidden)	*Identifiant du bureau associé à la question*
Ordre	Champ (type:hidden)	*Dernier du lot de questions du niveau*
Nom du champ	Case à cocher	*[par défaut : activée]*

Table(s) (BD) correspondante(s) : « interventions_questions, interventions_choix »

<u>La demande d'intervention</u>
Un utilisateur peut formuler une demande d'intervention en franchissant trois (3) étapes :

Étape 1 : Définir le bureau

Le module permet à différents bureau d'établir un formulaire de traitement de demandes d'intervention. La première étape consiste à définir le bureau à qui sera adresser la demande.

Champs	Objet dans le formulaire	Commentaires
A qui s'adresse la demande d'intervention	Bouton radio	

Table(s) (BD) correspondante(s) : « interventions_questions »

Étape 2 : Valider les questions préalables
Des questions préalables ont été établies par l'administrateur à l'aide de l'outil de formulation des questions. Les questions prévues pour cette étapes exigent une validation par l'utilisateur en cochant la cases réservées à chaque énoncées. L'étape pourra passer à l'étape suivante, une fois seulement que toutes les cases sont cochées

Champs	Objet dans le formulaire	Commentaires
Questions de l'avis préalable.	*Divers objets possibles : Cases à cocher.*	*Établit dans l'outil de formulation des questionnaires.*

Table(s) (BD) correspondante(s) : « interventions_questions, interventions_questions »

Étape 3 : Formulaire de demandes d'intervention
Le questionnaire est établi par l'administrateur à l'aide de l'outil de formulation des questions. Les questions se présentent en fonction de ce qui a été définit dans l'outil. Il peut s'agir de boutons radio, de cases à cocher, de champs, de zones de texte, de listes de déroulante ou de remarques.

Champs	Objet dans le formulaire	Commentaires
Questions de la demande d'intervention	*Divers objets possibles : Champ (type:text), Bouton radio, Case à cocher, Liste déroulante, Zone de texte ou remarques.*	*Établit dans l'outil de formulation des questionnaires.*

Table(s) (BD) correspondante(s) : « interventions, interventions_reponses, interventions_questions, interventions_choix »

Étape 4 : Validation de l'envoi de la demande
La dernière étape valide la demande en indiquant que celle-ci a bien été transmise. Un message contenant toutes les informations insérées dans le formulaire est transmis au bureau concerné.

<u>Le processus et le suivi des interventions</u>
Le suivi des interventions s'établit selon un processus en fonction de la résolution de la demande. Les demandes passent de « demande », à « en cours » à enfin « terminé ». Ce cheminement inclus des avis par e-mail pour l'avis d'une nouvelle demande comme pour l'évaluation de la prestation.

Les demandes sont listés et identifiées en fonction de leur état de résolution. Les administrateurs ont la possibilité de modifier une demande qu'à formuler un demandeur. Ce formulaire s'établit en fonction des questions élaborées pour la demande d'intervention.

A ces informations s'ajoutent l'état d'avancement de la demande ainsi que les questions relatives au suivi. Elles aussi créées sur mesure dans l'outil de formulation des questionnaires.

Champs	Objet dans le formulaire	Commentaires
Questions de la demande	*Divers objets possibles :*	*Établit dans l'outil de formulation des questionnaires.*

d'intervention	*Champ (type:text), Bouton radio, Case à cocher, Liste déroulante, Zone de texte ou remarques.*	
Questions du suivi	*Divers objets possibles : Champ (type:text), Bouton radio, Case à cocher, Liste déroulante, Zone de texte ou remarques.*	*Établit dans l'outil de formulation des questionnaires.*
Étape du processus	Bouton radio	*1:En demande ; 2:En cours ; 3:Terminé*
Envoyer une invitation à l'évaluation de la prestation	Case à cocher	

Table(s) (BD) correspondante(s) : « interventions, interventions_reponses, interventions_questions, interventions_choix »

L'évaluation d'une intervention

Une fois le l'intervention clôturée, un message peut être transmis à sont instigateur l'invitant à remplir un formulaire d'évaluation de la prestation.

L'utilisateur a ainsi accès aux demandes qu'il a formulé à chaque bureau. Il peut accéder à partir de cette liste à un formulaire d'évaluation conçu par l'administrateur à l'aide de l'outil de formulation des questionnaires. L'évaluation est transmise par e-mail au bureau concerné.

Champs	Objet dans le formulaire	Commentaires
Questions d'évaluation de la prestation	*Divers objets possibles : Champ (type:text), Bouton radio, Case à cocher, Liste déroulante, Zone de texte, remarques ou note (1 à 3).*	*Établit dans l'outil de formulation des questionnaires.*

Table(s) (BD) correspondante(s) : « interventions, interventions_reponses, interventions_questions, interventions_choix »

Les statistiques

L'administrateur peut établir un rapport des réponses obtenues dans l'évaluation des prestations pour une période donnée. Sont recensées toutes les interventions qui ont eu lieu pendant cette période. Les résultats pour chaque question créée avec l'outil de formulation des questionnaires sont exposés dans un diagramme.

A l'affichage le rapport s'établit automatiquement pour la période engagée depuis le début de l'année courante.

Champs	Objet dans le formulaire	Commentaires
Date de début de la période	*Champ (type:text)*	*Obligatoire*
Date de fin de la période	*Champ (type:text)*	*Obligatoire*

Table(s) (BD) correspondante(s) : « interventions, interventions_reponses »

Fonctionnalités

1. Concevoir les questionnaires
   Ces questionnaires concernent les demandes, les évaluations comme le suivi de traitement. Les tables concernées sont **« interventions_questions » et « interventions_choix »**.

La définition des composantes des formulaires s'établit selon les références suivantes :

1	*Champ texte*	
2	*Zone de texte*	
3	*Bouton radio*	*Choix multiples à établir*
4	*Case à cocher*	*Choix multiples à établir*
5	*Liste déroulante*	*Choix multiples à établir*
6	*Évaluation (note)*	
7	*Remarques;*	

Différents formulaires sont à créer par l'administrateur. Les types de formulaire sont définis selon les références suivantes :

0	*Questionnaire préalable au demandeur*
1	*Questionnaire de la demande d'intervention*
2	*Questionnaire du processus d'intervention*
3	*Questionnaire d'évaluation de la prestation*

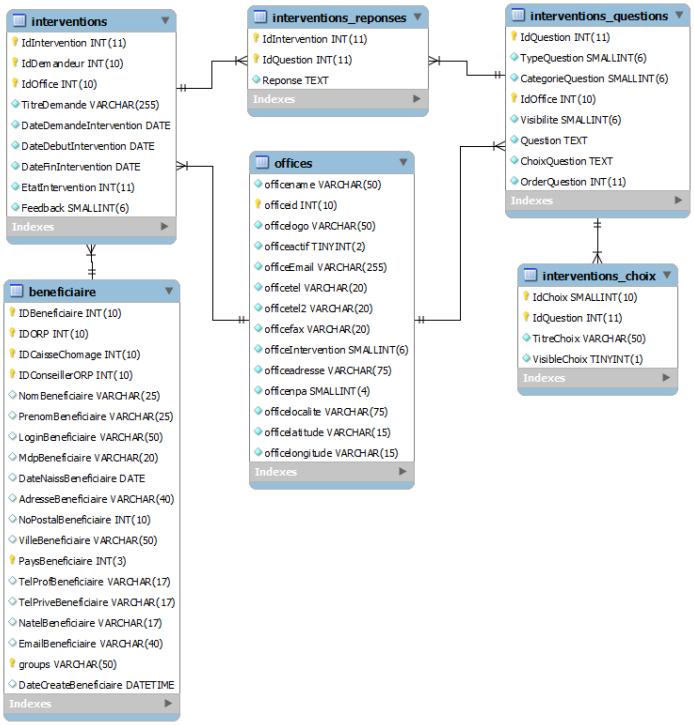
2. Suivi des interventions

Les demandes des interventions s'archivent dans la table **« interventions »** et **« interventions_reponses »**. Chaque question répondue dans un des formulaires conçu à l'aide de l'outil de formulation de questionnaires (demande d'intervention, processus d'intervention et évaluation) ajoute une ligne dans le tableau **« interventions_reponses »**. Les modifications de réponses suppriment les anciennes réponses dans le tableau et ajoute les réponses modifiées.

Le suivi du traitement des demande s'établit selon les références suivantes (correspondant au champ **« EtatIntervention »** de la table **« interventions »** :

1	*Demande d'intervention*
2	*intervention en cours*
3	*Intervention terminée.*

Table(s) de la base de données



Module « Timbrage - *timestamp* »

Présentation du module

Ce module sert à recenser les heures d'arrivée et de départ des utilisateurs. Il gère également les absences et permet d'établir des rapports de timbrage pour une période définie.

Objectifs

- Recenser l'information détaillée des entrées et sorties des utilisateurs.
- Permettre une gestion aisée des demandes d'absence.
- Faciliter le pointage et leur consultation.
- Établir des rapports clairs et faciles à consulter.

Améliorations à apporter

Dans cette nouvelle version du module, il serait nécessaire d'améliorer les éléments suivants :

- Pointage automatique de l'arrivée ou la sortie en fonction de l'heure de pointage et du précédent pointage établi par l'utilisateur.
- Définir un formulaire sécurisant les données insérées pour l'édition des types de pointage.

Interfaces

Les interfaces s'adressent à deux publics. D'une part, les administrateurs qui accèdent aux données, formulaires. D'autre part, les utilisateurs qui pointent et s'intéressent aux heures cumulées.

Liste et formulaire des types de pointage

L'administrateur a la possibilité d'établir de nouveaux types de pointage. Ceux-ci sont soit créés pour correspondre à de nouveaux motifs d'absence. Afin que ces types de pointage soit bien comprises par le système les informations suivantes doivent être indiquées :

- S'agit-il d'un pointage d'entrée ou de sortie ?
- S'agit-il d'une absence ?
- Ce pointage est-il associé à un autre (dans le cas d'une entrée, une sortie devrait lui être référée) ?

Les indications quant au type de timbrage s'établissent dans un formulaire. Ces informations sont réservées à la configuration du système de pointage. Leur modification ou leur peut entraîner des erreurs dans les rapports ou lors de l'insertion d'absence depuis l'administration. Raison pour laquelle le formulaire doit être indicatif et vérifier la validité des données inscrites. Ainsi lorsque est prévu l'insertion d'une absence à période fixe les champs pour indiquer le type de pointage de sortie sont automatiquement proposés.

Champs	Objet dans le formulaire	Commentaires
Titre du pointage	Champ (type:text)	*Obligatoire*
Entrée/sortie	Bouton radio	*0:sortie ; 1:entrée*
Type d'absence	Bouton radio	*0:présence ; 1:absence période fixe ; 2:absence période indéfinie*
Horaire	Champ (type:time)	
Sigle	Champ (type:text)	*Maximum 5 caractères*
Lien	Liste déroulante	*Première option de la liste vide*
Couleur	Champ (type:text)	*Valeur hexadécimale*

Table(s) (BD) correspondante(s) : « punchlist »

La gestion des pointages et absences

L'outil de gestion des absences se présente sous la forme de tableau de bord ayant l'apparence d'une grille qui permet d'avoir une vue d'ensemble sur les heures effectuées quotidiennement pour chaque utilisateur.

Chaque jour de cette grille est cliquable et fait apparaître un formulaire qui permet d'éditer les pointages et absences.

Description du formulaire

La pointeuse

Cet outil permet aux utilisateurs de pointer en plus de présenter en temps réel les personnes présentes et absentes et le dernier timbrage effectif. Un rafraîchissement automatique de l'interface à toutes les 45 secondes préserve l'actualité des informations.

Description du formulaire

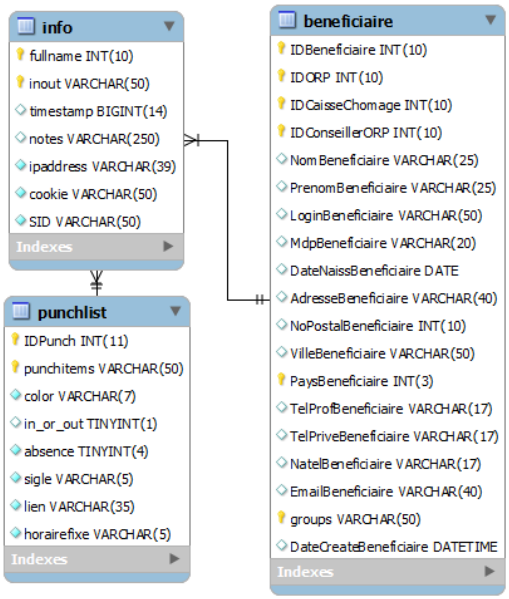
La rapport de pointage

Le rapport de pointage présente chaque pointage et absence pour une période définie en plus d'établir un calcul des heures de présence accumulées.

Description du formulaire

Fonctionnalités

Table(s) de la base de données

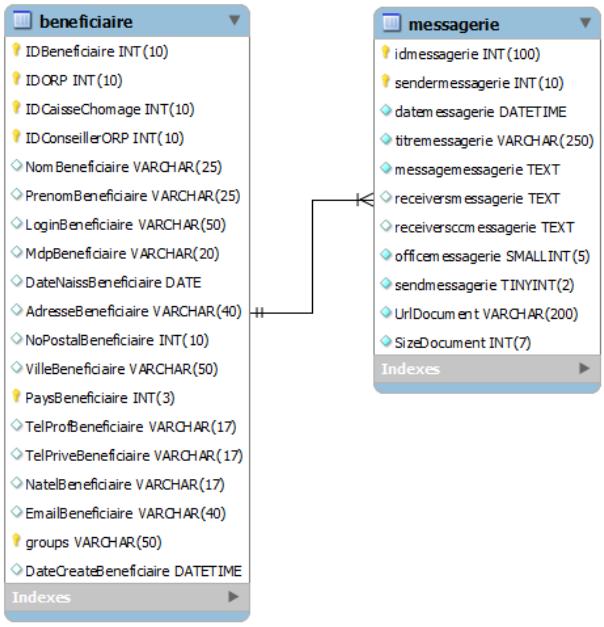


Module « Messagerie - *mailbox* »

L'un des outils utile dans la première version de l'extranet est le système de messagerie qui permet d'adresser un message à un autre utilisateur dans sa boîte e-mail personnel à partir de l'extranet.

Fonctionnalités

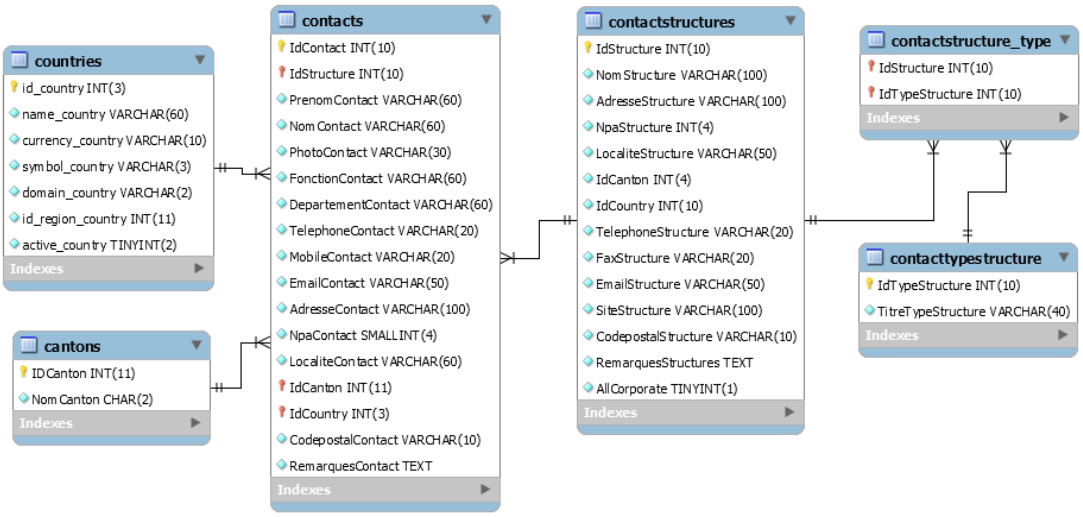
Table(s) de la base de données



Module « Contacts - *contacts* »

Fonctionnalités

Table(s) de la base de données



Module « PV - *reports* »

Présentation du module

Ce module permet d'établir des rapports de séances. Ces rapports peuvent être vus et édités par les différents bureaux et groupes associés. Les rapports disposent de thèmes dans lesquels des sujets peuvent être définis. Dans ces thèmes des libellés sont établis. Un outil de recherche (filtre) permet de faire apparaître les libellés en

fonction de dates de saisie ou de sujets. Un rapport doit également pouvoir être imprimé depuis le navigateur dans une mise en forme définie.

Objectifs

- Centraliser les informations relatives aux séances.
- Établir des rapports de séances rapidement par un outil de filtrage.
- Faciliter l'accès à l'historique d'un sujet.

Améliorations à apporter

Dans cette nouvelle version du module, aucune amélioration notoire n'est nécessaire. La nouvelle interface et les outils qu'elle dispose suffisent à améliorer la lisibilité et l'utilisation du module. Seule la liste des séances sera améliorée puisqu'elle sera visible par tous les utilisateurs.

Interfaces

Les interfaces présents dans ce module concernent principalement deux listes. Une première liste contient les rapports de séances. Les éléments figurant dans cette liste indique toutes les séances avec les bureaux et groupes qui disposent d'accès. Toutefois seules les séances dont l'utilisateur dispose de droits d'accès (groupe et bureau) pourront être accessibles et éditables.

La seconde liste dispose dans une même interface les thèmes, sujets et libellés. Cette liste dispose d'un outil de filtrage (recherche).

Formulaire de l'outil de filtrage

Champs	Objet dans le formulaire	Commentaires
Thèmes	Liste déroulante	
Date de début	Champ (type:date)	*Par défaut : date du jour*
Date de fin	Champ (type:date)	*Par défaut : date du jour*
Actif	Liste déroulante	*Au choix : Actif, inactif ou tous.*

Table(s) (BD) correspondante(s) : « pv_themes, pv_libelles »

Les formulaires du module concernent :

- l'édition d'une séance
- l'édition d'un thème
- l'édition d'un sujet
- l'édition d'un libellé.

Formulaire d'édition de séances

Champs	Objet dans le formulaire	Commentaires
Nom de la séance	Champ (type:text)	*Obligatoire (par défaut : date du jour)*
Bureaux associés	Liste de cases à cocher	*Obligatoire (par défaut : le bureau de l'utilisateur)*
Groupes associés	Liste de cases à cocher	*Obligatoire (par défaut : le groupe de l'utilisateur)*

Table(s) (BD) correspondante(s) : « pv_sujets »

Formulaire d'édition de thèmes apparaissant dans une fenêtre modale

Champs	Objet dans le formulaire	Commentaires
Nom du thème	Champ (type:text)	*Obligatoire*

Table(s) (BD) correspondante(s) : « pv_thèmes »

Formulaire d'édition de sujets apparaissant dans une fenêtre modale

Champs	Objet dans le formulaire	Commentaires
Nom du sujet	Champ (type:text)	*Obligatoire*

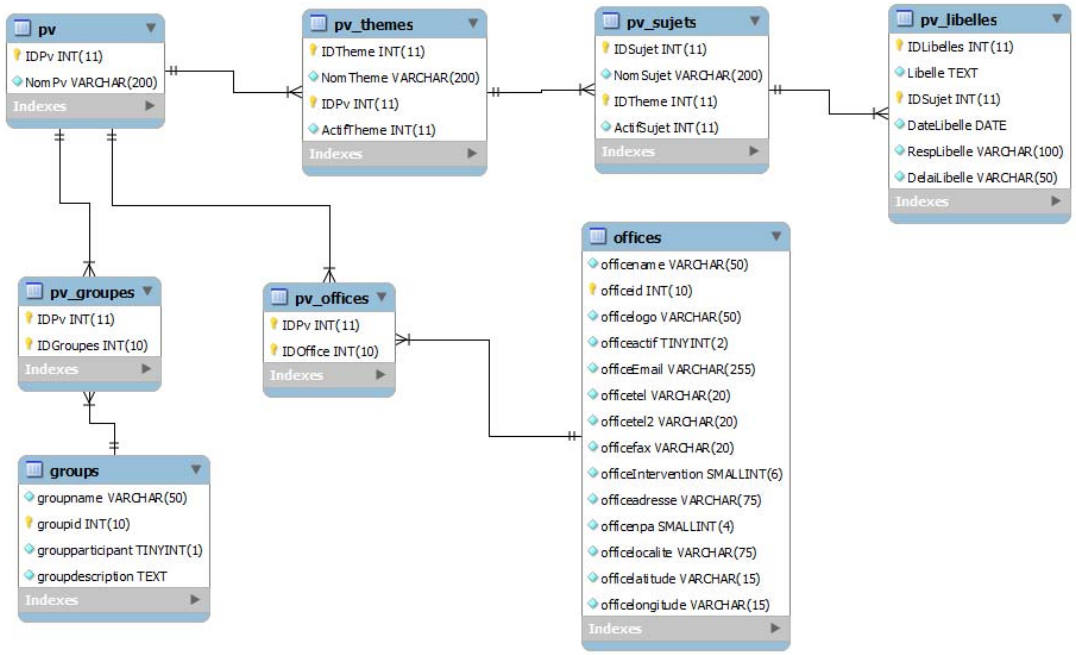
Table(s) (BD) correspondante(s) : « pv_sujets »

Formulaire d'édition de libellés apparaissant dans une fenêtre modale

Champs	Objet dans le formulaire	Commentaires
Date	Champ (type:date)	*Obligatoire (par défaut : date du jour)*
Libellé	Zone de texte	*Obligatoire*
Responsable	Champ (type:text)	
Délais	Champ (type:text)	

Table(s) (BD) correspondante(s) : « pv_libelles »

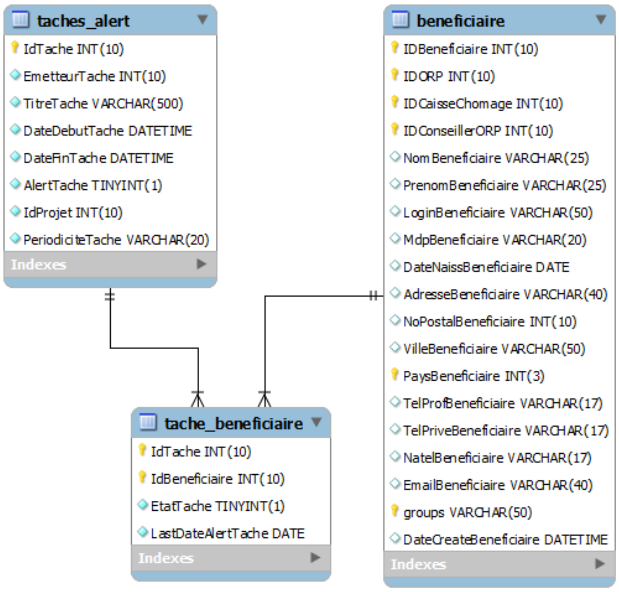
Table(s) de la base de données



Module « Agenda - *schedule* »

Fonctionnalités

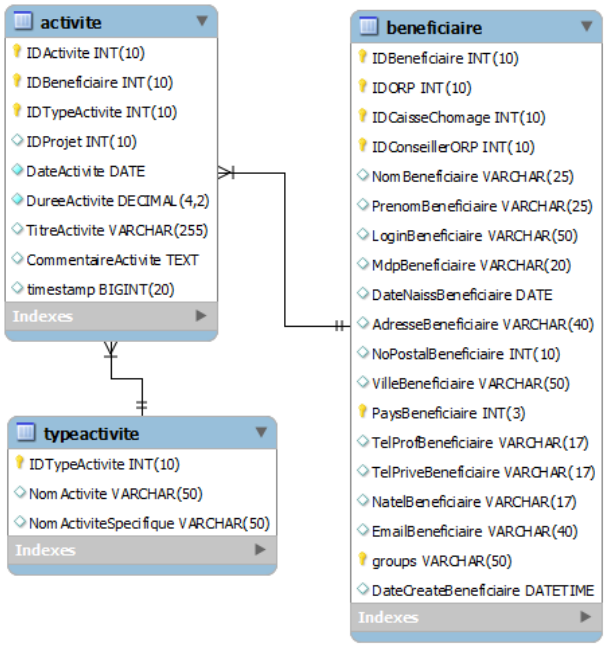
Table(s) de la base de données



Module « Activités - schedule »

Fonctionnalités

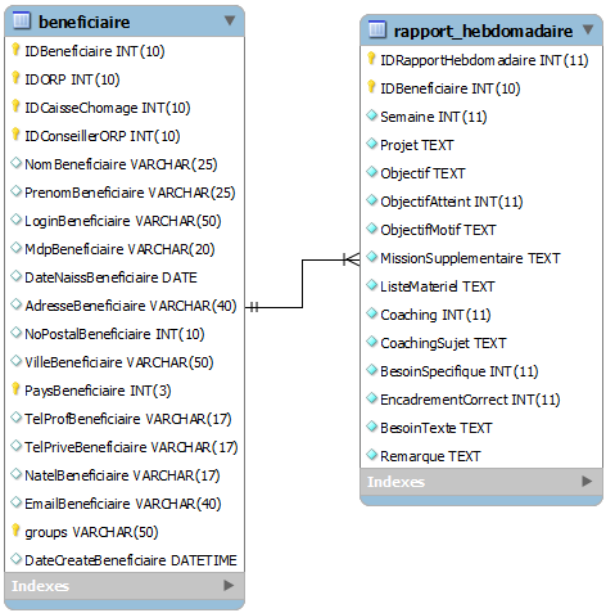
Table(s) de la base de données



Module « Activités hebdomadaires - *activityreports* »

Fonctionnalités

Table(s) de la base de données


beneficiaire
IDBeneficiaire INT(10)
IDORP INT(10)
IDCaisseChomage INT(10)
IDConseillerORP INT(10)
NomBeneficiaire VARCHAR(25)
PrenomBeneficiaire VARCHAR(25)
LoginBeneficiaire VARCHAR(50)
MdpBeneficiaire VARCHAR(20)
DateNaissBeneficiaire DATE
AdresseBeneficiaire VARCHAR(40)
NoPostalBeneficiaire INT(10)
VilleBeneficiaire VARCHAR(50)
PaysBeneficiaire INT(3)
TelProfBeneficiaire VARCHAR(17)
TelPriveBeneficiaire VARCHAR(17)
NatelBeneficiaire VARCHAR(17)
EmailBeneficiaire VARCHAR(40)
groups VARCHAR(50)
DateCreateBeneficiaire DATETIME
Indexes
rapport_hebdomadaire
IDRapportHebdomadaire INT(11)
IDBeneficiaire INT(10)
Semaine INT(11)
Projet TEXT
Objectif TEXT
ObjectifAtteint INT(11)
ObjectifMotif TEXT
MissionSupplementaire TEXT
ListeMateriel TEXT
Coaching INT(11)
CoachingSujet TEXT
BesoinSpecifique INT(11)
EncadrementCorrect INT(11)
BesoinTexte TEXT
Remarque TEXT
Indexes

Perspectives et évolutions

La mise en place d'une nouvelle architecture de développement facilite la maintenance et les futurs développements prévus pour l'extranet. Dans cette perspective, d'autres améliorations peuvent être prévues pour cet outil. Il s'agira de phases de développement qui pourront être prévus suite à la migration complète et définitive de la nouvelle mouture.

Encodage des caractères

L'encodage actuel des caractères dans la base de données et indiqué dans l'affichage des données dans la page HTML est ISO-8859-1. Il apparaîtrait utile de transformer ces données au format d'encodage UTF-8.

Base de données

L'amélioration de l'extranet devrait passer par une révision de la nomenclature de plusieurs tables et champs de la base de données afin qu'elle respecte les indications prévues dans ce document. La nouvelle architecture de développement facilite ce changement car les requêtes faites dans la base de données sont accompagnées par un « mapping » qui permet notamment de rééditer le nom des champs à des emplacements définis.