



Logiciel Otolithe

Manuel d'installation et d'utilisation

13/12/2018 - v2.1

Éric Quinton

IRSTEA - Centre de Bordeaux
50, avenue de Verdun, Gazinet
33612 CESTAS Cedex

Table des matières

1	Le logiciel Otolithe	1
1.1	Présentation	1
1.2	Fonctionnalités générales	1
1.3	Technologie employée	1
1.3.1	Base de données	1
1.3.2	Langage de développement et framework utilisé	1
1.3.3	Liste des composants externes utilisés	2
1.4	Sécurité	3
1.5	Licence	3
1.6	Copyright	3
2	Installer le logiciel	4
2.1	Consultez la documentation du framework !	4
2.2	Configurer le serveur	4
2.2.1	Configurer Apache	4
2.2.2	Modules PHP nécessaires	5
2.2.3	Configurer l'hôte virtuel et SSL	5
2.2.4	Configurer le dossier d'installation	6
2.2.5	Droits à attribuer au serveur web	8
2.3	Configurer l'application	8
2.3.1	Connexion à la base de données	8
2.3.2	Identification des utilisateurs	9
2.3.3	Configuration de l'accès à l'annuaire LDAP	10
2.3.4	Paramètres spécifiques	11
2.3.5	Paramètres stockés en base de données	11
2.4	Créer la base de données	12
2.4.1	Créer les tables de gestion des droits	12
2.4.2	Créer les tables applicatives	12
2.4.3	Login de connexion	12
2.4.4	Droits sur les tables	12
2.4.5	Scripts de modification	13
2.5	Mise en production	13
2.6	Installer une nouvelle version	13
2.6.1	Faites une sauvegarde de la base de données	13
2.6.2	Procédure automatique de mise à jour	13
2.6.3	Sauvegarder le fichier contenant les paramètres de l'application	14
2.6.4	Consultez le fichier news.txt	14
2.6.5	Mise à jour de la structure de la base de données	14
2.6.6	Reconfigurer les droits d'accès au serveur web	14
2.6.7	Supprimer les dossiers inutiles	14

2.6.8	Vérifier la configuration du chiffrement	15
3	Administrer l'application	16
3.1	Gérer les droits	16
3.1.1	Principe général	16
3.1.2	Créer un nouvel utilisateur	18
3.1.3	Créer un login utilisé dans la gestion des droits	19
3.1.4	Définir les groupes d'utilisateur	19
3.1.5	Créer une application	20
3.1.6	Définir les droits utilisables dans l'application	21
3.1.7	Cas particulier des groupes et des logins issus d'un annuaire LDAP	21
3.2	Droits spécifiques de l'application Otolithe	21
3.2.1	Droits à positionner	21
3.3	Gestion des traces	22
4	Paramétrer le logiciel	23
4.1	Créer les lecteurs	23
4.2	Créer une expérimentation	23
4.3	Importer des poissons et des pièces calcifiées	24
4.3.1	Modèle de fichier	25
4.4	Gérer les tables de paramètres	25
4.4.1	Table des espèces	26
4.4.2	Table des types de pièces calcifiées	26
5	Gérer les lectures	27
5.1	Sélectionner un poisson	27
5.1.1	Modifier un poisson	28
5.1.2	Ajouter des pièces calcifiées	28
5.2	Gestion des photos	29
5.2.1	Limitations quant au format et aux dimensions d'une photo	29
5.2.2	Repère de mesure et longueur de référence	29
5.3	Gestion des lectures	29
5.3.1	Tableau des lectures	30
5.3.2	Réaliser une lecture	30
5.3.3	Consulter les lectures	31
A	Structure de la base de données	32
	Bibliographie	41

Chapitre 1

Le logiciel Otolithe

1.1 Présentation

Le logiciel Otolithe a été conçu pour faciliter la lecture des pièces calcifiées de poissons (otolithes, écailles, rayons de nageoires, etc.). Il permet à chaque lecteur de positionner des points sur des photos, et de comparer ensuite chaque lecture.

Il a été conçu pour l'unité de recherche *Écosystèmes aquatiques et changements globaux* d'IRSTEA, à Cestas (33).

La première version est parue à l'automne 2013, la version 2.0 est sortie en août 2018.

Le code comprend environ 6400 lignes (commentaires compris), dont 2200 concernent l'affichage des pages web. Il a été écrit en PHP, les pages web sont générées en HTML et Javascript avec le composant Smarty.

1.2 Fonctionnalités générales

L'application est organisée autour de la notion d'expérimentation : une expérimentation regroupe les poissons qui font l'objet d'une lecture par un groupe de lecteurs définis.

Un poisson peut être rattaché à plusieurs expérimentations.

Les poissons peuvent être saisis ou importés, à partir d'un fichier au format CSV.

Pour un poisson, on peut décrire une ou plusieurs pièces calcifiées, et rattacher à celles-ci une ou plusieurs photos.

Les lecteurs positionnent des points sur les photos. Une fois toutes les lectures réalisées, il est possible de les afficher simultanément et, au besoin, réaliser une lecture consensuelle.

Les résultats des lectures peuvent être exportées dans un fichier au format CSV.

Les lecteurs ne peuvent accéder qu'aux poissons qui font partie des expérimentations qui leur sont ouvertes : il est ainsi possible de gérer des lots de poissons différents avec des lecteurs différents, tout en conservant la confidentialité des lectures.

1.3 Technologie employée

1.3.1 Base de données

L'application a été conçue pour fonctionner avec Postgresql, en version 9.5.

1.3.2 Langage de développement et framework utilisé

Le logiciel a été écrit en PHP, en s'appuyant sur le framework *Prototypephp* [21], développé parallèlement par l'auteur du logiciel.

Il utilise la classe *Smarty* [24] pour gérer l’affichage des pages HTML. Celles-ci sont générées en utilisant *Jquery* [11] et divers composants associés. Le rendu général est réalisé avec *Bootstrap* [6].

1.3.3 Liste des composants externes utilisés

Nom	Version	Licence	Usage	Site
PrototypePHP	4/7/2018	LGPL	Framework	github.com/ equinton/ prototypephp
Smarty	3.1.31	LGPL	Générateur de pages HTML	www.smarty.net
Smarty-gettext	1.1.1	LGPL	Gestion du multi-linguisme avec Smarty	https://github.com/smarty- gettext/smarty- gettext
PHPCAS	1.3.5	Apache 2.0	Identification auprès d’un serveur CAS	wiki.jasig.org/ display/ CASC/ phpCAS
Bootstrap	3.3.7	MIT	Présentation HTML	get.bootstrap.com
js-cookie-master	2.1.4	MIT	Gestion des cookies dans le navigateur	github.com/ js-cookie/ js-cookie
Datatables	1.10.15	MIT	Affichage des tableaux HTML	www.datatables. net
Datetime- moment		MIT	Formatage des dates dans les tableaux	datatables.net/ plug-ins/ sorting/ datetime-moment
Moment		MIT	Composant utilisé par datetime-moment	momentjs.com
JQuery	3.3.1	≈ BSD	Commandes Javascript	jquery.com
JQuery-ui	1.12.1	≈ BSD	Commandes Javascript pour les rendus graphiques	jqueryui.com
Jquery- timepicker-addon		MIT	Time picker	github.com/ trentrichardson/ jQuery- Timepicker- Addon
Magnific-popup	1.1.0	MIT	Affichage des photos	dimsemenov .com/plugins/ magnific-popup/ magnific-popup/
Smartmenus		MIT	Génération du menu HTML	www.smartmenus .org

Nom	Version	Licence	Usage	Site
AlpacaJS	1.5.23	Apache 2	Génération et saisie des métadonnées (pour une version future)	www.alpacajs.org

TABLE 1.1: Table des composants externes utilisés dans l'application

1.4 Sécurité

L'application a été conçue pour résister aux attaques dites opportunistes selon la nomenclature ASVS v3 [14] de l'OWASP [15]. Des tests d'attaque ont été réalisés en août 2018 avec le logiciel ZapProxy [16], et n'ont pas détecté de faiblesse particulière.

La gestion des droits est conçue pour qu'un lecteur ne puisse accéder qu'aux poissons faisant partie de ou des expérimentations auxquelles il est rattaché.

Hormis le droit *admin* qui permet de tout faire dans le logiciel, y compris paramétrer la gestion des droits, les droits suivants sont gérés :

- gestion : permet de modifier les données concernant un poisson, ajouter une photo, etc.
- gestionCompte : permet de rajouter des comptes, de créer une expérimentation et d'y rattacher des lecteurs.

Les lecteurs ne peuvent que réaliser des lectures ou modifier les leurs.

1.5 Licence

Le logiciel est diffusé selon les termes de la licence GNU AFFERO GENERAL PUBLIC LICENSE version 3, en date du 19 novembre 2007 [9].

1.6 Copyright

L'application est en cours de dépôt auprès de l'Agence de protection des programmes [3].

Chapitre 2

Installer le logiciel

2.1 Consultez la documentation du framework !

Le logiciel a été conçu à partir du framework *Prototypephp*. La documentation associée [20] récapitule l'ensemble des informations nécessaires pour réaliser l'installation générale (configuration du serveur, définition des droits d'accès, etc.).

De nombreuses passages ont été repris ici, mais il n'est pas inutile de se référer au document d'origine.

2.2 Configurer le serveur

L'application est conçue pour fonctionner à partir d'une adresse unique de type : *https://monsie.com*. Le chiffrement est obligatoire (protocole https). Il n'est pas possible d'installer l'application dans un sous-dossier, par exemple : *https://monsie.com/otolithe* ne fonctionnera pas.

À partir de la version 2.0, un script d'installation quasi-automatique est disponible et permettra, une fois le code publié :

- d'installer les paquetages nécessaires (Apache, PHP, Postgresql principalement) ;
- de télécharger la dernière version de l'application ;
- de créer la base de données, avec mise en place d'une sauvegarde automatique ;
- de pré-configurer le serveur pour qu'il soit prêt à être utilisé.

Pour déployer une nouvelle instance, une fois le serveur installé, dans un terminal, tapez les commandes suivantes :

```
wget https://adresse_a_definir/install/deploy_new_instance.sh
sudo -s
./deploy_new_instance.sh
```

Suivez les messages affichés à l'écran. Vous devrez notamment modifier le fichier :

```
/etc/apache2/sites-available/otolithe.conf
```

pour indiquer l'adresse DNS utilisée pour accéder à l'application et le certificat de chiffrement associé.

La configuration a été réalisée pour un serveur Linux fonctionnant avec Ubuntu 16.04 LTS Server ou Debian 9. Elle peut bien sûr être adaptée à d'autres distributions Linux. Par contre, rien n'a été prévu pour faire fonctionner l'application directement dans une plate-forme windows, même si, en théorie, cela devrait être possible.

2.2.1 Configurer Apache

Les modules suivants doivent être activés :


```
a2enmod ssl
a2enmod headers
a2enmod rewrite
```

2.2.2 Modules PHP nécessaires

Modules complémentaires nécessaires :

- *php-mbstring*
- *php-pgsql*
- *php7.0-xml*
- *php-xdebug* pour les phases de mise au point
- *php-curl* pour l'identification via un serveur CAS.

Le stockage et l'affichage des photos nécessite :

- *php-imagick*

2.2.3 Configurer l'hôte virtuel et SSL

L'application ne fonctionne qu'en mode SSL, les cookies de session n'étant pas transmis sur des liens non chiffrés. Voici un exemple de configuration à insérer dans le fichier `/etc/apache2/sites-available/default-ssl`

```
<Directory /var/www/html>
    Options FollowSymLinks MultiViews
    AllowOverride all
    Order allow,deny
    allow from all
</Directory>
SSLProtocol all -SSLv3
SSLCipherSuite ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA
-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:
ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE
-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256
-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE
-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE
-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-
SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256
-SHA:DES-CBC3-SHA:!DSS
SSLHonorCipherOrder on
SSLCompression off
SSLSessionTickets off
```

(attention : pas d'espace entre *Order allow* et la virgule).

La chaîne *SSLCipherSuite* est celle qui fonctionne avec Apache 2.4.24 et openssl 1.1.0f, et est issue du configurateur mis à disposition par la fondation Mozilla [13]. Vous pouvez également consulter le document édité par l'ANSSI [1].

Activez ensuite le mode SSL dans Apache :

```
a2ensite default-ssl
service apache2 restart
```

Cas particulier de l'identification en mode HEADER

Si vous identifiez vos utilisateurs derrière un proxy d'identification, comme LemonLdap par exemple, vous devrez limiter l'accès de l'application uniquement au proxy. La commande *Directory* devient donc :

```
<Directory /var/www/html>
  Options FollowSymLinks MultiViews
  AllowOverride all
  Order allow,deny
  allow from 10.1.2.3
</Directory>
```

10.1.2.3 correspond à l'adresse IP du serveur proxy d'identification.

2.2.4 Configurer le dossier d'installation

Le principe général est que le dossier contenant l'application contient, dans son nom, le numéro de version (v2.0 par exemple), et un lien virtuel (otolithe) pointe vers celui-ci. C'est le lien qui est la cible de l'adresse web : ainsi, à chaque nouvelle version, il suffit de mettre à jour le code de l'application et de faire pointer le lien vers le nouveau dossier pour que celle-ci soit opérationnelle.

À partir de la version 2.1, des scripts seront fournis pour réaliser automatiquement les mises à jour (dans le cas d'installations mono-instances).

Cas général : une seule instance hébergée dans le serveur

Utilisez le script fourni, qui créera automatiquement les dossiers nécessaires.

Cas particulier : faire cohabiter plusieurs instances avec le même code

Il est possible d'utiliser le même code applicatif pour alimenter des bases de données différentes (ou des données stockées dans des schémas différents). Cette fonctionnalité est basée sur l'attribution d'entrées DNS différentes.

Le mécanisme est décrit dans la figure 2.1 *Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents*, page 7.

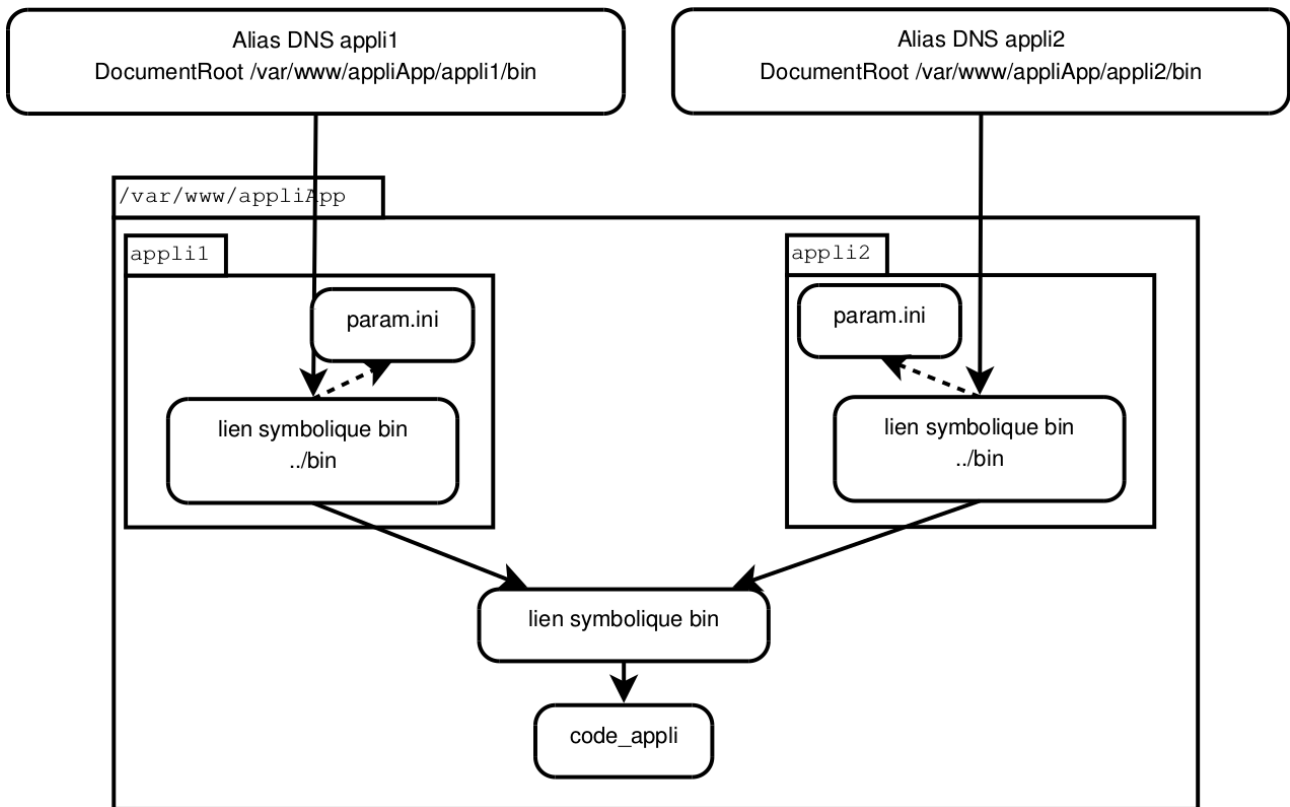


FIGURE 2.1 – Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents

Dans le paramétrage de l'alias DNS (en principe, dans `/etc/apache2/sites-available`), l'application pointe vers le dossier `/var/www/appliApp/appli1/bin`. `/var/www` correspond à la racine du site web, *appliApp* au dossier racine de l'application, *appli1* au dossier spécifique de l'alias DNS. Ce dossier *appli1* ne contient que deux fichiers : *param.ini*, qui contient les paramètres spécifiques, et *bin*, qui est un lien symbolique vers le dossier `../bin`.

Le dossier `../bin` (donc, dans `/var/www/appliApp`) est lui aussi un alias qui pointe vers le code réel de l'application, ici `code_appli`. Le fichier *param.inc.php* doit contenir les commandes suivantes pour que le fichier *param.ini* soit correctement chargé selon le contexte :

```
$chemin = substr($_SERVER["DOCUMENT_ROOT"],0, strpos($_SERVER["DOCUMENT_ROOT"],"/bin
"));
$paramIniFile = "$chemin/param.ini";
```

Le fichier *param.ini* sera cherché dans le dossier parent du code de l'application, c'est à dire soit dans *appli1*, soit dans *appli2* dans cet exemple. Il suffit qu'il contienne les paramètres adéquats pour rendre l'application utilisable dans des contextes différents à partir du même code initial.

Le fichier *param.ini* est le dernier qui est traité par l'application pour récupérer les paramètres. Ceux-ci sont lus dans l'ordre suivant :

param/param.default.inc.php → param/param.inc.php → ../param.ini

param.ini contiendra les entrées spécifiques liées au DNS utilisé pour accéder à l'application, en principe tout ou partie de celles-ci :

```
APPLI_titre="Gestion des otolithes d'EABX"
BDD_schema=col, public, gacl
BDD_login=compte_de_connexion
BDD_passwd=mot_de_passe_de_connexion
BDD_dsn=pgsql:host=serveur;dbname=base_de_donnees;sslmode=require
GACL_aco=col
```

APPLI_code=proto

Si un libellé contient une apostrophe, la chaîne doit être insérée dans des guillemets doubles, comme ici pour la variable *APPLI_titre*.

2.2.5 Droits à attribuer au serveur web

Le serveur web doit pouvoir accéder en lecture à l'ensemble des fichiers de l'application, et en écriture à deux dossiers :

- *display/templates_c* : fichier utilisé par Smarty pour compiler les modèles de documents HTML ;
- *img* : dossier de génération des images et des fichiers temporaires.

Deux scripts sont fournis pour attribuer les droits :

- **install/apache2/upgrade_rights.sh** : positionne les droits en utilisant les droits standards Linux (owner, group)
- **install/apache2/upgrade_rights_with_acl.sh** : positionne les droits à partir des ACL.

Les scripts doivent être lancés ainsi :

```
collec-2.0/install/apache2/upgrade_rights.sh v2.0
```

ou

```
collec-2.0/install/apache2/upgrade_rights_with_acl.sh v2.0
```

2.3 Configurer l'application

L'application est configurable par l'intermédiaire de trois fichiers :

param/param.default.inc.php → **param/param.inc.php** → **../param.ini**

Le premier fichier contient les paramètres par défaut. Il est systématiquement fourni à chaque nouvelle version de l'application.

Le second est spécifique de l'implémentation. Il comprend notamment les informations liées à la connexion à la base de données, à la méthode d'identification, ou à la recherche des attributs dans l'annuaire LDAP.

le troisième est destiné à offrir la possibilité d'accéder, à partir du même code applicatif, à plusieurs bases de données différentes (cf. 2.2.4 *Cas particulier : faire cohabiter plusieurs instances avec le même code*, page 6).

Voici les principaux paramètres utilisés :

2.3.1 Connexion à la base de données

Dans la pratique, deux connexions sont nécessaires : l'une pour accéder à la base des droits, l'autre aux données proprement dites. Voici les paramètres à définir :

Variable	Signification
BDD_login	compte de connexion à la base de données
BDD_passwd	mot de passe associé
BDD_dsn	adresse de la base de données sous forme normalisée
BDD_schema	schéma utilisé (plusieurs schémas peuvent être décrits, en les séparant par une virgule - fonctionnement propre à Postgresql)
GACL_dblogin	compte de connexion à la base de données des droits
GACL_dbpasswd	mot de passe associé
GACL_dsn	adresse normalisée
GACL_schema	schéma utilisé

Variable	Signification
GACL_aco	nom du code de l'application utilisé dans la gestion des droits

TABLE 2.1: Variables utilisées pour paramétrer les connexions

2.3.2 Identification des utilisateurs

Variable	Signification
ident_type	Type d'identification supporté. L'application peut gérer BDD (uniquement en base de données), LDAP (uniquement à partir d'un annuaire LDAP) LDAP-BDD (d'abord identification en annuaire LDAP, puis en base de données), CAS (serveur d'identification <i>Common Access Service</i> ¹), et enfin HEADER (identification derrière un proxy qui fournit le login dans une variable d'entête HTTP)
CAS_plugin	Nom du plugin utilisé pour une connexion CAS
CAS_address	Adresse du serveur CAS
CAS_port	Systématiquement 443 (connexion chiffrée)
CAS_CApth	chemin d'accès au certificat du serveur CAS
LDAP	tableau contenant tous les paramètres nécessaires pour une identification LDAP
ident_header_login_var	par défaut, AUTH_USER. Nom de la variable qui contiendra le login dans le cas d'une identification en mode HEADER (le radical HTTP_ ne doit pas être indiqué)
privateKey	clé privée utilisée pour générer les jetons d'identification (ré-identification automatique après une première connexion)
pubKey	clé publique utilisée pour générer les jetons d'identification
tokenIdentityValidity	durée de validité, en secondes, des jetons d'identification
MAIL_enabled	Si à 1, l'envoi de mail est géré par l'application
CONNEXION_max_attemps	nombre maximum d'essais de connexion avant blocage temporaire du compte
CONNEXION_blocking_duration	durée de blocage du compte
APPLI_mailToAdminPeriod	intervalle de temps entre l'envoi d'un mail de notification de blocage de compte à un administrateur
APPLI_admin_ttl	durée de vie d'une session d'administration (temps maximum entre deux accès à une page d'administration avant réidentification)
APPLI_lostPassword	Si à 1, autorise la récupération du mot de passe perdu, par envoi d'un mail avec un lien chiffré. Nécessite également que MAIL_enabled soit positionné à 1

TABLE 2.2: Variables utilisées pour paramétrer l'identification

1. serveur externe gérant l'identification des utilisateurs, et renvoyant à l'application le login utilisé

Ré-identification par jeton

L'application permet de conserver l'identification plus longtemps que celle définie dans le serveur, en jouant la connexion avec un jeton d'identification chiffré. Cela évite, par exemple, de devoir se ré-identifier toutes les heures si on accède au logiciel à partir d'un terminal mobile (smartphone ou tablette, par exemple).

Les trois dernières variables permettent de configurer ce mode d'identification.

Le framework peut générer un jeton chiffré après la première identification, qui sera analysé pour savoir si l'utilisateur peut être ré-identifié automatiquement.

Pour que ce mécanisme fonctionne, il faut :

- que le paramètre *tokenIdentityValidity* ait une durée de validité supérieure à la durée de vie de la session. Il est raisonnable de ne pas fixer une durée de vie supérieure à une journée de travail (10 heures). Le cookie transmis est protégé ;
- que les clés privée et publique, utilisées pour le chiffrement du jeton, soient accessibles au serveur web (variables *privateKey* et *publicKey*).

Les clés sont générées automatiquement avec le script d'installation automatique du serveur et de l'application.

Le jeton est chiffré avec la clé privée, ce qui lui permet d'être lu, le cas échéant, par l'application. Il contient le login et la date d'expiration.

Si l'utilisateur déclenche une déconnexion, le jeton est supprimé.

Pour plus d'informations, consultez comment fonctionne le mécanisme de ré-identification par jeton [22].

Identification par HEADER

Dans ce mode d'identification, le serveur web est placé derrière un serveur d'identification, appelé proxy d'identification. L'adresse de l'application pointe vers ce dernier.

Le proxy gère la connexion de l'utilisateur, et fournit à l'application le login dans une variable configurable. Cette variable est accessible dans le tableau `$_SERVER`, par exemple `$_SERVER["HTTP_AUTH_USER"]`.

Pour activer ce mécanisme, il faut modifier les paramètres suivants dans le fichier *param.ini.php* :

```
$ident_type = "HEADER";
$ident_header_login_var = "AUTH_USER";
```

la variable ne doit pas contenir la racine `HTTP_` (une fonction l'extrait automatiquement).

2.3.3 Configuration de l'accès à l'annuaire LDAP

Les paramètres LDAP sont stockés dans un tableau :

```
$LDAP = array(
    "address"=>"localhost",
    "port" => 389,
    "rdn" => "cn=manager,dc=example,dc=com",
    "basedn" => "ou=people,ou=example,o=societe,c=fr",
    "user_attr" => "uid",
    "v3" => true,
    "tls" => false,
    "groupSupport"=>true,
    "groupAttrib"=>"supannentiteaffectation",
    "commonNameAttrib"=>"displayname",
    "mailAttrib"=>"mail",
    'attributgroupname' => "cn",
```

```
'attributloginname' => "memberuid",
'basedngroup' => 'ou=example,o=societe,c=fr'
);
```

L'application peut non seulement identifier les utilisateurs auprès de l'annuaire LDAP, mais également récupérer les groupes auxquels ils appartiennent dans celui-ci.

Voici les paramètres à indiquer dans ce cas de figure (valable en principe pour tout annuaire compatible OpenLdap) :

Variable	Signification
address	adresse de l'annuaire
port	389 en mode non chiffré, 636 en mode chiffré
rdn	compte de connexion, si nécessaire
basedn	base de recherche des utilisateurs
user_attrib	nom du champ contenant le login à tester
v3	toujours à <i>true</i>
tls	<i>true</i> en mode chiffré
groupSupport	true si l'application recherche les groupes d'appartenance du login dans l'annuaire
groupAttrib	Nom de l'attribut contenant la liste des groupes d'appartenance
commonNameAttrib	Nom de l'attribut contenant le nom de l'utilisateur
mailAttrib	Nom de l'attribut contenant l'adresse mail de l'utilisateur
attributgroupname	Attribut contenant le nom du groupe lors de la recherche des groupes (cn par défaut)
attributloginname	attribut contenant les membres d'un groupe
basedngroup	base de recherche des groupes

TABLE 2.3: Variables utilisées pour paramétrer l'accès à l'annuaire LDAP

2.3.4 Paramètres spécifiques

Variable	Signification
APPLI_photoStockage	dossier contenant les photos générées par l'application, avant transmission au navigateur. Par défaut : <i>img</i>
APPLI_maxfilesize	Taille maximale des photos téléchargeables. Par défaut : 100000000

TABLE 2.4: Variables spécifiques

2.3.5 Paramètres stockés en base de données

À partir de la version 2, certains paramètres peuvent être stockés dans la base de données, pour éviter qu'ils ne soient dépendants de la configuration du serveur.

Ces paramètres sont accessibles depuis le menu *administration*, item *Paramètres de l'application*.

Voici la liste des paramètres actuellement décrits :

Variable	Signification
APPLI_title	Titre de l'application tel qu'il figure entre l'icône et le menu

TABLE 2.5: Paramètres stockés dans la base de données

2.4 Créer la base de données

La base de données est composée de deux schémas : l'un pour stocker les informations d'identification, les droits d'accès et les traces, l'autre pour les données proprement dites.

Le schéma *public* ne devrait jamais être utilisé pour stocker l'information : réservez-le pour les composants communs, comme Postgis.

Les tables de gestion des droits peuvent être communes à plusieurs jeux / applications différentes : la variable *GACL_aco* permet de séparer la gestion des droits pour chaque application, tout en travaillant à partir des mêmes utilisateurs (répartis le cas échéant dans des groupes différents selon le jeu de données considéré).

Les scripts de création des schémas dans la base de données sont stockés dans le dossier *install/pgsql*.

2.4.1 Créer les tables de gestion des droits

Script à utiliser : *gac_create_2.0.sql*. Les tables nécessaires vont être créées dans le schéma *gac* (ne modifiez pas le nom du schéma).

Le script crée un compte d'administration par défaut :

— login : **admin**

— mot de passe : **password**

Il devra être supprimé quand un autre compte d'administration aura été créé.

2.4.2 Créer les tables applicatives

Script à utiliser : *otolithe_create-2.0.sql*.

Par défaut, le script crée un schéma appelé *otolithe*. Il est possible de créer plusieurs schémas différents, si l'application supporte plusieurs jeux de données (cf. 2.2.4 *Cas particulier : faire cohabiter plusieurs instances avec le même code*, page 6). Dans ce cas de figure, remplacez *otolithe* par le nom du schéma voulu dans les deux premières lignes du script.

2.4.3 Login de connexion

Si la base de données et l'application sont hébergés dans deux serveurs différents, il est fortement conseillé de créer deux logins de connexion, un pour le schéma des droits, l'autre pour les schémas applicatifs. Ces logins ne doivent pouvoir être utilisés que depuis le serveur web hébergeant l'application.

Cette opération est possible en modifiant le fichier */etc/postgresql/9.5/main/pg_hba.conf* selon ce principe :

```
# Connexions pour les serveurs web
host nom_database userGac1 adresse_serveur/32 md5
host nom_database userData adresse_serveur/32 md5
```

Le login utilisé dans *userGac1* correspond à la variable *\$GACL_dblogin*, et *userData* à *\$BDD_login*. et en rechargeant ensuite la configuration de Postgresql avec la commande :

```
service postgresql reload
```

2.4.4 Droits sur les tables

Le compte utilisé pour la connexion au schéma des droits doit pouvoir modifier les informations présentes dans l'ensemble des tables de *gac*. Il ne devrait pas pouvoir accéder aux autres schémas

(hormis *public*), sauf en cas d'installation mono-serveur (base de données hébergée dans la même machine et connexion à distance impossible).

Le compte utilisé pour accéder aux schémas des données doit pouvoir modifier l'ensemble des informations dans les schémas de données, et lire la table *gacl.aclgroup*.

Le plus simple est d'utiliser le logiciel *pgAdmin* [17] pour attribuer les droits.

Le script d'installation automatique rend le compte *otolith* propriétaire de la base de données, ce qui simplifie la gestion des droits sur les tables.

2.4.5 Scripts de modification

Lors de la livraison de nouvelles versions, il est possible que des scripts de modification soient livrés pour mettre à niveau la base de données. Ces scripts doivent être exécutés dans tous les schémas contenant des données applicatives (pour plus de détails, consultez ci-après *Installer une nouvelle version*).

2.5 Mise en production

Une fois l'application configurée, et après avoir créé un nouveau compte d'administration :

- supprimez le compte *admin*, livré par défaut, qui ne doit pas être conservé. Sa désactivation n'est pas suffisante : si pour une raison ou pour une autre le compte est réactivé, n'importe qui pourra récupérer les droits totaux ;
- supprimez le dossier *install* qui contient les scripts de création des tables ;
- déplacez le dossier *database*, qui contient la documentation d'installation et de configuration (elle n'a pas à rester accessible depuis le site web) ;
- faites une revue des droits, pour vous assurer que tout est correctement configuré.

Vous pouvez également tester si la configuration du serveur est correcte en recourant à *ZAPProxy* [16], qui analysera la communication entre le serveur et un navigateur et identifiera les problèmes éventuels de non conformité (mauvaise réécriture des entêtes HTML suite à une mauvaise configuration du serveur Apache, par exemple).

2.6 Installer une nouvelle version

2.6.1 Faites une sauvegarde de la base de données

Il arrive fréquemment que la structure de la base de données évolue. Avant toute opération, assurez-vous de disposer d'une sauvegarde, dans un autre support.

Un programme de sauvegarde est disponible dans *install/pgsql/backup.sh*. Vous pouvez l'exécuter manuellement ainsi :

```
su postgres -c "install/pgsql/backup.sh"
```

La sauvegarde sera stockée dans */var/lib/postgresql/backup*.

Si vous avez utilisé le script d'installation automatique, le programme est également présent dans */var/lib/postgresql*.

2.6.2 Procédure automatique de mise à jour

Si l'application est installée dans un serveur dédié à cet usage (installation réalisée avec le script https://github.com/Irstea/otolith/blob/master/install/deploy_new_instance.sh), un script de mise à jour automatique peut être téléchargé. Il est disponible dans le dossier <https://github.com/Irstea/otolith/tree/master/install>, et est sous la forme :

```
upgrade-2.0-2.1.sh
```

où **2.0** correspond à la version courante de votre installation, et **2.1** à la version cible.

Pour utiliser ce script :

```
sudo -s
wget https://github.com/Irstea/otolithe/raw/master/install/upgrade-2.0-2.1.sh
chmod +x upgrade-2.0-2.1.sh
./upgrade-2.0-2.1.sh
```

Le script va télécharger le code de la nouvelle version et mettra à jour la base de données, si c'est nécessaire.

2.6.3 Sauvegarder le fichier contenant les paramètres de l'application

Le fichier *param/param.inc.php* contient vos paramétrages spécifiques. Lors de l'installation d'une nouvelle version, il va être supprimé.

Faites-en une copie, et remettez-le en place après avoir installé la nouvelle version.

2.6.4 Consultez le fichier news.txt

Le fichier *param/news.txt* contient la description des modifications apportées au logiciel. Il précise notamment si une mise à jour de la base de données doit être appliquée.

2.6.5 Mise à jour de la structure de la base de données

Le dossier *install/pgsql* contient les scripts de création et de mise à jour de la base de données. Les scripts de mise à jour sont nommés ainsi :

```
col_alter_versionAnterieure-versionMiseAJour.sql
```

versionAnterieure correspond à la version la plus ancienne qui doit être mise à jour, *versionMiseAJour* la version cible. Par exemple :

```
otolithe_alter_2.0-2.1.sql
```

indique que toutes les versions entre 2.0 et 2.1 doivent être mises à jour avec le script indiqué. Si vous avez « sauté » certaines versions du logiciel, il est possible que plusieurs scripts doivent être appliqués.

La mise à jour doit être appliquée dans tous les schémas contenant des données, notamment dans le cas où le même logiciel est utilisé pour gérer plusieurs jeux de données.

Avant d'exécuter les scripts, vérifiez leur contenu, et notamment le nom des schémas.

Ne relancez jamais l'exécution d'un script.

2.6.6 Reconfigurer les droits d'accès au serveur web

Après installation de la nouvelle version du code, n'oubliez-pas de reconfigurer les accès en lecture pour le compte utilisé pour faire fonctionner le serveur web, et en écriture pour les dossiers *img* et *display/templates_c* (cf. 2.2.5 Droits à attribuer au serveur web, page 8).

2.6.7 Supprimer les dossiers inutiles

Une fois la mise en production validée, supprimez le dossier *install* et déplacez le dossier *database*, et faites une revue des droits pour vous assurer qu'il n'y a pas eu de modification intempestive ou que la configuration est toujours correcte.

2.6.8 Vérifier la configuration du chiffrement

Avec un navigateur récent, ou en testant le site (s'il est accessible depuis internet) à partir de SSLLABS, vérifiez que l'application soit correctement configurée, notamment au niveau du serveur Apache.

Chapitre 3

Administrer l'application

3.1 Gérer les droits

Depuis la version 1.1, les scripts de création des bases de données intègrent la génération initiale des groupes et des droits associés, ceci afin de faciliter la phase de mise en route.

Toutefois, vous devrez créer des groupes d'utilisateurs correspondant à vos projets, et modifier ensuite les projets pour donner les droits adéquats aux groupes créés.

3.1.1 Principe général

Les droits sont gérés selon le principe initialement utilisé dans la bibliothèque PHPGACL [5], aujourd'hui obsolète.

Les logins sont déclarés dans des groupes organisés de manière hiérarchique : un groupe hérite des droits attribués à ses parents.

Les droits utilisés dans le logiciel sont associés à des groupes. Il est possible d'attribuer plusieurs droits à un même groupe, et un droit peut être détenu par des groupes différents.

Si le paramètre *\$LDAP["groupSupport"]* est positionné à *true*, les groupes dont fait partie le compte LDAP sont également récupérés. Si ces groupes se voient attribués des droits, les comptes associés les récupéreront automatiquement.

Voici le schéma des tables utilisées pour gérer les droits :

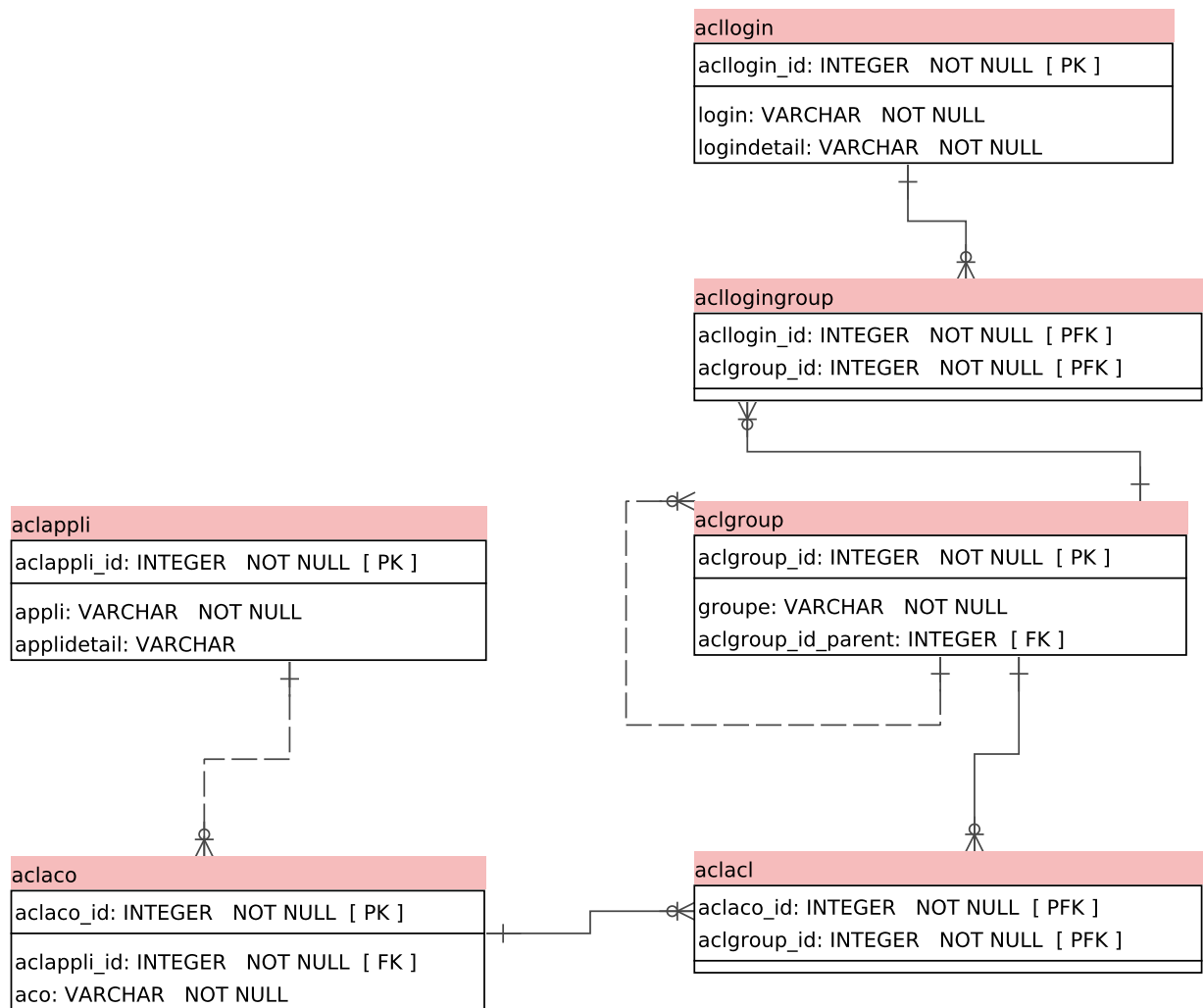


FIGURE 3.1 – Schéma des tables utilisées pour gérer les droits

Voici la description des tables :

acllogin : liste des logins utilisés. Si un compte est créé dans la base locale d'identification, un enregistrement est également créé dans cette table. Pour les identifications LDAP ou CAS, ils doivent être identiques. Si seuls les groupes LDAP sont utilisés pour un compte, il n'a pas besoin d'être décrit ici ;

aclappli : liste des applications gérées. Il est possible de gérer, à partir de la même base de données, plusieurs ensembles de droits, qui utilisent les mêmes logins.

aclaco : liste des droits déclarés dans l'application ;

aclgroup : liste des groupes contenant les logins, et qui détiennent les droits. Un groupe peut hériter d'un autre groupe. Les droits associés au groupe parent sont également attribués au groupe hérité ;

aclloggingroup : table permettant de déclarer les logins associés à un groupe ;

aclacl : table décrivant les droits détenus par un groupe.

Le module d'administration permet de saisir toutes ces informations. Il faut que l'utilisateur dispose du droit *admin*, c'est à dire qu'il fasse partie du groupe *admin* (configuration par défaut à l'initialisation de la base des droits) pour pouvoir accéder à ces fonctions.

3.1.2 Créer un nouvel utilisateur

Les utilisateurs peuvent être issus soit de l'annuaire LDAP, soit de la base interne. Pour créer un nouvel utilisateur dans la base locale :

- *Administration* → *Liste des comptes*
- *Nouveau login*
- renseignez au minimum le login.

The screenshot shows a user creation form with the following elements:

- Login ***: Input field containing 'test'.
- Nom**: Empty input field.
- Prénom**: Empty input field.
- Mèl**: Empty input field.
- Date**: Input field containing '28/06/2016'.
- Mot de passe**: Password input field with a key icon on the right.
- Répétez le mot de passe**: Password input field with a key icon on the right.
- Générez un mot de passe aléatoire**: Section containing a green 'Générer' button and an empty input field.
- actif**: Radio buttons for 'oui' (selected) and 'non'.
- Buttons**: 'Valider' (blue) and 'Supprimer' (red) buttons at the bottom.

FIGURE 3.2 – Écran de saisie d'un login de connexion

Pour créer le mot de passe, vous pouvez cliquer sur le bouton *Générer*, qui en générera un automatiquement. Envoyez-le par mèl à son destinataire (par *copier-coller*), en lui demandant de le modifier à la première connexion (icône en forme de clé, dans le bandeau, en haut à droite).

Les mots de passe doivent respecter les règles suivantes :

- ils doivent avoir une longueur minimale de 8 caractères ;
- ils doivent comprendre trois types de caractères différents parmi les minuscules, majuscules, chiffres et caractères de ponctuation ;
- ils ne peuvent pas être réutilisés pour le même login ;
- les mots de passe n'expirent pas.

Les mots de passe sont stockés sous forme d'empreinte, calculée en rajoutant un sel¹ et encodés en SHA256 : ils ne peuvent pas être retrouvés en cas de perte.

1. chaîne de caractère rajoutée au mot de passe – en général le login ou un identifiant – qui permet d'éviter que deux mots de passe identiques, associés à deux logins différents, aient la même empreinte

L'application n'intègre pas de module permettant de régénérer automatiquement un mot de passe en cas de perte : c'est au responsable applicatif d'en fournir un nouveau.

La création d'un compte entraîne la création d'une entrée identique dans la table des *aclogin*, utilisée pour attribuer les droits.

Pour désactiver temporairement un compte, sélectionnez *non* dans la zone *actif*. Si le compte ne doit plus être utilisé, supprimez-le.

Attention : si le compte disposait des droits d'administration, assurez-vous que vous avez toujours un compte disposant des mêmes droits avant la suppression.

3.1.3 Créer un login utilisé dans la gestion des droits

Indépendamment du compte de connexion, qui peut être soit issu de la base interne, soit récupéré auprès d'un annuaire LDAP ou d'un serveur CAS, l'application a besoin de connaître les utilisateurs pour pouvoir leur attribuer des droits.

À partir du menu, choisissez *Administration* → *ACL - logins*.

Vous pouvez modifier un login existant ou en créer un nouveau. Dans ce cas, vous devrez indiquer au minimum le login utilisé (identique à celui qui est employé pour la connexion à l'application : base de données interne, annuaire LDAP, serveur CAS).

Modification d'un login (module de gestion des droits)

[Retour à la liste des logins](#)

*Donnée obligatoire

Droits attribués

consult

param

FIGURE 3.3 – Écran de modification d'un login dans le module de gestion des droits

Sous l'écran de saisie figurent la liste des droits attribués à un login (en modification, le calcul n'est réalisé qu'à l'affichage de la page).

3.1.4 Définir les groupes d'utilisateur

Les groupes d'utilisateurs sont gérés selon un mécanisme d'héritage. Un groupe de haut niveau hérite des groupes précédents : si des droits ont été attribués à un groupe de niveau inférieur, un login associé à un groupe de niveau supérieur les récupère également.

Pour définir les groupes, dans le menu, choisissez *Administration* → *ACL - groupes de logins*.

Nouveau groupe racine...

Nom du groupe	Nombre de logins déclarés	Rajouter un groupe fils
consult		+
gestion	8	+
gestionCompte	3	+
admin	4	+

FIGURE 3.4 – Liste des groupes de logins

Ainsi, le login déclaré dans le groupe *admin* récupérera les droits attribués aux groupes *gestion-Compte*.

Pour créer un groupe, il existe deux possibilités :

- soit le groupe est à la base d’une nouvelle branche : utilisez alors *Nouveau groupe racine...* ;
- soit le groupe hérite d’un autre groupe : cliquez sur le signe + (*Rajouter un groupe fils*).

Vous pouvez indiquer les logins qui sont rattachés à ce groupe.

3.1.5 Créer une application

Le moteur utilisé pour faire fonctionner le logiciel Otolithe permet de gérer des droits différents pour des jeux de données différents, à partir du même code applicatif. Chaque couple *logiciel* ↔ *base de données* constitue donc une *application*, au sens de la gestion des droits.

Il est ainsi possible, à partir de la même base de données, de définir des droits différents selon les jeux de données utilisés (un jeu de données correspond à un schéma de base de données comprenant l’intégralité des tables applicatives).

À partir du menu, choisissez *Administration* → *ACL - droits* :


Modifier	Nom de l'application	Description
	otolithe	

FIGURE 3.5 – Liste des applications déclarées

Pour créer une nouvelle application, choisissez *Nouvelle application....*

[Retour à la liste des applications](#)

*** Nom de l'application :**

Description :

*Donnée obligatoire

FIGURE 3.6 – Écran de saisie d’une application

Le nom de l’application doit impérativement correspondre à la valeur *\$GACL_appli* dans les fichiers de paramètres : c’est ce qui permet au framework de savoir quels droits appliquer.

3.1.6 Définir les droits utilisables dans l'application

À partir de la liste des applications, cliquez sur le nom de celle pour laquelle vous voulez définir les droits utilisables. À partir de la liste, sélectionnez *Nouveau droit...*



FIGURE 3.7 – Écran de saisie des droits associés à une application

Le nom du droit doit être celui défini dans le corps de l'application (les droits sont positionnés dans les fichiers *param/actions.xml*, qui contient la liste des modules utilisables, et *param/menu.xml*, qui sert à générer le menu – cf. table 3.1 *Droits à positionner*, page 22).

Indiquez les groupes d'utilisateurs qui seront associés au droit courant.

3.1.7 Cas particulier des groupes et des logins issus d'un annuaire LDAP

Si vous avez paramétré l'application pour qu'elle s'appuie sur un annuaire LDAP pour gérer l'affectation des utilisateurs dans les groupes, vous n'êtes pas obligés de les déclarer explicitement dans le module de gestion des droits.

Droits attribués à un groupe LDAP

Tous les utilisateurs d'un groupe héritent d'un droit dans l'application.

- définissez le nom du groupe (en respectant la casse) dans le tableau des groupes d'utilisateurs (par exemple, EABX);
- sélectionnez le nom de ce groupe dans les droits utilisables;
- tous les utilisateurs de l'annuaire LDAP récupéreront automatiquement les droits attribués à ce groupe.

Droits attribués à un utilisateur particulier de l'annuaire LDAP

Un utilisateur s'identifie auprès de l'annuaire LDAP, mais dispose de droits particuliers.

- créez son login dans la gestion des droits;
- rajoutez-le dans le groupe d'utilisateurs adéquat.

3.2 Droits spécifiques de l'application Otolithe

3.2.1 Droits à positionner

Voici les droits nécessaires pour faire fonctionner correctement l'application :

Droit	Usage
admin	Gestion des utilisateurs et des droits

Droit	Usage
gestionCompte	Ajout d'un utilisateur (lecteur), création d'une expérimentation, rattachement d'un lecteur à une expérimentation, importation de poissons dans une expérimentation, suppression d'une lecture
gestion	Ajout d'un poisson dans une expérimentation, de photos, etc.
consult	Droit technique, permettant la consultation des informations générales, sans possibilité de modification. Les lecteurs disposent de ce droit
lecteur	Réalisation d'une lecture d'une photo, avec possibilité de supprimer ses propres lectures. Droit attribué par défaut à tout lecteur. C'est un droit qui est positionné automatiquement par le logiciel dès lors que le lecteur a été enregistré

TABLE 3.1: Liste des droits utilisés

Ces droits doivent être définis pour chaque application (couple *logiciel* \leftrightarrow *base de données*) gérée par la base de gestion des droits.

3.3 Gestion des traces

Tous les appels lancés par les utilisateurs vers les modules de l'application sont enregistrés dans la table *gacl.log*, qui ne doit être accessible qu'aux personnes dûment autorisées. Les traces sont supprimées au bout d'un an (script de nettoyage exécuté lors de la connexion d'un utilisateur).

Voici un exemple de trace générée :

```
log_id login nom_module log_date commentaire ipaddress
log_id login nom_module log_date commentaire ipaddress
5349 eric.quinton otolithe-photoGetPhoto 2018-08-27 11:17:29 ok 10.33.64.101
5347 eric.quinton otolithe-photolectureSwap 2018-08-27 11:17:26 ok
10.33.64.101
5348 eric.quinton otolithe-photolectureDisplay 2018-08-27 11:17:26 ok
10.33.64.101
5346 eric.quinton otolithe-photoGetThumbnail 2018-08-27 11:17:18 ok
10.33.64.101
5345 eric.quinton otolithe-photoDisplay 2018-08-27 11:17:17 ok 10.33.64.101
5344 eric.quinton otolithe-photoGetThumbnail 2018-08-27 11:17:14 ok
10.33.64.101
5343 eric.quinton otolithe-pieceDisplay 2018-08-27 11:17:13 ok 10.33.64.101
```

La colonne *commentaire* précise des informations concernant soit la connexion, soit l'écriture en base de données : dans ce cas, l'identifiant considéré est indiqué. L'adresse IP est en principe celle de l'utilisateur, y compris en prenant en compte le passage par un serveur Reverse-proxy².

Parallèlement, les messages d'erreur sont envoyés au processus Linux SYSLOG, qui enregistre les traces dans le fichier */var/log/apache2/error.log*.

2. serveur mis en entrée du réseau privé, qui permet de masquer les adresses internes et de contrôler les accès depuis Internet

Chapitre 4

Paramétrer le logiciel

4.1 Créer les lecteurs

La liste des lecteurs peut être consultée depuis le menu *Administration* → *Lecteurs*.



The screenshot shows a web form for creating a user. It has three input fields: 'Nom : *' with the value 'Daverat', 'Prénom :' with the value 'Françoise', and 'Login de connexion : *' with the value 'francoise.daverat'. Below these is a section titled 'Expérimentations autorisées :'. It contains a list of checkboxes for different experiments: 'Aloson 2013' (checked), 'Aloson 2014' (checked), 'Aloson 2015' (checked), 'Aloson 2016' (unchecked), and 'CAPTUREMER_2017' (unchecked).

FIGURE 4.1 – Détail d’un lecteur

Les informations à indiquer pour chaque lecteur sont les suivantes :

- son nom (obligatoire)
- son prénom
- son login de connexion (issu de la base de données interne ou de l’annuaire LDAP, le cas échéant)

Il suffit de cocher les expérimentations auxquelles il aura droit pour lui permettre de réaliser des lectures.

4.2 Créer une expérimentation

Les expérimentations sont les éléments de base de l’application. Elles sont utilisées pour gérer les droits attribués aux lecteurs.

Pour créer une expérimentation, choisissez *Paramètres* → *Expérimentations*. Vous pouvez modifier une expérimentation existante ou en créer une nouvelle.

Nom : * Immortel argentée française

Description :

Date de début : 01/01/2013

Date de fin : 31/12/2014

Liste des lecteurs rattachés

- ☐ Marie-Laure Acolas
- ☐ Bernadette Bounket
- ☐ Sarah Bureau du Colombier
- ☐ Georges Carrel
- ☒ Françoise Daverat
- ☐ Christine Gazeau
- ☐ Debora Heroin
- ☒ Philippe Jatteau
- ☐ Marion Philippon
- ☒ Éric Quinton
- ☐ Marine Randon
- ☐ Nathalie Reynaud
- ☒ Christian Rigaud
- ☒ Eric Rochard

Valider **Supprimer**

FIGURE 4.2 – Écran de saisie/modification d’une expérimentation

La liste des lecteurs potentiels est affichée, il suffit de sélectionner ceux qui seront retenus pour lire les photos de l’expérimentation considérée.

À noter que seul le nom de l’expérimentation est obligatoire.

4.3 Importer des poissons et des pièces calcifiées

Pour éviter une saisie qui peut être fastidieuse, le logiciel permet d’importer un lot de poissons et leurs pièces calcifiées associées. L’importation automatique des photos n’est par contre pas possible.

Pour importer des poissons, choisissez *Lectures* → *Import*.

* Nom du fichier à importer (CSV) : Aucun fichier sélectionné.

Séparateur utilisé :

Encodage du fichier :

FIGURE 4.3 – Écran de gestion des importations de poisson

L'importation est réalisée en deux étapes. Pendant la première, la conformité du fichier est vérifiée : si des anomalies sont détectées, elles sont affichées.

Si aucune anomalie n'est détectée, l'importation peut être déclenchée.

Voici la liste des colonnes qui peuvent être utilisées :

- **exp_id** : code numérique de l'expérimentation (obligatoire)
- **espece_id** : code de l'espèce (obligatoire)
- **tag** : N° de l'étiquette posée sur le poisson (le *codeindividu* ou le *tag* sont obligatoires)
- **codeindividu** : identifiant du poisson ou hptag
- **sexe_id** : sexe du poisson (1 : mâle, 2 : femelle, 3 : juvénile, 4 : indifférencié)
- **longueur** : longueur du poisson (mm)
- **poids** : poids du poisson (g)
- **remarque** : remarques concernant le poisson
- **parasite** : parasites éventuels rencontrés
- **age** : âge du poisson
- **piecetype_id** : code du type de pièce calcifiée à analyser Consultez la liste des types de pièces
- **piececode** : si le type de pièce est indiqué, vous pouvez renseigner un code spécifique attaché à la pièce
- **peche_date** : date de la pêche, au format aaaa-mm-dd ou dd/mm/aaaa
- **site** : site de la pêche
- **zonesite** : zone précise de la pêche (précision concernant le site)
- **campagne** : campagne de pêche
- **peche_engin** : engin utilisé, sous forme textuelle
- **personne** : références du pêcheur
- **opérateur** : références de l'opérateur ayant traité le poisson

les dernières informations (site, zonesites, etc.) sont stockées sous forme de chaîne de caractères, il n'y a pas de tables qui leur sont dédiées.

4.3.1 Modèle de fichier

Un modèle de fichier utilisable pour l'importation peut être téléchargé. Le lien figure en bas de la page web.

4.4 Gérer les tables de paramètres

Deux tables de paramètres sont accessibles depuis l'application : la table des types de pièces calcifiées et la table des espèces.

4.4.1 Table des espèces

La liste des espèces gérées par le logiciel est accessible depuis le menu *Paramètres* → *Espèces*. Il est possible de modifier ou rajouter un taxon (nom latin et nom français uniquement) :



FIGURE 4.4 – Écran de modification d'un taxon

Le code affiché dans la liste est celui à utiliser pour les importations.

4.4.2 Table des types de pièces calcifiées

La liste des types de pièces peut être étendue depuis le menu *Paramètres* → *Pièces*. Seul le nom de la pièce est à indiquer.

Id	Nom
1	Otolithe
2	Otolithe gauche
3	Otolithe droit
4	Pectorale gauche
5	Pectorale droite
6	Rayon
7	Ecaille

FIGURE 4.5 – Liste des types de pièces calcifiées

Le code affiché (colonne *Id*) est celui à utiliser pour les importations.

Chapitre 5

Gérer les lectures

5.1 Sélectionner un poisson

Le menu *Lectures* permet d'accéder à une fenêtre de recherche des poissons.

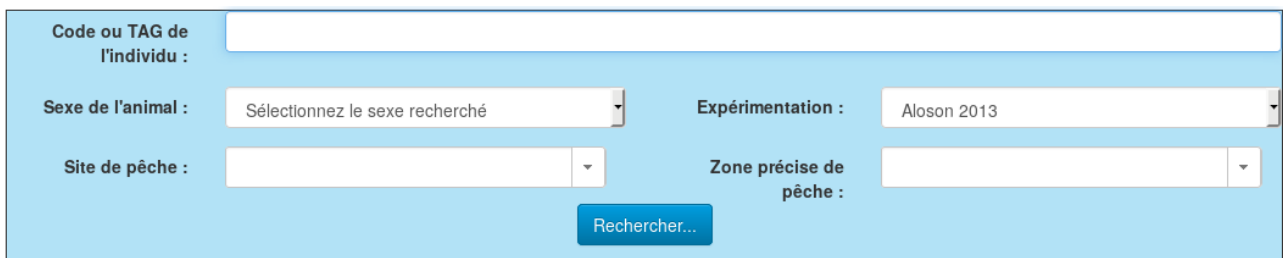
La figure montre une interface de recherche pour les poissons. Elle est présentée sous la forme d'un formulaire à onglets avec un onglet actif de couleur bleu clair. Le formulaire contient cinq champs de saisie : un champ de texte pour le 'Code ou TAG de l'individu', un menu déroulant pour le 'Sexe de l'animal' (avec le texte 'Sélectionnez le sexe recherché'), un menu déroulant pour l' 'Expérimentation' (avec 'Aloson 2013' sélectionné), un menu déroulant pour le 'Site de pêche' et un menu déroulant pour la 'Zone précise de pêche'. Un bouton bleu 'Rechercher...' est situé en bas au centre du formulaire.

FIGURE 5.1 – Fenêtre de recherche des poissons

Seules les expérimentations pour lequel l'utilisateur dispose des droits de lecture sont affichées. Il est possible de rechercher les poissons en utilisant un des deux codes disponibles (code de l'individu ou TAG).

La sélection d'un poisson permet d'atteindre sa page de détail :

Données générales

Modifier...

Code de l'individu : 601
Tag : 955000003734216
Espèce : Anguilla anguilla
Sexe : Femelle
Longueur (mm) : 576
Poids (g) : 311.5
Remarque :
Parasite :
Age :

Données concernant la pêche

Site : Certes
Zone précise : Certes
Date de pêche : 18/12/2012
Campagne :
Engin : Verveux
Pêcheur : Glese
Opérateur : Sarah

Pièces rattachées

Nouvelle pièce

Type	Code	Traitement réalisé	Nbre photos rattachées
Otolithe gauche	Oto601g	grinding, polishing, staining	1

Expérimentation(s)

Immortel argentée française

FIGURE 5.2 – Fenêtre de détail d'un poisson

5.1.1 Modifier un poisson

Si les droits sont suffisants, l'utilisateur peut modifier les informations concernant un poisson.

Retour à la liste

Retour au poisson

Individu

Espèce : *

Anguilla anguilla
Sexe :
Femelle
Code de l'individu : 601
Tag : 955000003734216

Liste des expérimentations de rattachement

☐ Aloson 2013
☐ Aloson 2014
☐ Aloson 2015
☐ Aloson 2016
☐ CAPTUREMER_2017

FIGURE 5.3 – Fenêtre de modification d'un poisson

La sélection de l'espèce se fait en tapant quelques caractères de l'espèces à chercher (nom latin ou français), puis en sélectionnant dans la liste déroulant l'information pertinente.

À noter que cet écran permet de l'associer avec une ou plusieurs expérimentations (au moins une à sélectionner – ne pas supprimer l'expérimentation initiale sous peine de « perdre » le poisson et de ne pas pouvoir le retrouver par l'application).

5.1.2 Ajouter des pièces calcifiées

Si elles n'ont pas été créées lors de l'importation, les pièces calcifiées peuvent être rajoutées manuellement. Seul le type de la pièce est obligatoire.

5.2 Gestion des photos

L'ajout d'une photo n'est possible que depuis le détail d'une pièce calcifiée. Il est possible d'en ajouter autant que nécessaire pour la même pièce, mais la lecture n'est réalisable que photo par photo.

5.2.1 Limitations quant au format et aux dimensions d'une photo

Si le format TIFF est souvent plébiscité pour son absence de perte d'informations, il présente l'inconvénient de ne pas être supporté par les navigateurs et ne peut pas être utilisé pour réaliser les lectures. De plus, il est assez gourmand en espace de stockage.

Le logiciel autorise toutefois l'importation de photos au format TIFF, sachant que celles-ci seront transformées en JPG au moment de la lecture (le format d'origine est conservé dans la base de données).

Les logiciels associés aux microscopes peuvent générer des photos au format TIFF mal formées (messages d'erreur lors de leur ouverture). Si une photo n'arrive pas à être importée, il convient de l'ouvrir avec le logiciel GIMP [25], puis l'enregistrer de nouveau : les messages d'erreur seront supprimés lors de cette étape. Ce logiciel est également très efficace pour réduire la taille d'une photo trop volumineuse.

Pour des raisons de performance, il est déconseillé d'importer des photos de plus de 50 Mo, celles-ci étant retraitées par le serveur avant d'être stockées.

Les photos sont stockées dans la base de données dans leur format original et sous forme de miniature, pour réduire les temps de traitement.

5.2.2 Repère de mesure et longueur de référence

Il est courant d'insérer un repère de mesure sur une photo. En connaissant sa longueur en pixels (dans la photo d'origine), il est alors facile de calculer des distances sur la photo, et notamment les écartements entre les points lors de la lecture.

Si ces deux informations sont renseignées, les lecteurs n'auront pas besoin de mesurer individuellement la taille de la longueur de référence pour recalculer les dimensions mesurées sur la photo.

5.3 Gestion des lectures

Il est possible de réaliser autant de lectures que nécessaire, celles-ci étant datées.

Pour des questions de performances, la taille de la photo transmise au navigateur peut volontairement être réduite. Les résolutions par défaut proposées sont les suivantes :

- 800x600
- 1024x768
- 1280x1024
- 1600x1300
- taille originale

Les lecteurs peuvent choisir la résolution qui leur convient le mieux, sachant que plus la photo est détaillée, plus elle est volumineuse.

Le placement des points sur la photo est recalculé en tenant compte du facteur de réduction : le logiciel permet ainsi de visualiser l'ensemble des lectures effectuées quelle que soit la résolution utilisée par chaque lecteur.

5.3.1 Tableau des lectures

Consultations individuelles, globales, modifications avec visualisation des points déjà tracés

Afficher l'age calculé (nbre de points positionnés - 1) par chaque lecteur

Modification unique	Lecteur	Date de lecture	Résolution	Longueur de référence mesurée	Longueur totale lue	Longueur réelle calculée	Lecture consensuelle	Supprimer	Consulter...	Lecture à modifier
	Eric Rochard	04/03/2016 11:05:34	800x2103		2274.3154686449			✗	<input checked="" type="checkbox"/>	<input type="radio"/>
	Sarah Bureau du Colombier	19/12/2013 10:32:13	800x2103		1799.0933727478			✗	<input checked="" type="checkbox"/>	<input type="radio"/>
	Françoise Daverat	17/12/2013 10:09:15	800x2103		1112.846127629			✗	<input checked="" type="checkbox"/>	<input type="radio"/>
	Françoise Daverat	17/12/2013 10:04:13	800x2103		1012.4539034269			✗	<input checked="" type="checkbox"/>	<input type="radio"/>
	Christian Rigaud	16/12/2013 10:51:43	800x2103		1702.1674597326			✗	<input checked="" type="checkbox"/>	<input type="radio"/>

Affichage de l'élément 1 à 5 sur 5 éléments

Résolution (approximative) d'affichage : 800x600 Facteur de transparence des cercles affichés : Opaque

Avec création d'une nouvelle lecture : ☐ Déclencher l'affichage des lectures sélectionnées, avec ou sans création/modification d'une lecture

FIGURE 5.4 – Tableau récapitulatif des lectures réalisés sur une photo

Le tableau des lectures permet de consulter l'ensemble des lectures réalisées, soit individuellement, soit globalement. Le formulaire en bas de tableau permet ainsi de créer une nouvelle lecture en affichant celles qui sont sélectionnées, par exemple pour créer une lecture consensuelle.

Par défaut, pour ne pas influencer les lecteurs, le nombre de segments déterminés par les lecteurs précédents n'est pas affiché. Il suffit de cliquer sur le lien *Afficher l'age calculé (nbre de points positionnés - 1) par chaque lecteur* pour obtenir le nombre de segments identifiés.

Pour faciliter la visualisation des points sur les photos, il est possible d'ajuster leur facteur de transparence, depuis opaque jusqu'à totalement transparent.

5.3.2 Réaliser une lecture

L'écran est organisé en plusieurs parties. Le haut affiche la photo, où pourront être placés les points. Sous la photo, un formulaire permet de modifier le comportement de la lecture ou de préciser des informations complémentaires. Enfin, la page se termine par le tableau des points saisis.

Un clic sur la photo positionne un point, dont les coordonnées seront affichées dans le tableau en bas d'écran. Pour supprimer un point, double-cliquez sur celui-ci.

Type de lecture

Le logiciel permet de positionner un premier point (initial) plus grand que les autres, pour identifier une zone centrale correspondant au stade larvaire (par exemple).

Pour cela, positionnez le *type de lecture pour le prochain point à Point initial avec cercle élargi*. La taille du cercle sera celle indiquée dans *Rayon (en pixels) du cercle élargi*.

Il est également possible de positionner sur la photo une ligne pour aider à placer les points. Choisissez *Tracé d'une ligne sur la photo (aide à la mesure)*, et placez deux points. Un trait sera affiché sur la photo, et vous pourrez positionner vos points le long de celui-ci (décalage d'un pixel au minimum pour des questions techniques). Attention : cette ligne n'est pas sauvegardée.

Enfin, si cette information n'a pas été indiquée dans le détail de la photo, le lecteur devra positionner deux points de part et d'autre de la longueur de référence pour pouvoir calculer le plus précisément possible les distances entre chaque point. Pour cela, choisissez l'option *Mesure de la longueur de référence*.

Numérotation des points

Les points sont numérotés de 10 en 10. Il est possible d'insérer un point entre deux autres, mais il est alors nécessaire de modifier son numéro pour qu'il s'insère correctement au moment des calculs.

Toutefois, si le premier point saisi est celui au centre de la pièce calcifiée (le point de base), le logiciel peut recalculer automatiquement l'ordre des points en recherchant celui qui est le plus près, sans tenir compte de l'ordre de numérotation. C'est le fonctionnement par défaut.

Données complémentaires

Des informations complémentaires peuvent être renseignées :

- nature de la strie finale : hyaline (ou vitreuse), obscure, ou non déterminée
- fiabilité de la lecture : 0 pour très incertaine, 0,5 pour incertaine, et 1 pour fiable
- lecture consensuelle : à renseigner s'il s'agit de la lecture de vérification
- année de naissance estimée : à partir du nombre de segments et de la date de pêche du poisson (rappelée dans le cartouche en haut d'écran), il est possible de déterminer l'année de naissance

Légende

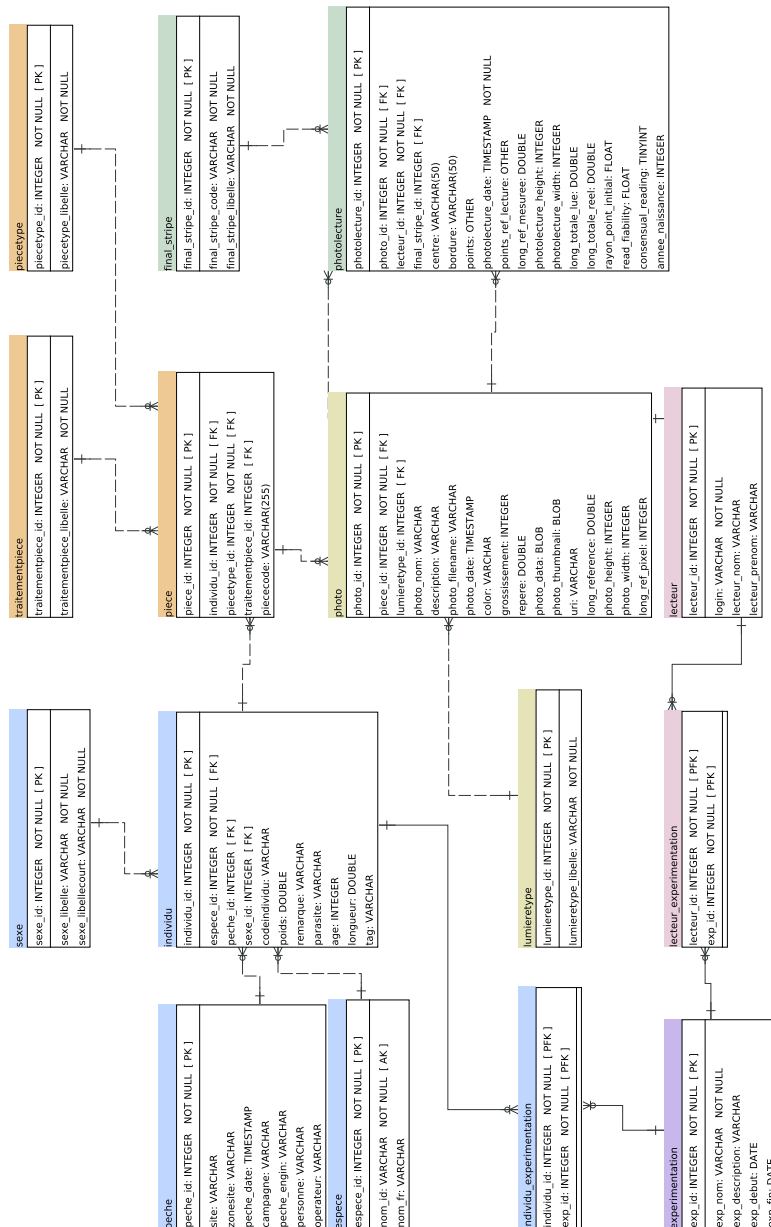
Si plusieurs lectures sont affichées, la légende présente les codes de couleur utilisés pour représenter les points positionnés par chaque lecteur, ainsi que les différents paramètres saisis par chacun.

5.3.3 Consulter les lectures

Il est possible de consulter une photo avec les lectures associées, sans pouvoir créer de points. Seules la photo et la légende sont affichées.

Annexe A

Structure de la base de données



otolithe

List of tables

- [espece](#)
- [experimentation](#)
- [final_stripe](#)
- [individu](#)
- [individu_experimentation](#)
- [lecteur](#)
- [lecteur_experimentation](#)
- [lumieretype](#)
- [peche](#)
- [photo](#)
- [photolecture](#)
- [piece](#)
- [piecetype](#)
- [sexe](#)
- [traitementpiece](#)

espece (Physical Name: espece)

Liste des especes

Logical Column Name	Physical Column Name	Type	PK	Nullable
espece_id (PK)	espece_id	INTEGER	PK	NOT NULL
nom_id	nom_id	VARCHAR(0)		NOT NULL
nom_fr	nom_fr	VARCHAR(0)		

Referenced By

- [individu](#) referencing (espece_id)

experimentation (Physical Name: experimentation)

Experimentation a laquelle est rattache le poisson

Logical Column Name	Physical Column Name	Type	PK	Nullable
exp_id (PK)	exp_id	INTEGER	PK	NOT NULL
exp_nom	exp_nom	VARCHAR(0)		NOT NULL

exp_description	exp_description	VARCHAR(0)
exp_debut	exp_debut	DATE
exp_fin	exp_fin	DATE

Referenced By

- [lecteur_experimentation](#) referencing (exp_id)
- [individu_experimentation](#) referencing (exp_id)

final_stripe (Physical Name: final_stripe)

Natures de la strie finale

Logical Column Name	Physical Column Name	Type	PK	Nullable
final_stripe_id (PK)	final_stripe_id	INTEGER	PK	NOT NULL
final_stripe_code	final_stripe_code	VARCHAR(0)		NOT NULL
Code utilisé				
final_stripe_libelle	final_stripe_libelle	VARCHAR(0)		NOT NULL

Referenced By

- [photolecture](#) referencing (final_stripe_id)

individu (Physical Name: individu)

Logical Column Name	Physical Column Name	Type	PK	Nullable
individu_id (PK)	individu_id	INTEGER	PK	NOT NULL
espece_id (FK)	espece_id	INTEGER		NOT NULL
peche_id (FK)	peche_id	INTEGER		
sexe_id (FK)	sexe_id	INTEGER		
codeindividu	codeindividu	VARCHAR(0)		
poids	poids	DOUBLE		
remarque	remarque	VARCHAR(0)		
parasite	parasite	VARCHAR(0)		
age	age	INTEGER		
longueur	longueur	DOUBLE		

tag tag VARCHAR(0)

References

- [peche](#) through (peche_id)
- [espece](#) through (espece_id)
- [sexe](#) through (sexe_id)

Referenced By

- [individu_experimentation](#) referencing (individu_id)
- [piece](#) referencing (individu_id)

individu_experimentation (Physical Name: individu_experimentation)

Logical Column Name	Physical Column Name	Type	PK	Nullable
individu_id (PK) (FK)	individu_id	INTEGER	PK	NOT NULL
exp_id (PK) (FK)	exp_id	INTEGER	PK	NOT NULL

References

- [experimentation](#) through (exp_id)
- [individu](#) through (individu_id)

lecteur (Physical Name: lecteur)

personne realisant la lecture d'une photo

Logical Column Name	Physical Column Name	Type	PK	Nullable
lecteur_id (PK)	lecteur_id	INTEGER	PK	NOT NULL
login	login	VARCHAR(0)		NOT NULL
lecteur_nom	lecteur_nom	VARCHAR(0)		
lecteur_prenom	lecteur_prenom	VARCHAR(0)		

Referenced By

- [lecteur_experimentation](#) referencing (lecteur_id)
- [photolecture](#) referencing (lecteur_id)

lecteur_experimentation (Physical Name: lecteur_experimentation)

Table des experimentations autorisees pour un lecteur

Logical Column Name	Physical Column Name	Type	PK	Nullable
lecteur_id (PK) (FK)	lecteur_id	INTEGER	PK	NOT NULL
exp_id (PK) (FK)	exp_id	INTEGER	PK	NOT NULL

References

- [lecteur](#) through (lecteur_id)
- [experimentation](#) through (exp_id)

lumieretype (Physical Name: lumieretype)

Logical Column Name	Physical Column Name	Type	PK	Nullable
lumieretype_id (PK)	lumieretype_id	INTEGER	PK	NOT NULL
lumieretype_libelle	lumieretype_libelle	VARCHAR(0)		NOT NULL

Referenced By

- [photo](#) referencing (lumieretype_id)

peche (Physical Name: peche)

Date de peche et lieu de capture

Logical Column Name	Physical Column Name	Type	PK	Nullable
peche_id (PK)	peche_id	INTEGER	PK	NOT NULL
site	site	VARCHAR(0)		
zonesite	zonesite	VARCHAR(0)		
peche_date	peche_date	TIMESTAMP		

campagne	campagne	VARCHAR(0)
peche_engin	peche_engin	VARCHAR(0)
personne	personne	VARCHAR(0)
opérateur	opérateur	VARCHAR(0)

Referenced By

- [individu](#) referencing (peche_id)

photo (Physical Name: photo)

photos associées à une pièce

Logical Column Name	Physical Column Name	Type	PK	Nullable
photo_id (PK)	photo_id	INTEGER	PK	NOT NULL
piece_id (FK)	piece_id	INTEGER		NOT NULL
lumieretype_id (FK)	lumieretype_id	INTEGER		
photo_nom	photo_nom	VARCHAR(0)		
description	description	VARCHAR(0)		
photo_filename	photo_filename	VARCHAR(0)		
photo_date	photo_date	TIMESTAMP		
color	color	VARCHAR(0)		
grossissement	grossissement	INTEGER		
repere	repere	DOUBLE		
photo_data	photo_data	BLOB		
photo_thumbnail	photo_thumbnail	BLOB		
uri	uri	VARCHAR(0)		
long_reference	long_reference	DOUBLE		
photo_height	photo_height	INTEGER		
Hauteur de la photo originale				
photo_width	photo_width	INTEGER		
Largeur de la photo originale				
long_ref_pixel	long_ref_pixel	INTEGER		
Longueur de référence en pixels - valeur par défaut pour photolecture, si non lu				

References

- [piece](#) through (piece_id)
- [lumieretype](#) through (lumieretype_id)

Referenced By

- [photolecture](#) referencing (photo_id)

photolecture (Physical Name: photolecture)

Lecture realisee par une personne

Logical Column Name	Physical Column Name	Type	PK	Nullable
photolecture_id (PK)	photolecture_id	INTEGER	PK	NOT NULL
photo_id (FK)	photo_id	INTEGER		NOT NULL
lecteur_id (FK)	lecteur_id	INTEGER		NOT NULL
final_stripe_id (FK)	final_stripe_id	INTEGER		
centre	centre	VARCHAR(50)		
bordure	bordure	VARCHAR(50)		
points	points	[1111]		
photolecture_date	photolecture_date	TIMESTAMP		NOT NULL
points_ref_lecture	points_ref_lecture	[1111]		
Emplacement des points utilises pour lire la longueur de reference				
long_ref_mesuree	long_ref_mesuree	DOUBLE		
photolecture_height	photolecture_height	INTEGER		
Hauteur de la photo utilisee pour la lecture				
photolecture_width	photolecture_width	INTEGER		
Largeur de la photo affichee pour realiser la lecture				
long_totale_lue	long_totale_lue	DOUBLE		
Somme des segments entre chacun des points				
long_totale_reel	long_totale_reel	DOUBLE		
Longueur totale réelle calculée pour le lecteur (long_reference / long_ref_mesuree * long_totale_lue)				
rayon_point_initial	rayon_point_initial	FLOAT		
read_fiability	read_fiability	FLOAT		
Fiabilité de la lecture				
consensual_reading	consensual_reading	TINYINT		
1 si lecture consensuelle				
annee_naissance	annee_naissance	INTEGER		
Année de naissance estimée				

References

- [lecteur](#) through (lecteur_id)
- [final_stripe](#) through (final_stripe_id)
- [photo](#) through (photo_id)

piece (Physical Name: piece)

Pieces analysees

Logical Column Name	Physical Column Name	Type	PK	Nullable
piece_id (PK)	piece_id	INTEGER	PK	NOT NULL
individu_id (FK)	individu_id	INTEGER		NOT NULL
piecetype_id (FK)	piecetype_id	INTEGER		NOT NULL
traitementpiece_id (FK)	traitementpiece_id	INTEGER		
piececode	piececode	VARCHAR(255)		

References

- [piecetype](#) through (piecetype_id)
- [traitementpiece](#) through (traitementpiece_id)
- [individu](#) through (individu_id)

Referenced By

- [photo](#) referencing (piece_id)

piecetype (Physical Name: piecetype)

Type de piece

Logical Column Name	Physical Column Name	Type	PK	Nullable
piecetype_id (PK)	piecetype_id	INTEGER	PK	NOT NULL
piecetype_libelle	piecetype_libelle	VARCHAR(0)		NOT NULL

Referenced By

- [piece](#) referencing (piecetype_id)

sexe (Physical Name: sexe)

Logical Column Name	Physical Column Name	Type	PK	Nullable
sexe_id (PK)	sexe_id	INTEGER	PK	NOT NULL
sexe_libelle	sexe_libelle	VARCHAR(0)		NOT NULL
sexe_libellecourt	sexe_libellecourt	VARCHAR(0)		NOT NULL

Referenced By

- [individu](#) referencing (sexe_id)

traitementpiece (Physical Name: traitementpiece)

Logical Column Name	Physical Column Name	Type	PK	Nullable
traitementpiece_id (PK)	traitementpiece_id	INTEGER	PK	NOT NULL
traitementpiece_libelle	traitementpiece_libelle	VARCHAR(0)		NOT NULL

Referenced By

- [piece](#) referencing (traitementpiece_id)

Bibliographie

- [1] ANSSI. Recommandations de sécurité relatives à tls, 2016. URL http://www.ssi.gouv.fr/uploads/2016/09/guide_tls_v1.1.pdf.
- [2] Apache-FOP. The apache fop project, 2016. URL <http://xmlgraphics.apache.org/fop/>.
- [3] APP. Agence pour la protection des programmes, 2016. URL <http://www.app.asso.fr>.
- [4] ArchLinux. Clamav, 2016. URL <https://wiki.archlinux.org/index.php/ClamAV>.
- [5] Mike Benoit and Dan Cech. Phpgacl, generic access control lists, 2006. URL <http://phpgacl.sourceforge.net>.
- [6] bootstrap. Bootstrap is the most popular html, css, and js framework for developing responsive, mobile first projects on the web, 2016. URL <http://getbootstrap.com>.
- [7] Cisco. Clamav® is an open source antivirus engine for detecting trojans, viruses, malware and other malicious threats, 2015. URL <http://www.clamav.net/>.
- [8] Justin Ellingwood. How to set up master slave replication on postgresql on an ubuntu 12.04 vps, 2013. URL <https://www.digitalocean.com/community/tutorials/how-to-set-up-master-slave-replication-on-postgresql-on-an-ubuntu-12-04-vps>.
- [9] Free Software Foundation. Gnu affero general public license, 2007. URL <https://www.gnu.org/licenses/agpl.html>.
- [10] Nils Hamerlinck. Configurer la rÉplication d'un serveur postgresql, 2015. URL <http://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-184/Configurer-la-replication-d-un-serveur-PostgreSQL>.
- [11] JQuery. Site officiel, 2015. URL <http://jquery.com/>.
- [12] Stanislas Lange. Installer php 7 sous debian 8 jessie via le dépôt dotdeb, 2016. URL <https://angristan.fr/installer-php-7-debian-8-jessie-depot-dotdeb/>.
- [13] Mozilla. Mozilla ssl configuration generator, 2016. URL <https://mozilla.github.io/server-side-tls/ssl-config-generator/>.
- [14] OWASP. Application security verification standard (2014), 2014. URL https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf.
- [15] OWASP. Site institutionnel, 2015. URL <https://www.owasp.org>.
- [16] OWASP. Zed attack proxy project, 2015. URL https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project.
- [17] pgAdmin. pgadmin postgresql tools, 2016. URL <https://pgadmin.org>.

- [18] postgresql. Binary replication tutorial, 2015. URL https://wiki.postgresql.org/wiki/Binary_Replication_Tutorial.
- [19] Eric Quinton. Php - utiliser clamav pour rechercher les virus dans les documents téléversés, 2016. URL <http://www.linux-professionnel.net/programmation/php---codes-divers/php---utiliser-clamav-pour-rechercher-les-virus-dans-les-documents-televerses>.
- [20] Eric Quinton. Documentation d'utilisation du framework prototype php, 2016. URL <https://github.com/equinton/prototypephp/blob/bootstrap/database/documentation/prototypephp-documentation.pdf>.
- [21] Eric Quinton. Prototypephp, 2016. URL <https://github.com/equinton/prototypephp/tree/bootstrap>.
- [22] Eric Quinton. Ré-identification par jeton, 2016. URL <http://www.linux-professionnel.net/programmation/php---codes-divers/re-identification-par-jeton>.
- [23] Greg Reinacker. Zero to postgresql streaming replication in 10 mins, 2013. URL <http://www.rassoc.com/gregr/weblog/2013/02/16/zero-to-postgresql-streaming-replication-in-10-mins>.
- [24] smarty. Smarty, php template engine, 2016. URL <http://www.smarty.net>.
- [25] The Gimp Team. Gimp - gnu image manipulation program, 2018. URL <https://www.gimp.org/>.