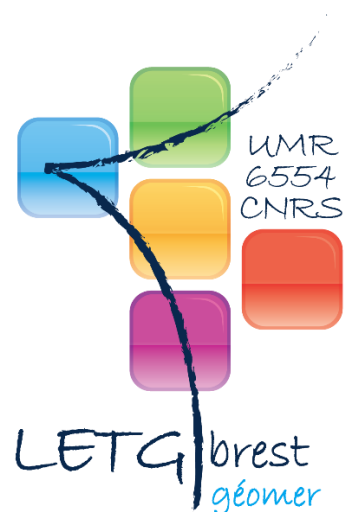




INSTITUT
UNIVERSITAIRE
EUROPÉEN
DE LA MER



Rapport de stage

Mise à jour et développement de nouvelles fonctionnalités sur l'application web ManageChart.

Table des matières

Introduction.....	4
LETG-Brest Géomer	4
L'IUEM	4
Description des applications	5
Indigéo.....	5
ManageChart.....	6
Analyse du cahier des charges	7
Analyse de l'existant.....	7
Les données.....	7
Dictionnaire des données.....	7
Règles de gestion.....	8
Modèle conceptuel des données	9
Contraintes d'intégrité	10
Dépendances fonctionnelles	10
Utilisateurs et droits.....	11
Les vues	12
Diagrammes de cas d'utilisations.....	21
Plan de navigation	22
Diagramme d'activités de l'authentification	23
Technologies utilisées.....	23
Framework et librairies	24
Planification	25
Projet de stage.....	26
Actions demandées	26
Modifications apportées aux données.....	27
Modifications apportées aux vues	28
Fonctionnalités ajoutées	34
Ajout du type d'axes X « Catégorie »	34
Déploiement d'un serveur d'export « HighCharts » côté client.....	35
Ajout d'un système de pagination sur les résultats retourné par les requêtes	36
Ajout d'une fonctionnalité permettant l'export au format CSV des résultats retourné par les requêtes	38
Ajout d'une option de tri sur les tableaux listant avec un tri par « id » décroissant	39

Mise en place d'une solution permettant la gestion de graphiques avec de multiples axes Y pour les graphiques temporels	39
Ajout de YUI Compressor pour la compression des scripts et fichiers de style	43
Autres améliorations apportées.....	44
Amélioration de la gestion des messages d'erreurs retourné par les requêtes	44
Ajout d'un champ « Nom de connexion » dans les formulaires d'enregistrement et d'édition de connexion à une base de données	45
Correction sur l'export au format image d'un graphique	46
Correction sur la façon dont s'ouvre l'URL du crédit	46
Suppression du « navigator »	47
Correction sur le déploiement de la version de production de l'application	47
Technologies utilisées	47
Framework et librairies	48
Conclusion	48
Remerciements	48
Définition d'un graphique	51
Fonction wichQuery	53
Requête avec pagination	54
Pagination dynamique.....	57

Introduction

Dans le cadre de la présentation du titre de « Développeur Logiciel », j'ai effectué mon stage de fin de formation au sein de l'antenne Brestoise de l'Unité Mixte de Recherche LETG du CNRS (Littoral, Environnement, Géomatique, Télédétection). Ce laboratoire de géographie de l'environnement fait partie de l'IUEM (Institut Universitaire Européen de la Mer) regroupant l'ensemble des laboratoires de l'UBO (Université de Bretagne Occidentale) travaillant sur le milieu marin.

Sous la tutelle de Mr Rouan, j'ai travaillé sur une application web, « ManageChart », ayant pour objectif l'édition de représentations graphiques à partir de bases de données scientifiques. Chaque représentation graphique générée est accessible à partir d'une [URL](#) paramétrable et intégrable dans une seconde application web « Indigéo (Infrastructure scientifique de données et d'informations géo-spatialisées sur l'environnement) » à l'aide d'iframe.

L'objectif de mon stage consistait en l'ajout de nouvelle fonctionnalité, la mise à jour des différentes librairies et dépendances utilisées, la correction de bogues et l'amélioration de la gestion des exceptions.

LETG-Brest Géomer

Le LETG-Brest Géomer regroupe des enseignants et chercheurs en géographie ainsi que des ingénieurs. Leurs activités de recherche sont axées sur l'environnement littoral des écosystèmes tempérés et tropicaux, sous six thèmes :

- Dynamiques de l'occupation des sols,
- Dynamiques géomorphologiques des littoraux,
- Risques côtiers,
- Fréquentations et usages,
- Modélisation des activités humaines,
- Géomatique.

L'IUEM

L'IUEM est un institut de recherche dédié à l'océan et au littoral dont les objectifs sont les suivants :

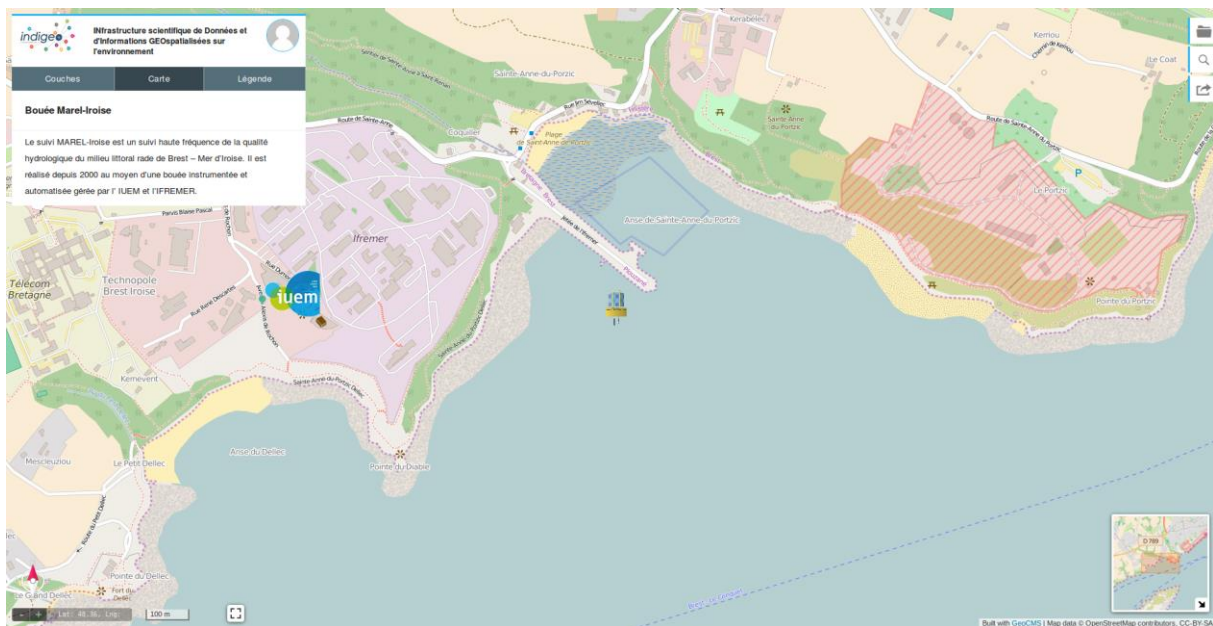
- Accroître la connaissance du monde marin,
- Étudier et observer les interactions de ce monde marin avec l'atmosphère et les espaces continentaux,
- Former des chercheurs et des cadres dans ces domaines,
- Contribuer à l'observation des modifications (naturelles ou non) de ce milieu.

Description des applications

Indigéo

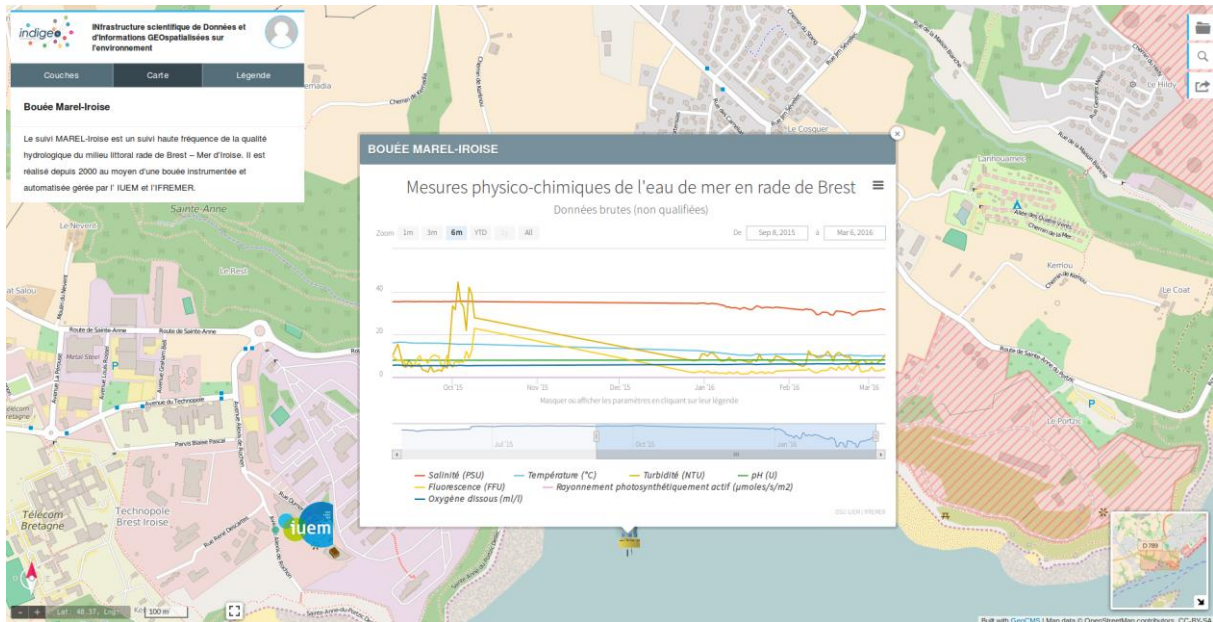
Indigéo est une infrastructure de données géographique dédiée à la recherche et à l'observation scientifique de l'environnement dans l'Ouest de la France.

C'est une application web de cartographie basée sur l'infrastructure de données spatiales « [geOrchestra](#) » répondant à la [directive INSPIRE](#) du parlement européen et du conseil de l'union européenne du 14 Mars 2007 visant à établir une infrastructure d'information géographique dans la communauté européenne. Elle est constituée de plusieurs briques logicielles libres et open-source : un catalogue de métadonnées « [GeoNetwork](#) », un serveur de données géo-référencées « [GeoServer](#) » ainsi qu'un Système de gestion de contenu géo-spatiales « [GeoCMS](#) »



Bouée instrumentée et automatisée servant au suivi haute fréquence de la qualité hydrologique du milieu littoral rade de Brest – Mer d'Iroise

Il est possible, lors de l'édition d'une carte, d'associer à un objet géographique interrogeable l'[URL](#) d'une représentation graphique.



Dans cet exemple, en cliquant sur l'icône représentant la bouée, on affiche la représentation graphique qui lui est associé.

ManageChart

ManageChart se décompose en quatre parties : Gestion des utilisateurs, des bases de données, des requêtes et des graphiques. Chacune étant dépendantes de la précédente. Ainsi, dans l'optique de générer une représentation graphique, il sera indispensable :

- D'être un utilisateur connu avec les droits d'accès correspondant,
- De créer, si elle n'existe pas, une connexion à une base de données,
- De créer, si elle n'existe pas, une requête permettant d'exploiter les données présente en base,
- De créer une représentation graphique à partir des données retournées par la requête.

Il existe trois types d'utilisateurs : l'invité, le scientifique et l'administrateur. Les différences entre ces trois types d'utilisateurs sont détaillées au chapitre « [Diagramme de cas d'utilisations](#) ».

Analyse du cahier des charges

- Mise à jour des librairies et dépendances,
- Correction de bogues,
- Amélioration de l'interface utilisateur,
- Correction des traductions,
- Ajout de nouvelles fonctionnalités.

Analyse de l'existant

Les données

Dictionnaire des données

Le dictionnaire des données reprend, pour chaque entité, la liste de ses attributs. Les entités sont générées en base de données par [Doctrine](#), qui est l'[ORM](#) par défaut utilisé par [Symfony2](#), grâce à la commande [PHP](#) : « doctrine:schema:update ».

Entité account :

id	auto_increment
username	varchar (255)
username_canonical	varchar (255)
email	varchar (255)
email_canonical	varchar (255)
enabled	bool
salt	varchar (255)
password	varchar (255)
last_login	timestamp (22)
locked	bool
expired	bool
expires_at	timestamp (22)
confirmation_token	varchar (255)
password_requested_at	timestamp (22)
roles	text
credentials_expired	bool
credentials_expire_at	timestamp (22)

Entité chart

id	auto_increment
namechart	varchar (255)
subtitlechart	varchar (255)
legendchart	bool
creditschart	text
xaxis_title	varchar (255)
xaxis_unit	varchar (255)
xaxis_type	varchar (255)
exportprintchart	bool
exportcsvchart	bool
typechart	text
gapsizechart	int
titlechart	varchar (255)
urlcreditschart	text
invertedchart	bool

Entité datasource

id	auto_increment
namebdd	varchar (255)
descriptionbdd	text
hostbdd	varchar (255)
portbdd	int
loginbdd	varchar (255)
passwordbdd	varchar (255)
typebdd	int
typestrbdd	varchar (255)
datebdd	timestamp (22)

Entité attributspatial

id	auto_increment
nameattribut	varchar (255)
valueattribut	varchar (255)
typeattribut	varchar (255)
keywordattribut	varchar (255)

Entité yaxis

id	auto_increment
titleyaxis	varchar (255)
typeyaxis	varchar (255)

Entité series

id	auto_increment
titleserie	varchar (255)
unitserie	varchar (255)
typeserie	varchar (255)
colorserie	varchar (255)
markerserie	bool
dashstyleserie	varchar (255)
parameterdatalist	varchar (255)

Entité datalist

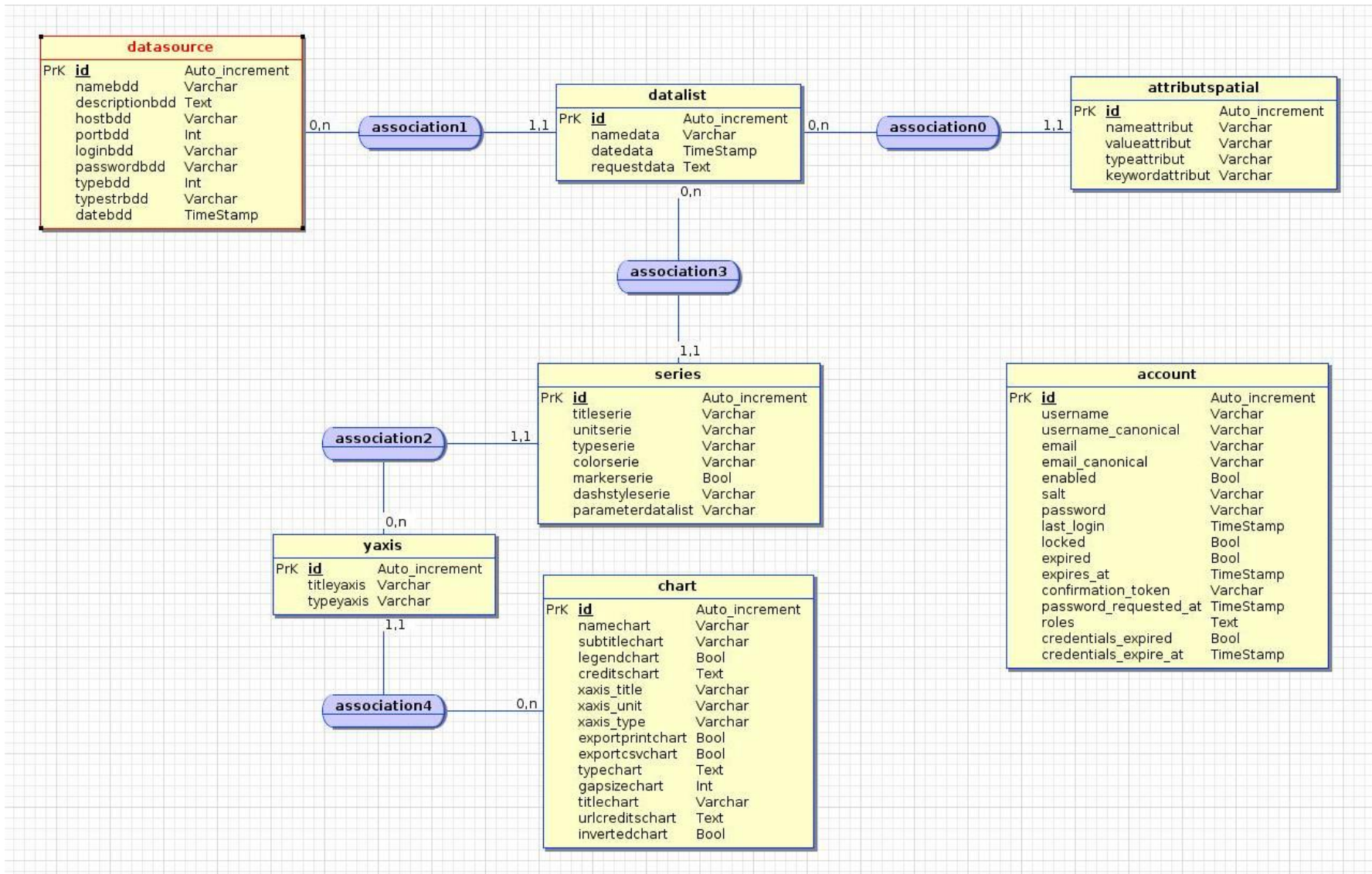
id	auto_increment
namedata	varchar (255)
datedata	timestamp (22)
requestdata	text

Règles de gestion

L'entité « account » servant à la gestion des utilisateurs, étant géré par [FOSUserBundle](#), n'a aucune interaction avec le reste de l'application.

datalist	0,n	→	association0	←	1,1	attributspacial
datasource	0,n	→	association1	←	1,1	datalist
yaxis	0,n	→	association2	←	1,1	series
datalist	0,n	→	association3	←	1,1	series
chart	0,n	→	association4	←	1,1	yaxis

Modèle conceptuel des données



Contraintes d'intégrité

Les contraintes d'intégrité sont représentées par les restrictions établies aux données. Elles sont principalement définies par [les cardinalités 0,1 et 1,1](#).

Dans notre cas, [les règles de gestion](#) montrent que :

- Un « attributspatial » ne peut être associé qu'à une seule « datalist »,
- Une « datalist » ne peut être associée qu'à une seule « datasource »,
- Une « series » ne peut être associée qu'à un seul « yaxis » ainsi qu'à une seule « datalist »,
- Un « yaxis » ne peut être associé qu'à un seul « chart ».

Les contraintes d'intégrité sont également définies les contraintes « UNIQUE » et « NOT NULL » dont :

- « UNIQUE » ne représente que les champs « id » qui sont auto-incrémentés,
- « NOT NULL » s'applique à tous les champs qui ne peuvent être vides. Ici, seuls les champs « subtitlechart », « xaxis_title », « xaxis_unit » de l'entité « chart » ainsi que le champ « unitserie » de l'entité « series » peuvent être vides. Tous les autres attributs sont donc concernés par cette contrainte.

Dépendances fonctionnelles

Les dépendances fonctionnelles permettent de déterminer pour chaque entité, un attribut source. Pour chaque valeur de cet attribut source, on identifie les attributs déterminés.

Entité account : { id } ↓

{ username, username_canonical, email, email_canonical, enabled, salt, password, last_login, locked, expired, expires_at, confirmation_token, password_requested_at, roles, credentials_expired, credentials_expire_at }

Entité chart : { id } ↓

{ namechart, subtitlechart, legendchart, creditschart, xaxis_title, xaxis_unit, xaxis_type, exportprintchart, exportcsvchart, typechart, gapsizechart, titlechart, urlcreditschart, invertedchart }

Entité datasource : { id } ↓

{ namebdd, descriptionbdd, hostbdd, portbdd, loginbdd, passwordbdd, typebdd, typestrbdd, datebdd }

Entité series : { id } ↓

{ titleserie, unitserie, typeserie, colorserie, markerserie, dashstyleserie, parameterdatalist }

Entité attributspatial : { id } ↓

{ nameattribut, valueattribut, typeattribut, keywordattribut }

Entité datalist : { id } ↓

{ namedata, datedata, requestdata }

Entité yaxis : { id } ↓

{ titleyaxis, typeyaxis }

Utilisateurs et droits

La gestion des utilisateurs est effectuée par le bundle « FOSUserBundle » qui intègre tout un système de gestion d'utilisateur. Après une configuration minimale, il peut être directement utilisable. Cependant, il est tout à fait possible de surcharger les formulaires, les vues, les traductions...

Ce bundle permet l'inscription de nouveaux utilisateurs (avec la possibilité de générer un email de confirmation), l'édition et la suppression d'utilisateurs ainsi que le changement de mot de passe. Ici, le formulaire d'enregistrement a été volontairement attribué aux administrateurs, ce qui signifie qu'un nouvel utilisateur doit forcément demander à un administrateur de lui créer un compte.

La gestion des droits est défini dans [Symfony2](#) dans un fichier décrivant tous les paramètres de sécurité de l'application. C'est donc dans ce fichier que l'on définit les rôles et où l'on définit les droits d'accès pour chaque rôle. Dans l'application « ManageChart » sont définis les rôles : « administrateur » et « scientifique ». Une route est ensuite associée à chaque rôle.

















Les vues

L'utilisateur « invité » peut visualiser la liste des graphiques et récupérer les [URL](#).

Graphiques

id	Nom	Type	URL pour l'iframe
196	groupe travail	Classique	http://localhost/ManageChar
173	test2	Classique	http://localhost/ManageChar
175	test4	Classique	http://localhost/ManageChar
176	test5	Classique	http://localhost/ManageChar
177	test12	Temporel	http://localhost/ManageChar
178	test6	Classique	http://localhost/ManageChar
179	test7	Classique	http://localhost/ManageChar
172	test1	Classique	http://localhost/ManageChar
206	testExportcsv	Classique	http://localhost/ManageChar
209	testurl	Classique	http://localhost/ManageChar
211	testparamimage	Classique	http://localhost/ManageChar
212	testmultiaxeY	Temporel	http://localhost/ManageChar
171	test11	Temporel	http://localhost/ManageChar
219	testmultiaxeY2	Temporel	http://localhost/ManageChar
220	testmultiaxeY3	Temporel	http://localhost/ManageChar

































Graphiques

Actions	id	Nom	Type	URL pour l'iframe
	196	groupe travail	Classique	http://localhost/ManageChar
	173	test2	Classique	http://localhost/ManageChar
	175	test4	Classique	http://localhost/ManageChar
	176	test5	Classique	http://localhost/ManageChar
	177	test12	Temporel	http://localhost/ManageChar
	178	test6	Classique	http://localhost/ManageChar
	179	test7	Classique	http://localhost/ManageChar
	172	test1	Classique	http://localhost/ManageChar
	206	testExportcsv	Classique	http://localhost/ManageChar
	209	testurl	Classique	http://localhost/ManageChar
	211	testparamimage	Classique	http://localhost/ManageChar
	212	testmultiaxeY	Temporel	http://localhost/ManageChar
	171	test11	Temporel	http://localhost/ManageChar
	219	testmultiaxeY2	Temporel	http://localhost/ManageChar
	220	testmultiaxeY3	Temporel	http://localhost/ManageChar
	221	testCategory	Classique	http://localhost/ManageChar

[Nouveau graphique](#)[Nouveau graphique temporel](#)

L'utilisateur avec le rôle « scientifique » accède à la gestion des graphiques. Il peut les visualiser, les modifier ou en créer de nouveaux. Cependant, il ne peut pas en supprimer. De plus, il peut éditer son profile utilisateur.

Graphiques

Actions	id	Nom	Type	URL pour l'iframe
 	196	groupe travail	Classique	http://localhost/ManageChar
 	173	test2	Classique	http://localhost/ManageChar
 	175	test4	Classique	http://localhost/ManageChar
 	176	test5	Classique	http://localhost/ManageChar
 	177	test12	Temporel	http://localhost/ManageChar
 	178	test6	Classique	http://localhost/ManageChar
 	179	test7	Classique	http://localhost/ManageChar
 	172	test1	Classique	http://localhost/ManageChar
 	206	testExportcsv	Classique	http://localhost/ManageChar
 	209	testurl	Classique	http://localhost/ManageChar
 	211	testparamimage	Classique	http://localhost/ManageChar
 	212	testmultiaxeY	Temporel	http://localhost/ManageChar
 	171	test11	Temporel	http://localhost/ManageChar
 	219	testmultiaxeY2	Temporel	http://localhost/ManageChar
 	220	testmultiaxeY3	Temporel	http://localhost/ManageChar
 	221	testCategory	Classique	http://localhost/ManageChar

Nouveau graphique

Nouveau graphique temporel

L'utilisateur avec le rôle « administrateur » à les droits de lecture, de création, d'édition et de suppression sur les graphiques mais également sur les connexions à des bases de données, sur les requêtes et sur les utilisateurs. Il peut éditer son profil utilisateur.

Édition d'une connexion à une base de données

Nom*	<input type="text" value="testData"/>
Description*	<input type="text" value="test d'entrée de donnée"/>
Hôte*	<input type="text" value="127.0.0.1"/>
Port*	<input type="text" value="5432"/>
Identifiant*	<input type="text" value="admin_geomer"/>
Mot de passe*	<input type="password"/>
Type*	<input type="text" value="PostgresSQL"/>

Exemple d'édition d'une connexion à une base de données. Ce formulaire concerne la création et l'édition d'une connexion à une base de données accessible par un « administrateur ».

Édition requête SQL

Nom*

BDD*

Requête *

```
SELECT s.name,
COUNT(e.id_metadata) as
count, e.date_extraction
FROM
geobs_harvester.extraction e,
geobs_harvester.sdi s WHERE
e.id_sdi=s.id_sdi GROUP BY
s.name, e.date_extraction
ORDER BY count DESC
```

Une requête SQL doit respecter le format suivant :

- 1er champ : x
- 2eme champ : y
- 3eme champ : nom du parametre en y
- 4eme champ : unite du parametre en y (en option)

Les valeurs peuvent être indifféremment des valeurs numériques ou des chaînes de caractères mais toutes les valeurs d'un même champ pour une série donnée doivent être du même type puisque la verification n'est faite que sur la première ligne

Pour une date la valeur doit etre un **UNIX TIMESTAMP**

- Postgres : EXTRACT(EPOCH FROM '2007-11-30 10:30:19')
- MySQL : UNIX_TIMESTAMP('2007-11-30 10:30:19')

Attention les graphiques attendent des millisecondes depuis le 01-01-1970 et non des secondes comme le renvoi ces fonctions. Il faut donc rajouter un facteur 1000

[JavaScript Date class](#)

Ajouter un attribut spatial

Envoyer

Insérer les attributs spatiaux dans la requête

Tester la requête sans traitement des données

Tester la requête avec traitement des données

Exemple d'édition d'une requête. Ce formulaire concerne la création et l'édition d'une requête. Il est indispensable de sélectionner une connexion à une base de données. Il est également possible d'ajouter dynamiquement la clause « WHERE » de la requête, un formulaire imbriqué apparaissant lors du clic sur « Ajouter un attribut spatial ». Formulaire accessible par un « administrateur ».

Nouveau graphique

Nom*

Titre

Sous-titre

Titre axe X

Unité axe X

Type axe X

*

Crédits *

Url Crédits

Légende

Export impression

Export CSV

Inverser les axes

Ajouter un axe Y

Envoyer

Exemple d'édition d'un graphique. Ce formulaire concerne la création et l'édition d'un graphique. On définit ici les caractéristiques et options générale du graphique. Il est ensuite possible, d'ajouter un axe Y au graphique en cliquant sur le bouton « Ajouter un axe Y » faisant ainsi apparaître un formulaire imbriqué. (Voir ci-dessous). Ce formulaire est accessible à un « scientifique ».

Axe Y n°1 *

Supprimer l'axe Y

Titre

Type *

linéaire

Ajouter une série

Ajouter toutes les séries d'une requête

Choisissez une requête

Ajouter toutes les series

Ajouter un axe Y

Envoyer

Ce formulaire imbriqué permet l'ajout d'un axe Y au graphique. Il est possible d'ajouter autant d'axe Y que souhaité. Ce formulaire imbriqué est différent dans le cas d'un graphique temporel. En effet, celui-ci ne peut avoir qu'un seul axe Y et son type est déjà connu. De ce fait, seul le titre de l'axe Y est éditable. Dans tous les cas, le bouton « Ajouter une série » apparait, qui au clic, affiche un second formulaire imbriqué permettant l'ajout d'une série de données à notre axe Y (Voir ci-dessous). Une autre possibilité consiste en l'ajout à l'axe Y de toutes les séries de données d'une requête et affiche, autant de fois qu'il y a de série de données, le second formulaire imbriqué avec les champs pré-rempli.

Série n°1 *	Titre *	<input type="text"/>	Ajouter une série
Supprimer la série			Ajouter toutes les séries d'une requête
	Unité	<input type="text"/>	
	Requête *	<input type="text" value="Choisissez une requête"/>	<input type="text" value="Choisissez une requête"/>
	Paramètre *	<input type="text"/>	Ajouter toutes les series
	Type *	<input type="text" value="ligne"/>	
	Couleur *	<input type="text" value="#3399CC"/>	
	Marqueur *	<input type="checkbox"/>	
	Style de ligne *	<input type="text" value="Ligne"/>	

Exemple pour l'ajout d'une série de données à un axe Y. La sélection d'une requête complète le champ « paramètre » en lui donnant pour options les séries de données. Il est ensuite possible d'éditer les options de la série.

Édition du profile

Nom d'utilisateur :
*

Adresse e-mail :*

Mot de passe actuel :
*

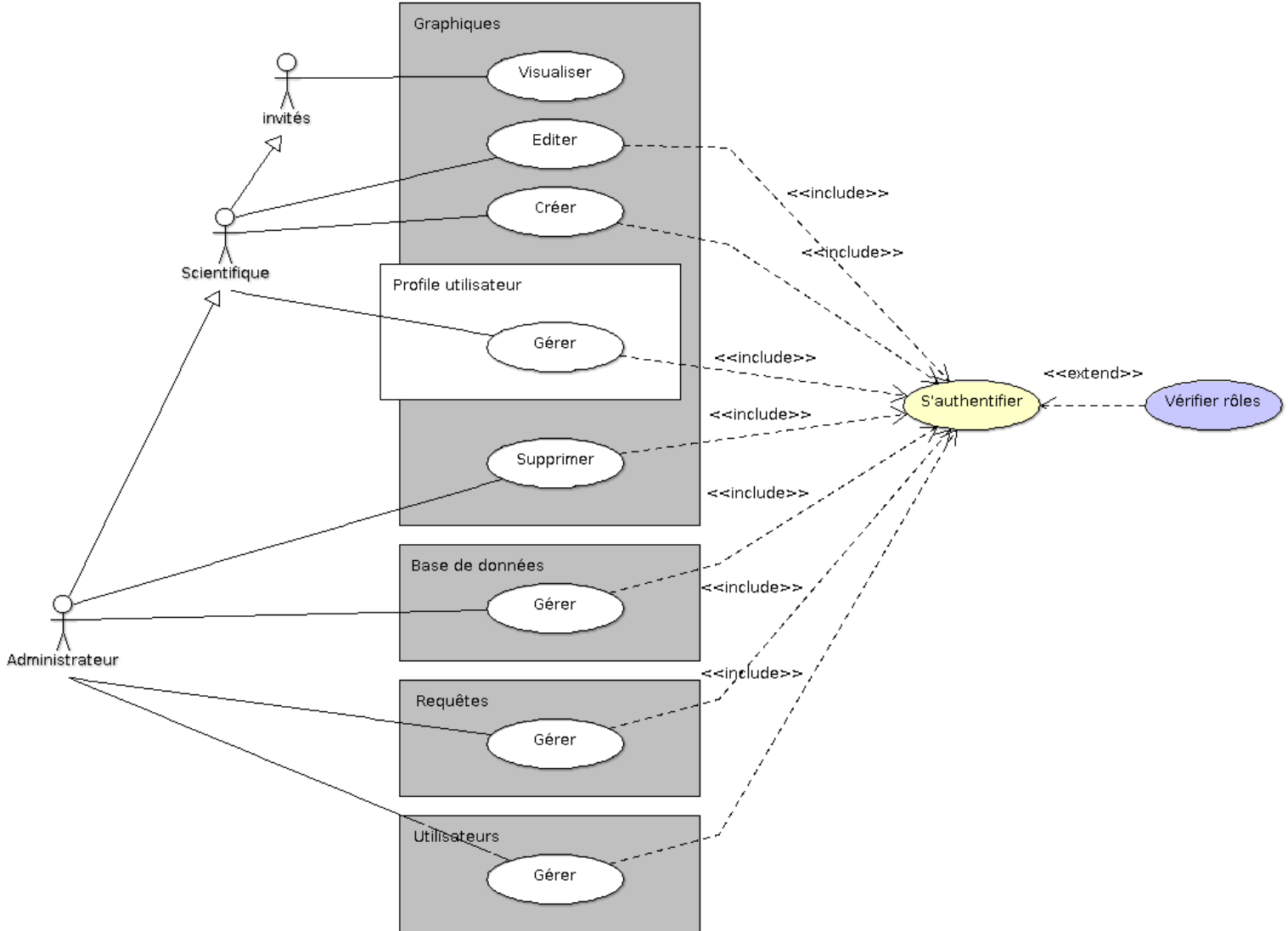
Roles*

Envoyer

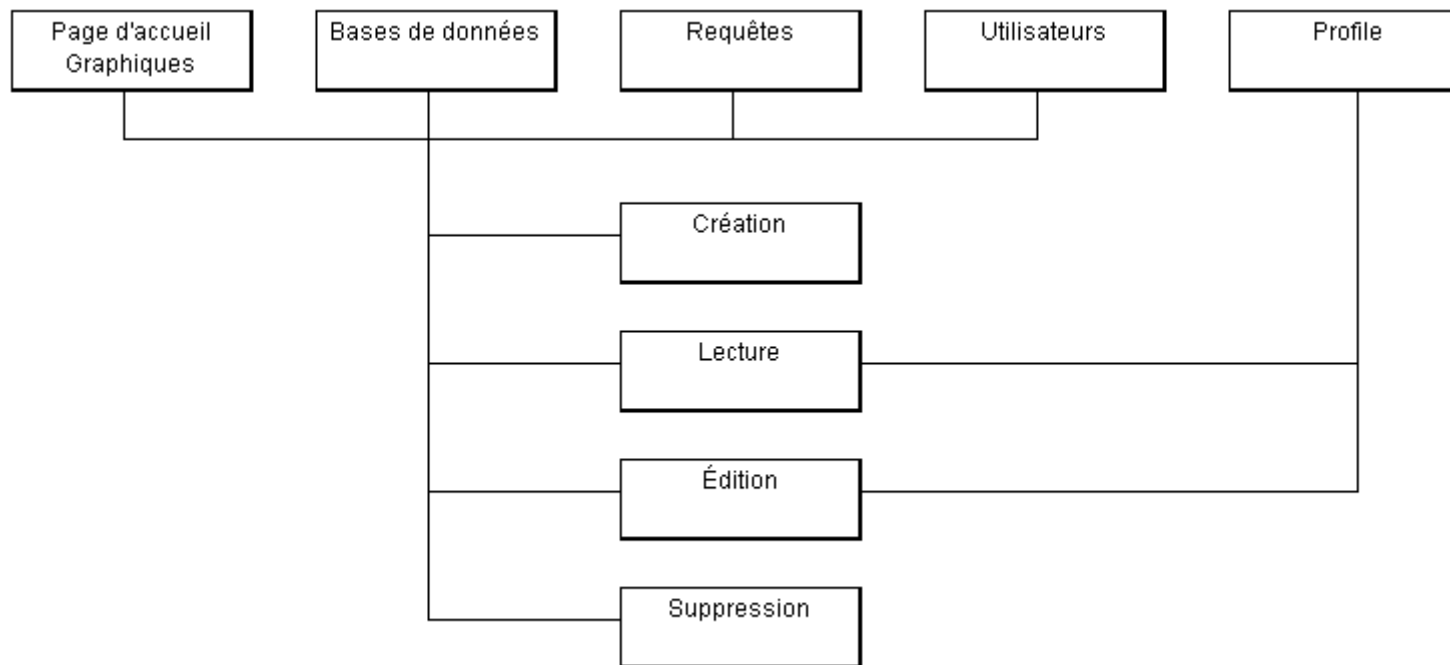
Changer le mot de passe

Exemple d'édition de profile utilisateurs. Ce formulaire est accessible par un « scientifique ». Cependant, celui-ci ne pourra pas modifier le rôle qui lui a été attribué.

Diagrammes de cas d'utilisations



Plan de navigation

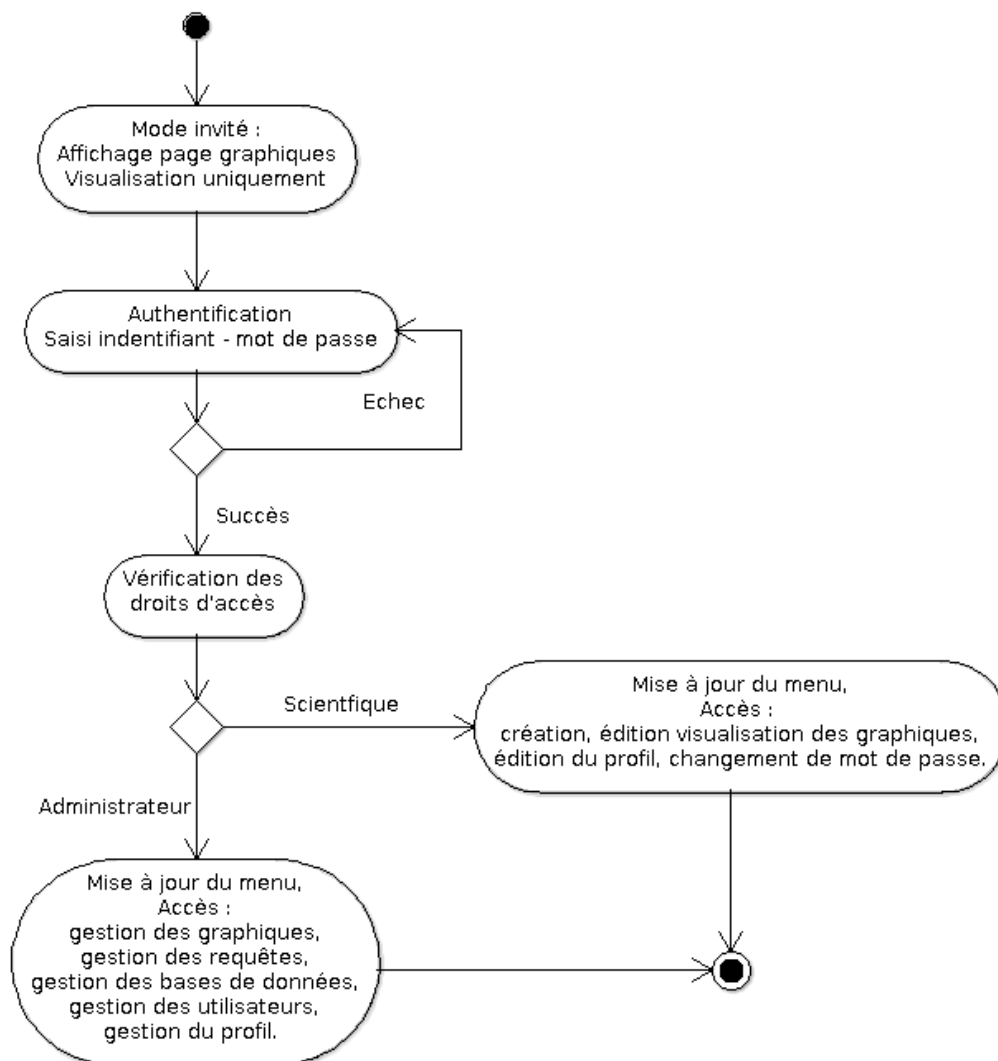


Ce plan de navigation concerne uniquement le rôle « administrateur »

L'utilisateur « invité » n'ayant pas de menu, il n'a accès qu'à la page d'accueil. Il peut, de ce fait, uniquement visualiser les graphiques.

L'utilisateur « scientifique » peut se connecter à l'application et ainsi gérer son profil utilisateurs mais ne peut modifier son rôle utilisateur. Il accède à la page des graphiques et peut les visualiser, les éditer ou en créer de nouveau. Mais il ne peut, en aucun cas, en supprimer.

Diagramme d'activités de l'authentification



Technologies utilisées

[PHP5](#) et [JavaScript](#) associé au [HTML5](#) et [CSS3](#) ont été retenus pour concevoir l'application « ManageChart ». Afin d'effectuer un suivi, le logiciel de gestion de versions Git a également été sélectionné.

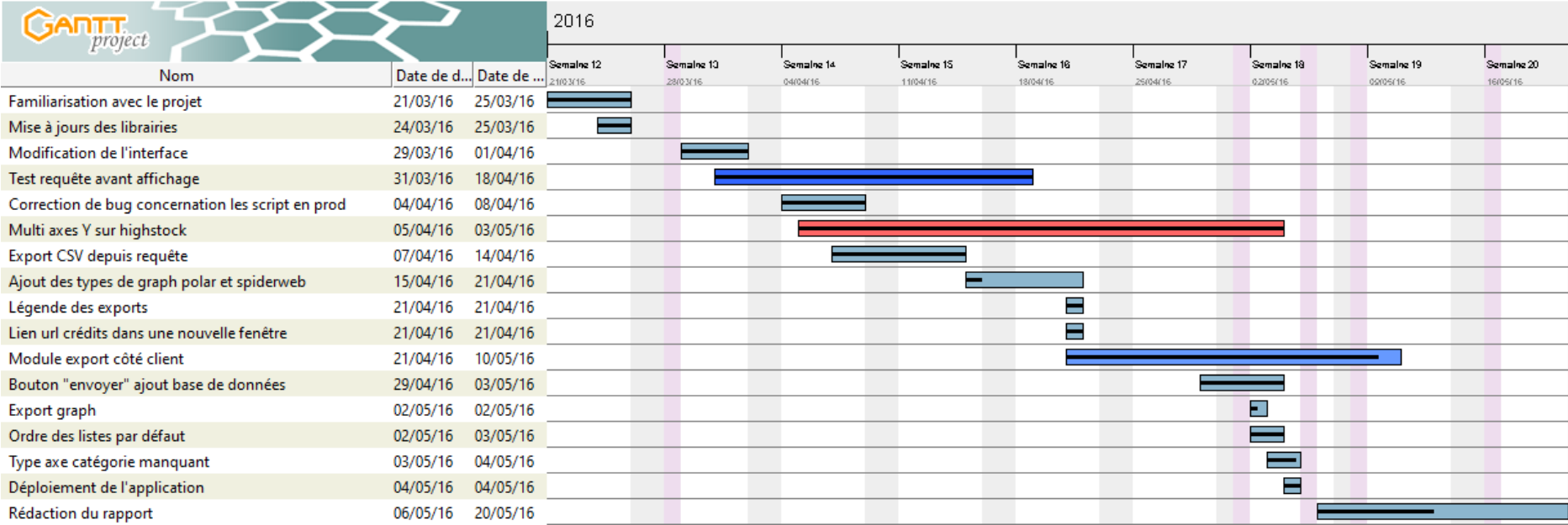
Framework et librairies

Afin de respecter le pattern [MVC](#) et de faciliter la maintenabilité de l'application, le Framework [Symfony2](#) a été choisi. Dans le même optique et afin d'accroître la rapidité de programmation, les librairies jQuery pour apporter le dynamisme et [Bootstrap](#) pour l'apparence générale de l'application ainsi que pour l'aspect « [Responsive](#) » de l'application.

Le but de l'application « ManageChart » étant de fournir des représentations graphiques, l'ensemble de librairies « [HighCharts](#) », permettant la création de graphiques interactifs, a été sélectionné car elles offrent une grande variété de graphiques et d'options d'exports. Elle se décline en quatre librairies :

- « highcharts » permettant la génération de graphiques,
- « higstocks » permettant la génération de graphiques spécifiquement temporel,
- « exporting » permettant l'ajout d'options d'export au format image et PDF, ainsi que l'impression papier,
- « export-csv » permettant l'ajout d'options d'export au format [CSV](#), XLS, ainsi que la visualisation dans le navigateur. Celle-ci dépend de la précédente.

Planification



Projet de stage

Actions demandées

Ajout de fonctionnalités :

- Ajout du type d'axes X « Catégorie »,
- Déploiement d'un serveur d'export « [HighCharts](#) » côté client,
- Ajout d'un système de pagination sur les résultats retournés par les requêtes,
- Ajout d'une fonctionnalité permettant l'export au format [CSV](#) des résultats retourné par les requêtes,
- Ajout d'une option de tri sur les tableaux listant avec, par défaut, un tri par « id » décroissant,
- Mise en place d'une solution permettant la gestion de graphiques avec de multiples axes Y pour les graphiques temporels,
- Ajout de YUI Compressor pour la compression des scripts et fichiers de style.

Modifications :

- Amélioration de la gestion des messages d'erreurs retourné par les requêtes ainsi que pour les formulaires de gestions d'utilisateurs,
- Amélioration générale de l'interface et ajout de méthode [Responsive](#) sur les éléments non pris en compte par [Bootstrap](#),
- Ajout d'un champ « Nom de connexion » dans les formulaires d'enregistrement et d'édition de connexion à une base de données, permettant une identification plus facile des connexions dans les formulaires d'enregistrement et d'édition de requêtes.
- Modification sur les type de série de données « Nuage de point » et « Colonne horizontale »

Correction de bogues :

- Correction sur l'export au format image d'un graphique, les noms des axes se chevauchant les uns les autres,
- Correction sur la façon dont s'ouvre l'[URL](#) du crédit, celui-ci s'ouvrant dans l'onglet courant,
- Suppression du « navigator » (Barre de défilement des graphiques temporels) lors des exports aux formats [CSV](#), XLS ou en visualisation dans le navigateur,
- Correction sur le déploiement de la version de production de l'application, les scripts et fichiers de style sont générés mais introuvables par l'application.

Modifications apportées aux données

L'application utilisant des bases de données « [PostgreSQL](#) » mais aussi « [MySQL](#) », il peut y avoir, dans « ManageChart », plusieurs bases de données portant le même nom. Il est également possible d'avoir des bases de données de même nom et de même type si elles ne sont pas sur le même serveur. Pour remédier à cela, un champ « Nom de connexion » a été ajouté aux formulaires d'enregistrement et d'édition d'une connexion à une base de données. Celui-ci est complété par l'utilisateur « administrateur » permet une identification plus aisée de la connexion pour son utilisation lors de l'édition d'une requête.

L'attribut « namecon » a été ajouté à l'entité « datasource ».

Entité datasource

id	auto_increment
namebdd	varchar (255)
descriptionbdd	text
hostbdd	varchar (255)
portbdd	int
loginbdd	varchar (255)
passwordbdd	varchar (255)
typebdd	int
typestrbdd	varchar (255)
datebdd	timestamp (22)
Namecon	varchar (255)

































Ce « [Nom de connexion](#) » est ensuite récupéré dans les formulaires d'enregistrement et d'édition de requête et est concaténé avec le nom de la base de données et le type de la base de données. Ainsi, la connexion a une base de données est facilement identifiable.

Modifications apportées aux vues

ManageChart Bases de données Requêtes SQL Graphiques Utilisateurs admin

Graphiques

Nouveau graphique Nouveau graphique temporel

Actions	id	Nom	Type	URL pour l'iframe
 	221	testCategory	Classique	<input type="text" value="http://localhost/ManageC"/>
 	220	testmultiaxeY3	Temporel	<input type="text" value="http://localhost/ManageC"/>
 	219	testmultiaxeY2	Temporel	<input type="text" value="http://localhost/ManageC"/>
 	212	testmultiaxeY	Temporel	<input type="text" value="http://localhost/ManageC"/>
 	211	testparamimage	Classique	<input type="text" value="http://localhost/ManageC"/>
 	209	testurl	Classique	<input type="text" value="http://localhost/ManageC"/>
 	206	testExportcsv	Classique	<input type="text" value="http://localhost/ManageC"/>
 	196	groupe travail	Classique	<input type="text" value="http://localhost/ManageC"/>
 	179	test7	Classique	<input type="text" value="http://localhost/ManageC"/>
 	178	test6	Classique	<input type="text" value="http://localhost/ManageC"/>
 	177	test12	Temporel	<input type="text" value="http://localhost/ManageC"/>
 	176	test5	Classique	<input type="text" value="http://localhost/ManageC"/>
 	175	test4	Classique	<input type="text" value="http://localhost/ManageC"/>
 	173	test2	Classique	<input type="text" value="http://localhost/ManageC"/>
 	172	test1	Classique	<input type="text" value="http://localhost/ManageC"/>
 	171	test11	Temporel	<input type="text" value="http://localhost/ManageC"/>

Ici, la page d'accueil vue par un « administrateur ». Le menu a été modifié et la partie concernant le profile utilisateur a été déplacé à droite. Sur chaque page, les boutons de création ont été déplacés vers le haut de page pour être toujours visible. Les boutons de suppression et d'éditations ont également été modifié sur toutes les pages de manière à avoir les mêmes dimensions et le même comportement au redimensionnement de la page. On voit également ici la [fonction de tri](#) apporté aux tableaux listant.

Nouvelle connexion à une base de données

Nom de connexion	<input type="text"/>	*
Nom Bdd	<input type="text"/>	*
Type	<input type="text" value="PostgreSQL"/>	*
Description	<input type="text"/>	*
Hôte	<input type="text"/>	*
Port	<input type="text"/>	*
Identifiant	<input type="text"/>	*
Mot de passe	<input type="password"/>	*
	<input type="button" value="Enregistrer"/>	<input type="button" value="Test connexion"/>

Le formulaire d'enregistrement et d'édition d'une connexion à une base de données, accessible pour un « administrateur ». Le formulaire a été modifié afin de correspondre à la dernière version de [Bootstrap](#). Les boutons ont été alignés. Le champ « Nom de connexion » fait son apparition et une méthode a été implémenté afin de gérer les messages d'erreurs sur le test de connexion.

Nouvelle requête SQL

Nom

BDD

Requête

Une requête SQL doit respecter le format suivant :

- 1er champ : x
- 2eme champ : y
- 3eme champ : nom du parametre en y
- 4eme champ : unite du parametre en y (en option)

Les valeurs peuvent être indifféremment des valeurs numériques ou des chaînes de caractères mais toutes les valeurs d'un même champ pour une série donnée doivent être du même type puisque la verification n'est faite que sur la première ligne

Pour une date la valeur doit être un [UNIX TIMESTAMP](#)

- Postgres : `EXTRACT(EPOCH FROM '2007-11-30 10:30:19')`
- MySQL : `UNIX_TIMESTAMP('2007-11-30 10:30:19')`

Attention les graphiques attendent des millisecondes depuis le 01-01-1970 et non des secondes comme le renvoi ces fonctions. Il faut donc rajouter un facteur 1000

[JavaScript Date class](#)

Enregistrer

Ajouter un attribut spatial

Test sans traitement

Test avec traitement

Le formulaire d'enregistrement et d'édition des requête, accessible pour un « administrateur ». L'apparence a été modifié pour correspondre à la dernière version de « [Bootstrap](#) » et les boutons ont été alignés. On peut voir que dorénavant le champ « Bdd » correspond à la concaténation du nom de connexion, du nom de la base de données et du type de la base de données. Un bouton permettant l'agrandissement du champ « Requête » a été ajouté afin que la mise en forme ne soit pas perturbé par la modification de la taille de ce champ.

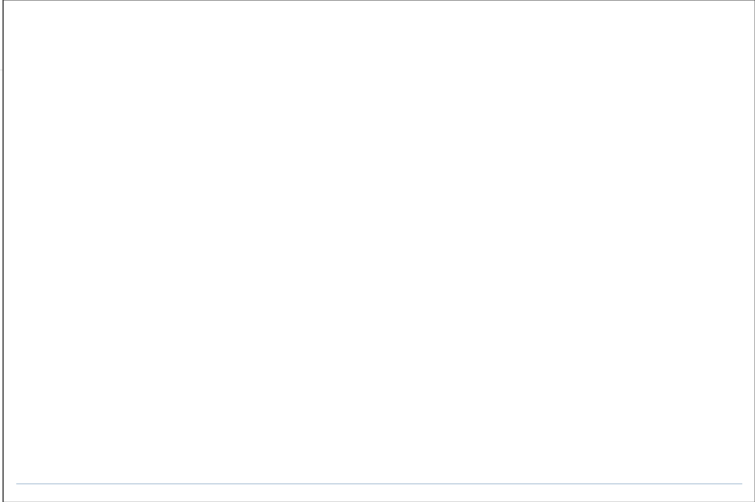
ManageChart Bases de données Requêtes SQL Graphiques Utilisateurs admin

Masquer le graphique Fixez moi

Nouveau graphique

Nom	<input type="text"/>	Crédits	<input type="text"/>
Titre	<input type="text"/>	Url Crédits	<input type="text"/>
Sous-titre	<input type="text"/>	Légende	<input checked="" type="checkbox"/>
Titre axe X	<input type="text"/>	Export Impression	<input type="checkbox"/>
Unité axe X	<input type="text"/>	Export CSV	<input type="checkbox"/>
Type axe X	<input type="text" value="linéaire"/>	Inverser les axes	<input type="checkbox"/>

Ajouter un axe Y Enregistrer



Le formulaire d'enregistrement et d'édition d'un graphique, accessible pour un « scientifique ». De même que pour les pages précédentes, l'apparence a été modifiée pour correspondre à la dernière version de « [Bootstrap](#) » et les boutons ont été alignés. L'[URL](#) du crédit s'ouvre à présent dans un nouvel onglet du navigateur. La sélection de l'export CSV entraîne la sélection dynamique de l'export pour impression dont il dépend. Le type d'axes X « Catégorie » a été ajouté. De plus, les champs « légende » et « inverser les axes » ont été améliorés pour que les modifications soient visibles dans la fenêtre de visualisation. L'appui sur le bouton « Ajouter un axe Y » fait apparaître le formulaire ci-dessous.

Axe Y n°1

Titre

Supprimer l'axe Y

Type *

Ajouter toutes les séries d'une requête

Ajouter toutes les series

Ajouter une série

Le formulaire imbriqué d'ajout d'un axe Y au graphique. Ce formulaire est appelé autant de fois que l'utilisateur souhaite ajouter d'axes. Ce formulaire implémente, à présent, également les graphiques temporels. Tous les champs ont été inversé, ainsi le bouton d'ajout d'un axe Y n'est plus déplacé vers le bas, chaque nouveau formulaire d'ajout d'un axe Y est ajouté au-dessus du précédent. Le bouton d'ajout d'une série de données a été placé en bas de formulaire et l'appui sur ce bouton fait apparaître le second formulaire imbriqué ci-dessous.

Série n°1 * Supprimer la série

Titre *	<input type="text"/>	Unité	<input type="text"/>
Requête *	<input type="text" value="Choisissez une requête"/>	Paramètre *	<input type="text"/>
Type *	<input type="text" value="ligne"/>	Couleur *	<input type="text" value="#3399CC"/>
Marqueur *	<input type="checkbox"/>	Style de ligne *	<input type="text" value="Ligne"/>

Le formulaire d'ajout d'une série de données à un axe Y. L'utilisateur peut ajouter pour chaque axe Y autant de séries de données qu'il le souhaite. Tout comme le formulaire imbriqué d'ajout d'axes Y, celui-ci a été remanié pour que le bouton d'ajout d'une série de données ne soit plus déplacé vers le bas et que chaque nouvelle série de données soit ajoutée au-dessus de la précédente. Une correction a été apportée sur les types de série « nuage de point » et « colonne horizontale » afin qu'ils soient visibles dans la fenêtre de visualisation.

Fonctionnalités ajoutées

Ajout du type d'axes X « Catégorie »

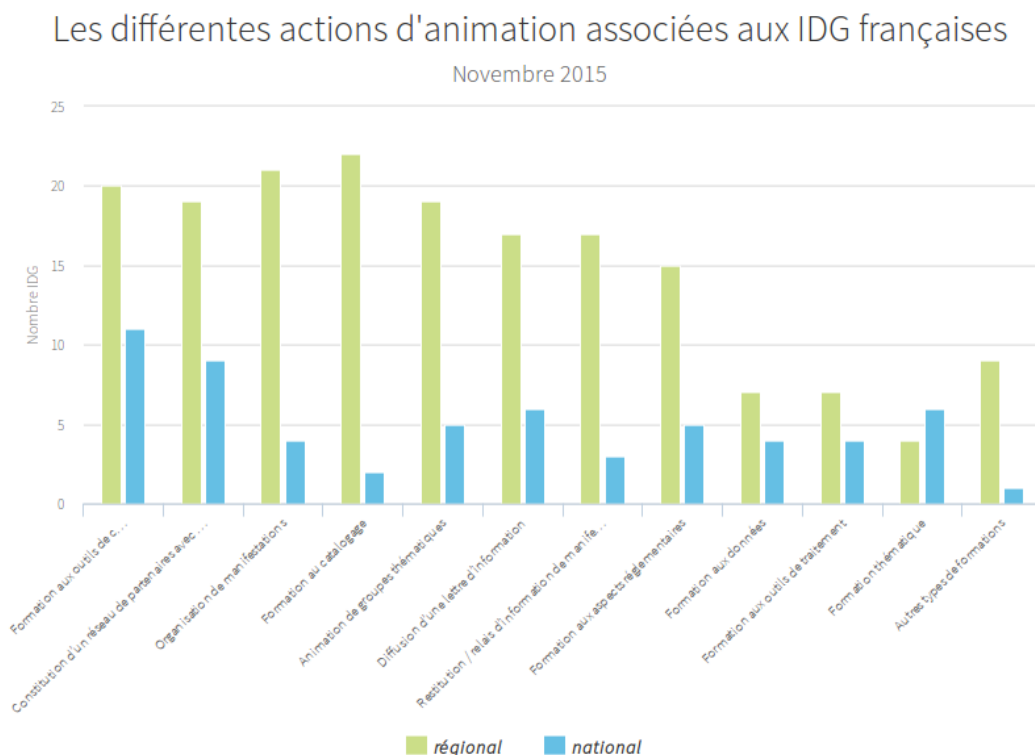
Lors du déploiement d'une version plus récente de l'application, ce type d'axe X a été supprimé puisqu'il avait été ajouté directement dans la version de production. Les type d'axes X sont défini par la librairie « [HighCharts](#) ».

Dans le but de faciliter la maintenabilité de l'application, une class [PHP](#) « AvailableTypeAxis » a été créé et dont le seul but est de répertorier dans un array les différents types d'axes X.

```
class AvailableTypeAxis
{
    public static $typesAxis = array(
        'linear' => 'lineaire',
        'datetime' => 'temporel',
        'logarithmic' => 'logarithmique',
        'category' => 'categorie'
    );
}
```

Cet array est ensuite récupéré par le Controller et injecté dans le champ de sélection du type d'axe X. L'action de l'utilisateur sur ce champ entraîne dynamiquement la modification du graphique.

Ainsi, pour chaque nouveau type d'axes X que nous souhaiterions ajouter, il suffira de l'ajouter à la suite dans la classe [PHP](#) « AvailableTypeAxis » et de s'assurer que la version de la librairie « [HighChart](#) » utilisé, prenne en compte ce type d'axe X.



Déploiement d'un serveur d'export « [HighCharts](#) » côté client

Lors de l'export d'un graphique, celui-ci est, par défaut, envoyé au serveur d'export « [highchart.com](#) » qui traite la demande et retourne le fichier exporté. Afin que les données ne transitent plus par internet, « Highsoft » propose la solution de déployer un serveur d'export côté client.

En suivant les instructions disponibles sur le site d'[HighCharts](#) et de ce [tutoriel](#), les étapes suivantes ont permis le déploiement du serveur d'export :

1. Installation de [JAVA](#) et déclaration de la variable d'environnement JAVA_HOME,
2. Installation de [Tomcat](#) et déclaration de la variable d'environnement CATALINA_HOME,
3. Installation de [Maven](#),
4. Installation de [PhantomJS](#),
5. Téléchargement de « highchart-export-server » et le décompresser,
6. Se placer à l'intérieur du dossier « highcharts-export » et lancer la compilation avec [Maven](#),
7. Se placer à l'intérieur du dossier « highchart-export-web » et effectuer un nettoyage des paquet [Maven](#) puis relancer la compilation,
8. Déplacer le fichier compilé « highchart-export-web.war » dans le répertoire « webapps » de [Tomcat](#),
9. Démarrer [Tomcat](#).

Le serveur d'export est ensuite disponible à l'adresse :

« <http:// { IP du serveur } :8080/highcharts-export-web/> »

Enfin, dans le but d'utiliser notre serveur d'export, il est nécessaire d'ajouter au graphique l'option `exporting.url` et de lui passer en valeur l'adresse de notre serveur.

Ajout d'un système de pagination sur les résultats retournés par les requêtes

Lors de l'enregistrement ou de l'édition d'une requête, il est possible de la tester et de visualiser le résultat retourné par la requête sous forme de tableau. Il est ainsi possible de tester la requête avec traitement des données. Le traitement des données étant effectué suite à l'affichage du tableau, l'ajout d'une fonctionnalité de pagination entraînerait un traitement erroné puisqu'effectué sur le tableau visible. La pagination a, de ce fait, été implémenté uniquement sur le test de la requête sans traitement de données.

Ainsi, lorsque l'utilisateur demande un test de la requête sans traitement de données, on indique qu'il s'agit d'un test de la requête sans traitement des données, on teste le type de base de données, on indique une LIMIT par défaut de 10 et si le type de base de données est PostgreSQL, on indique un OFFSET.

Annexe [wichQuery](#)

```
if (type == "QB") {
    var url = '{{ path('data_list_query_brut_postgres', {"typebdd":
"postgres", "limit": "LIMIT", "offset": "OFFSET", "nbPage": "nbPage"}) }}';
    url = url.replace("OFFSET", "OFFSET " + offset);
    url = url.replace("LIMIT", "LIMIT " + limit);
    url = url.replace("nbPage", limit);
}
```

Cette [URL](#) est ensuite transmise au Contrôleur par une fonction jQuery [Ajax](#) et contient également un paramètre « nbPage » dont la valeur est égale à celle de la LIMIT. Le nombre de pages visible à afficher est ensuite calculé dans le Contrôleur comme suit :

```
$count = 0;
foreach ($listData as $data) {
    $count = $count + 1;
}
$nbPages=ceil($count/$nbPage);
```

Pour chaque valeur retournée par la requête, on incrémente de 1 la valeur de la variable \$count, pour obtenir en fin de boucle, le nombre total de résultats retournés par la requête. Puis on calcule le nombre de pages à afficher en faisant {nombre de résultats de la requête} / {LIMIT}.

Par exemple, si la requête retourne 76 résultats, le nombre de pages sera par défaut de 8.

Enfin, si le nombre de résultats de la requête est supérieur à la LIMIT, on ajoute la pagination à la variable que récupère la fonction jQuery [Ajax](#).

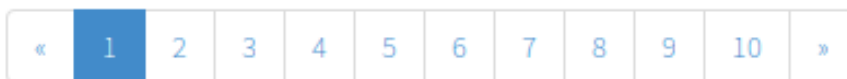
Annexe [requête avec pagination](#)

```
if ($count > $nbPage) {
    $html .= '<div class="pull-right"><nav><ul id="paging"
class="pagination"><li class="prev"><a href="#" aria-label="Previous"><span
aria-hidden="true">&laquo;</span></a></li>;
    for ($i = 1; $i <= $nbPages; $i++) {
        $html .= '<li class="page pagehead"><a href="#">' . $i .
'</a></li>;
    }
    $html .= '<li class="next"><a href="#" aria-label="Next"><span
aria-hidden="true">&raquo;</span></a></li></ul></nav></div>';
}
```

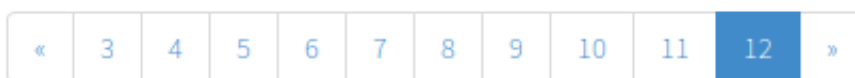
Pour chaque page, un élément de liste avec pour valeur, le numéro de pages est implémenté. Mais à ce stade, si nous avons 101 pages, nous avons également une pagination de 101 pages. Pour remédier à cela, un script a été mis en place pour afficher une pagination comprenant les dix premières pages. L'utilisateur a ensuite le choix entre la page précédente s'il ne se trouve pas sur la première page, la page suivante s'il ne se trouve pas sur la dernière page mais il peut aussi sélectionner une page en cliquant sur son numéro. L'affichage de la pagination de 10 pages évolue en fonction de la page sur laquelle se trouve l'utilisateur.

Un champ de sélection a également été ajouté permettant à l'utilisateur de spécifier une LIMIT, les options de ce champ sont une boucle de pas de 10 de la variable du nombre de résultat retourné par la requête (\$count).

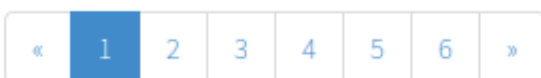
Reprenons ici l'exemple d'une requête retournant un résultat de 101 pages avec la LIMIT par défaut.



Ici, l'utilisateur se trouvant sur la première page, il peut sélectionner une page parmi les neuf suivantes ou cliquer sur le boutons affichant la page suivante qui l'emmènera sur la seconde page.



Ici, l'utilisateur se trouve sur la douzième page, la pagination est décalée pour garder un affichage de dix pages.



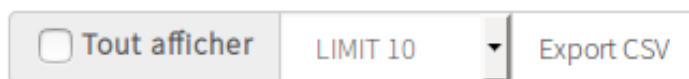
En modifiant la LIMIT, ici à 190, le nombre de page devient inférieur à 10. La pagination contient toutes les pages.

Ajout d'une fonctionnalité permettant l'export au format [CSV](#) des résultats retourné par les requêtes

Le résultat de la requête étant retourné sous forme de tableau [HTML](#), il est possible de récupérer ce tableau, de le convertir au format CSV et de permettre l'enregistrement de ce fichier.

Nous avons [vu](#) que la pagination ne pouvait se faire sur le test de la requête avec traitement puisque le traitement s'effectue sur le tableau complet. De la même façon, si nous voulons récupérer le tableau et l'enregistrer au format [CSV](#). Il nous faut donc désactiver la pagination. Ainsi, deux boutons ont été ajouté. Le premier est une case à cocher permettant de passer outre la pagination et d'afficher tous les résultats de la requête en passant une [URL](#) sans LIMIT a la fonction jQuery [Ajax](#). L'action sur cette case à cocher entraîne la suppression du champ de sélection de la LIMIT et de la pagination. Le second est un bouton d'export au format [CSV](#), l'action sur celui-ci entraîne la sélection de la précédente case à cocher dynamiquement.

L'export est ensuite exécuté par une fonction jQuery prenant en paramètre le nom du fichier à créer et le tableau contenant les données à convertir au format [CSV](#).



The screenshot shows a horizontal control bar with three elements: a checkbox labeled 'Tout afficher' which is currently unchecked, a dropdown menu showing 'LIMIT 10', and a button labeled 'Export CSV'.

Ici nous demandons l'affichage par défaut d'une requête sans traitement de donnée.



The screenshot shows the control bar after the 'Export CSV' button was clicked. The 'Tout afficher' checkbox is now checked, and the 'Export CSV' button has been replaced by a link that reads 'Télécharger Localisation Des Contributeurs.csv'.

Suite à l'action sur le bouton export [CSV](#), la case à cocher est sélectionnée, le champ de sélection de la LIMIT est supprimé ainsi que la pagination et le bouton est modifier pour devenir un lien vers le fichier CSV et dont le nom est le nom du fichier.

Le fichier exporté contient les valeurs de chaque cellule du tableau séparé par une virgule. Chaque ligne correspondant à une ligne du tableau.

```
"upper", "count", "date_extraction"|
" ", "3", "2016-01-28"
"/", "1", "2016-01-28"
"0", "1", "2016-01-28"
"13129", "1", "2016-01-28"
"13200", "1", "2016-01-28"
"33 090 BORDEAUX CEDEX", "8", "2016-01-28"
"33525 BRUGES", "6", "2016-01-28"
"33651 MARTILLAC", "1", "2016-01-28"
"40012 MONT DE MARSAN", "1", "2016-01-28"
"48001 MENDE", "1", "2016-01-28"
"48005 MENDE CEDEX", "54", "2016-01-28"
```

Ajout d'une option de tri sur les tableaux listant avec un tri par « id » décroissant

Afin de permettre le tri sur les tableaux listant les graphiques, les bases de données, les requêtes et les utilisateurs, le plug-in jquery-tablesorter a été ajouté à l'application. Il permet l'application d'option de tri avancé sur des tableaux HTML.

```
$("#tableUser").tablesorter({  
    headers: { 0: { sorter: false }},  
    sortList: [[1,1]]  
});
```

On applique la méthode « tablesorter » au tableau listant les utilisateurs en précisant que l'en-tête 0 qui correspond à la colonne du tableau dans laquelle se trouvent les boutons d'éditations et de suppressions, ne bénéficie pas de cette option de tri. On lui assigne un tri par défaut grâce à l'option « sortList » dont la première valeur correspond à l'en-tête (1 correspondant à la colonne du tableau contenant les id) et dont la seconde valeur correspond à l'ordre de tri (1 pour décroissant).

Mise en place d'une solution permettant la gestion de graphiques avec de multiples axes Y pour les graphiques temporels

Afin de pouvoir ajouter plusieurs axes Y au graphique temporel, le formulaire d'ajout d'un axe Y a été modifié dans le but de correspondre à celui utilisé lors de l'enregistrement ou l'édition d'un graphique non temporel. Cependant, le type d'axe Y a été supprimé puisqu'il est spécifique et ne peut être modifié.

Le but étant, dans le cas d'un graphique temporel, d'avoir un comportement différent vis à vis d'un graphique non temporel. Nous voulons que pour chaque nouvel axe Y, celui-ci vienne dynamiquement se placer sous le précédent. La fonction « AjouterYAxis » a été modifiée pour obtenir ce résultat. Une vérification sur le type de graphique est effectuée afin de conserver le comportement par défaut sur un graphique non temporel.

```

var chartHeight = 400 * (index + 1);
var yAxisHeight = (100 - (index + 1)) / (index + 1);
var yAxisTop = (yAxisHeight + 1) * index;

// Mise à jour de la taille du graphique
chart.setSize(chart.chartWidth, chartHeight);

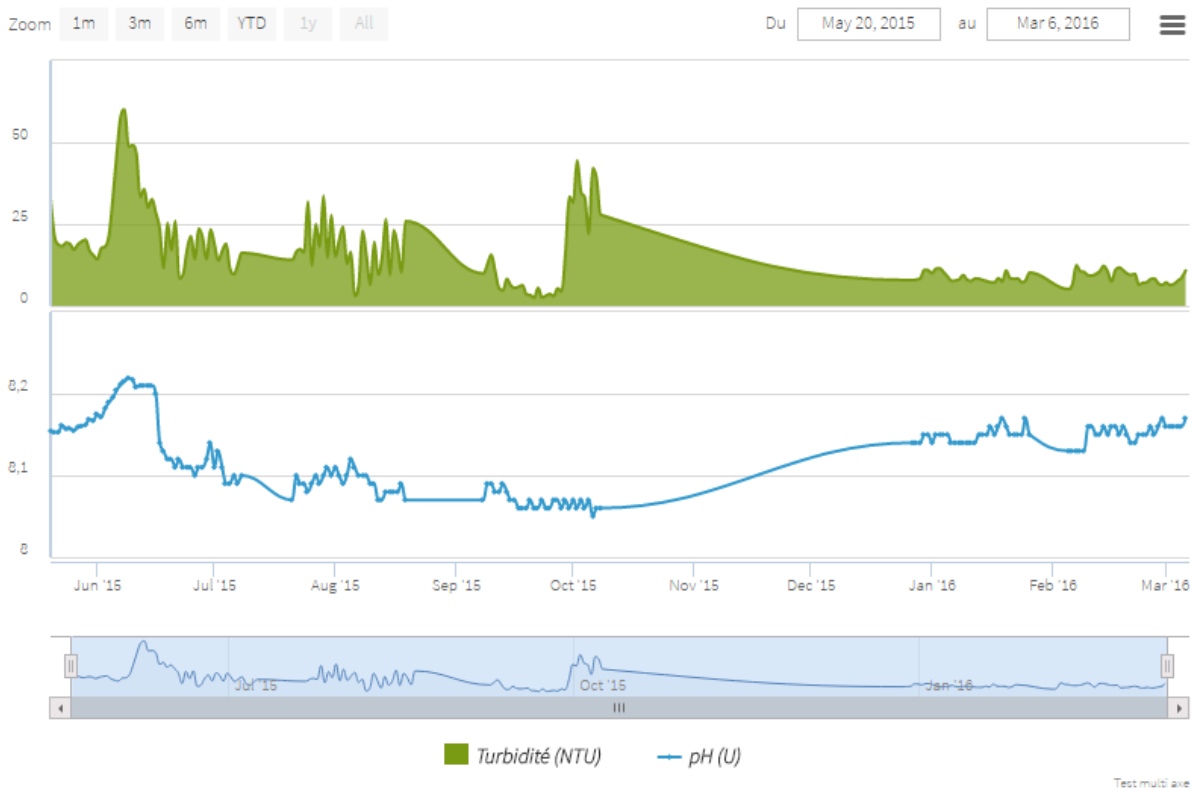
// on rajoute les axes Y
if (index == 0) {
    chart.addAxis({
        id: 'yAxis' + indexYAxisForm,
        title: {
            text: $(titleYAxis).val()
        },
        type: shareVars['firstTypeAxis'],
        height: yAxisHeight + '%',
        lineWidth: 2
    });
} else {
    chart.addAxis({
        id: 'yAxis' + indexYAxisForm,
        title: {
            text: $(titleYAxis).val()
        },
        type: shareVars['firstTypeAxis'],
        top: yAxisTop + '%',
        height: yAxisHeight + '%',
        offset: 0,
        lineWidth: 2
    });
}

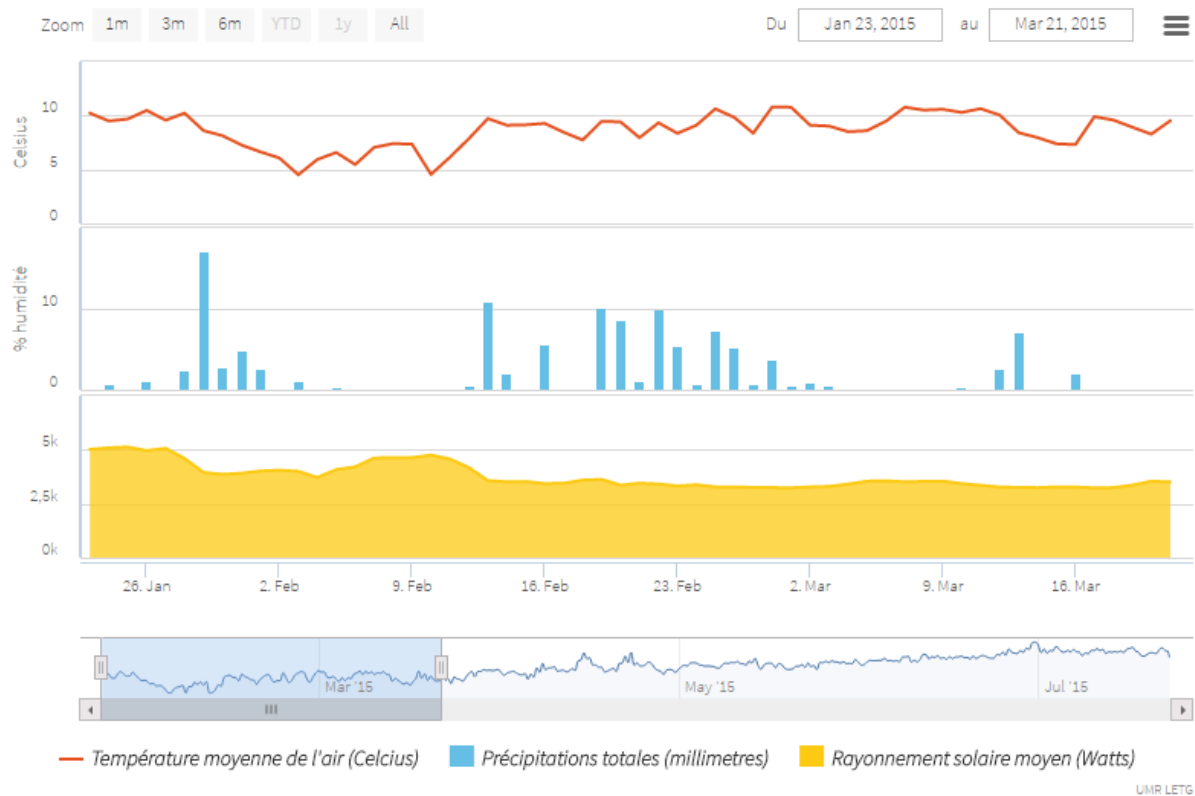
// on met à jour la hauteur des axes Y précédent celui que l'on vient
d'ajouter
for (var i = 2; i < index + 2; i++) {
    chart.yAxis[i].update({
        height: yAxisHeight + '%'
    });
}

// on met à jour la position par rapport au top des axes Y sans changer le
premier
for (var i = 3; i < index + 2; i++) {
    chart.yAxis[i].update({
        top: ((100 / (index + 1)) * (i - 2)) + '%'
    });
}

```


Pour chaque axe Y, la variable index est incrémenté de 1. La hauteur de l'axe étant calculé dynamiquement. Lorsque la variable index est de 0, le premier axe est ajouté et a une hauteur de 99%. La hauteur étant de $(100\% - (\text{index} + 1)) / (\text{index} + 1)$. Si la variable index est différent de 0, on ajoute l'axe en lui assignant une position par rapport au haut du graphique. Il faut ensuite mettre à jour les hauteurs et position par rapport au haut du graphique des axes précédents le dernier afin qu'ils aient tous une hauteur identique et une position par rapport au haut afin qu'il soit les uns derrière les autres.





Ajout de YUI Compressor pour la compression des scripts et fichiers de style

Yui Compressor est un compresseur de fichier JavaScript. Il peut aussi compresser des fichiers de style [CSS](#). Son utilisation n'est pas obligatoire mais permet de réduire la taille de ces fichiers et de rendre leur traitement plus optimal.

Son ajout dans [Symfony](#) se fait en trois étapes :

1. Ajout du fichier [JAVA](#) « yui-compressor.jar » dans le répertoire : app/Ressources/Java
2. Ajout dans le fichier « config.yml » des filtres assetic comme suit :

```
filters:
  cssrewrite:
    apply_to: "\.css$"
  yui_css:
    jar: '%kernel.root_dir%/Resources/java/yuicompressor-2.4.8.jar'
    apply_to: "\.css$"
  yui_js:
    jar: '%kernel.root_dir%/Resources/java/yuicompressor-2.4.8.jar'
    apply_to: "\.js$"
```

3. Ajout dans le layout principale de l'application, lors de l'appel des scripts et fichiers de style, des filtres. Optionnellement, un point d'interrogation spécifie de ne pas exécuter les filtres en environnement de développement.

```
filter='cssrewrite, ?yui_css' //pour les CSS
filter="?yui_js" //pour les JS
```

Lors du déploiement de l'application, la commande [PHP](#) « assetic:dump » appliqué à l'environnement de production, associe les fichiers par type. De sorte qu'avant l'exécution de Yui Compressor, il n'y ait plus qu'un script JS et un fichier de style CSS. Ces deux fichiers sont ensuite compressés par Yui Compressor.

Autres améliorations apportées

Amélioration de la gestion des messages d'erreurs retourné par les requêtes

Les messages d'erreur retourné par le test de connexion à une base de données, les tests avec ou sans traitement de requête et ceux concernant la gestion des utilisateurs était affiché tel quels dans le navigateur.

```
exception 'Exception' with message '
Query :
SELECT total.name,(open.count::float/total.count::float) * 100 as count,total.date FROM (SELECT s.name as name, COUNT(e.id_metadata) as count, e.date_extraction as
date FROM geobs_harvester.sdi s, geobs_harvester.extraction e, geobs_harvester.metadata m, geobs_harvester.dataidentification di, geobs_harvester.keyword k WHERE
s.id_sdi = e.id_sdi AND e.id_metadata=m.id_metadata AND m.dataidentification_id dataidentification=di.id_dataidentification AND di.id_dataidentification =
k.id_dataidentification GROUP BY s.name, e.date_extraction) total JOIN (SELECT s.name as name, COUNT(e.id_metadata) as count FROM geobs_harvester.sdi s,
geobs_harvester.extraction e, geobs_harvester.metadata m, geobs_harvester.dataidentification di, geobs_harvester.keyword k WHERE s.id_sdi = e.id_sdi AND
e.id_metadata=m.id_metadata AND m.dataidentification_id dataidentification=di.id_dataidentification AND di.id_dataidentification = k.id_dataidentification AND
(keywords LIKE '%open data%' OR keywords LIKE '%donn%es ouverte%') GROUP BY s.name, e.date_extraction) open ON total.name=open.name ORDER BY count DES ;
Warning: pg_query(): Query failed: ERROR: syntax error at or near "DES" LINE 1: ...xtraction) open ON total.name=open.name ORDER BY count DES ; ^ in /opt/lampp
htdocs/ManageChart_versionActuelle/src/Mc/BddBundle/Controller/PostgresBDD.php line 66' in /opt/lampphtdocs/ManageChart_versionActuelle/src/Mc/DataListBundle
/Entity/DataList.php:140 Stack trace: #0 /opt/lampphtdocs/ManageChart_versionActuelle/src/Mc/DataListBundle/Controller/DataListController.php(41):
Mc\DataListBundle\Entity\DataList->executeQuery('default', true) #1 /opt/lampphtdocs/ManageChart_versionActuelle/src/Mc/DataListBundle/Controller
/DataListController.php(60): Mc\DataListBundle\Controller\DataListController->getDataList('62', 'SELECT total.na...', '') #2 [internal function]: Mc\DataListBundle
\Controller\DataListController->queryBrutAction() #3 /opt/lampphtdocs/ManageChart_versionActuelle/vendor/jms/cg/src/CG/Proxy/MethodInvocation.php(63):
ReflectionMethod->invokeArgs(Object(EnhancedProxy4b7a927e_b8caa45c09d2a5d1d01de5c9094b116197bd1b28)_CG_\Mc\DataListBundle\Controller
\DataListController), Array) #4 /opt/lampphtdocs/ManageChart_versionActuelle/vendor/jms/security-extra-bundle/JMS/SecurityExtraBundle/Security/Authorization
/Interception/MethodSecurityInterceptor.php(120): CG\Proxy\MethodInvocation->proceed() #5 /opt/lampphtdocs/ManageChart_versionActuelle/vendor/jms/cg/src/CG
/Proxy/MethodInvocation.php(58): JMS\SecurityExtraBundle\Security\Authorization\Interception\MethodSecurityInterceptor->intercept(Object(CG\Proxy
\MethodInvocation)) #6 /opt/lampphtdocs/ManageChart_versionActuelle/app/cache/dev/jms_diextra/proxies/Mc-DataListBundle-Controller-DataListController.php(39):
CG\Proxy\MethodInvocation->proceed() #7 [internal function]: EnhancedProxy4b7a927e_b8caa45c09d2a5d1d01de5c9094b116197bd1b28)_CG_\Mc\DataListBundle
\Controller\DataListController->queryBrutAction() #8 /opt/lampphtdocs/ManageChart_versionActuelle/app/bootstrap.php.cache(2947): call_user_func_array(Array,
Array) #9 /opt/lampphtdocs/ManageChart_versionActuelle/app/bootstrap.php.cache(2909): Symfony\Component\HttpKernel\HttpKernel->handleRaw(Object(Symfony
\Component\HttpFoundation\Request), 1) #10 /opt/lampphtdocs/ManageChart_versionActuelle/app/bootstrap.php.cache(3058): Symfony\Component\HttpKernel
\HttpKernel->handle(Object(Symfony\Component\HttpFoundation\Request), 1, true) #11 /opt/lampphtdocs/ManageChart_versionActuelle
/app/bootstrap.php.cache(2308): Symfony\Component\HttpKernel\DependencyInjection\ContainerAwareHttpKernel->handle(Object(Symfony\Component
\HttpFoundation\Request), 1, true) #12 /opt/lampphtdocs/ManageChart_versionActuelle/web/app_dev.php(28): Symfony\Component\HttpKernel
Kernel->handle(Object(Symfony\Component\HttpFoundation\Request)) #13 {main}
```

Le résultat de la requête étant contenu dans une variable, il est possible avec des « Regex » de tester cette variable pour savoir si elle contient un tableau HTML ou une chaîne de caractère puis d'identifier chaque chaîne de caractère afin de traiter les différents messages d'erreurs retourné.

On testera si l'erreur concerne la connexion à la base de données, une colonne, une relation, un schéma, une table, un point-virgule ou tout autre erreur de syntaxe.

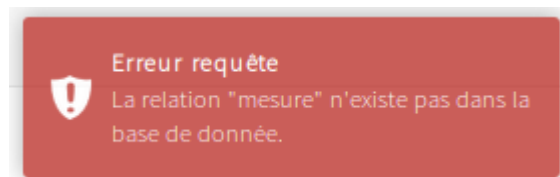
```
// la table n'existe pas
var regexErrorTable = /Table '([\w\W\d\D]++)' doesn't exist/;
var hasErrorTable = regexErrorTable.test(msg);
```

On déclare un pattern puis on teste le résultat de la requête. La variable « hasErrorTable » vaut true uniquement si la « Regex » est vérifié. On pourra à ce moment-là afficher un message d'erreur personnalisé.

Le plug-in jQuery toastr a été ajouté à l'application pour gérer les messages d'erreurs. Celui-ci permet l'affichage de notification paramétrable (couleur, taille, position...).

```
if (hasErrorTable == true) {  
    var tableName = msg.match(regexErrorTable)[1];  
    toastr.error('{{ 'test.qb.error.table.description1'|trans() }} "' +  
tableName + '" {{ 'test.qb.error.description2'|trans() }}', '{{  
'test.qb.error.name'|trans() }}');  
}
```

Ainsi, si la « Regex » est vérifié, on récupère le nom de la table concerné, puis on crée une notification.



Ce comportement a été répété sur les pages d'enregistrement et d'édition d'une connexion à une base de données, lors du test de la connexion. Ainsi que sur les pages de connexion et d'enregistrement d'un utilisateur.

Ajout d'un champ « Nom de connexion » dans les formulaires d'enregistrement et d'édition de connexion à une base de données

Voir [modifications apportées aux données](#)

Correction sur l'export au format image d'un graphique

Lors de l'export sous format image d'un graphique dont la longueur des noms des séries de données est excessive, ceux-ci se chevauche. Il s'agit d'un comportement connu mais non résolu par « Highsoft ». Afin de rendre visible les noms des séries de données sans pour autant corriger ce bogue, nous avons défini une longueur de zone de texte pour les graphiques afin que cela ne se produise plus.

```
exporting:{
  filename: '{{ chart.titleChart|e('js') }}',
  chartOptions: {
    legend: {
      itemWidth: 250,
      itemStyle: {
        fontSize:'10px'
      }
    }
  },
  // necessite l'ajout de l'adresse du serveur distant iuem
  url:'http://wapps.univ-brest.fr:8080/highcharts-export-web/',
}
```

On implique que la longueur de la zone de texte de chaque nom de série doit faire 250px.

Correction sur la façon dont s'ouvre l'URL du crédit

Nous avons vu que lorsque l'URL de crédit est renseignée, celui-ci s'ouvre à présent dans un nouvel onglet du navigateur.

Un événement a été associé au clic sur le crédit, celui-ci permet l'ouverture de l'URL passé en paramètre dans un nouvel onglet du navigateur.

```
events: {
  load: function() {
    this.credits.element.onclick = function() {
      window.open(
        '{{ chart.urlcreditsChart|e('js') }}',
        '_blank'
      );
    }
  }
},
```

Suppression du « navigator »

Le « navigator » est la barre de défilement des graphiques temporel. Celle-ci étant constitué d'une série de données, tout comme les séries que nous ajoutons dans nos graphiques. Lors de l'export au format [CSV](#) ou XLS des données d'un graphique temporel, le comportement par défaut autorise l'export des données du « navigator ».

```
navigator: {  
  series: {  
    includeInCSVExport: false  
  }  
},
```

On précise que le « navigator » ne doit pas être exporté dans les fichiers CSV, XLS ou en mode visualisation dans le navigateur internet.

Correction sur le déploiement de la version de production de l'application

Lors du déploiement de l'application, la commande [PHP](#) « assetic:dump » associant les scripts JS et les fichiers de styles [CSS](#) en deux fichiers distinct, fonctionnait. Mais à la fin du déploiement, les fichiers générés et ceux attendu par l'application ne portait pas le même nom.

La cause du problème était la façon dont on appelait les scripts et fichiers de style. En effet, dans le but d'optimiser le code, sachant que tous les scripts se trouve dans un dossier et que les fichiers de style se trouve tous dans un autre dossier, il est possible de réduire le nombre de ligne de code en appelant tous les scripts en une fois, de même que pour les fichiers de style.

Ainsi l'appel de ces fichiers devient : {Répertoire}/*. {Extension}

Ce fut, en fait ce qui posait problème. Résolu en appelant ces fichiers un par un.

Technologies utilisées

Les technologies utilisées sont les même que celle qui avait été choisi pour développer l'application. Cependant, la technologie [JAVA](#) a été ajouté à l'application par l'apport de « yui compressor ».

Framework et librairies

Les librairies et Framework utilisé ont été mis à jour vers les dernières versions à ce jour. De plus les plug-in jQuery suivant ont été ajouté :

- jquery-tablesorter permettant le tri paramétrable de tableaux,
- jquery-textareafullscreen permettant l'agrandissement de champ textarea,
- toastr permettant l'ajout et le paramétrage de notification.

Conclusion

L'application « ManageChart » étant un projet existant, les différentes étapes de création de projet non pas eu lui d'être. Cependant, ce projet fut, d'une façon différente, tout autant enrichissant car c'était pour moi la première fois que je reprenais le travail d'un autre développeur. Le code déjà écrit étant aussi enrichissant que ce que je lui ai apporté, puisque le travail de développeur consiste aussi en la collaboration. J'ai donc constaté l'importance de commenter son code et de faciliter la relecture tout en faisant en sorte de faciliter, à mon tour le travail d'un développeur qui viendrait à travailler sur ce projet par la suite.

Ce stage, m'a apporté un grand nombre de compétence dont celle du Framework Symfony2, des librairies « HighSoft », du logiciel de gestion de versions Git, ainsi que de l'outil de gestion de projet Redmine.

Suite à l'écriture de ce rapport, il m'a été demandé d'ajouter un rôle intermédiaire entre le « scientifique » et l'« administrateur ». Celui-ci, a les mêmes droits que le scientifique mais se voit également attribuer la gestion des requêtes.

Remerciements

Je tiens à remercier dans un premier temps le centre de formation AFPA pour son accueil et pour m'avoir permis de suivre la formation de Développeur Logiciel dispensé par Mr Delpech, que je souhaite également remercier pour le temps qu'il nous a consacré et pour les connaissances du métier qu'il a partagé avec nous durant la période de formation.

Je remercie également Mr Rouan pour m'avoir permis d'effectuer ce stage, pour le temps qu'il a consacré au suivi de mon travail et pour ses conseils.

Rapport de stage | Lexique

Ajax	Ajax (Asynchronous JavaScript And XML) est une architecture informatique permettant de développer des applications web interactives côté client. Wikipédia
Bootstrap	Conçu par Mark Otto et Jacob Thornton en 2010 sous le nom de <i>Twitter Blueprint</i> et maintenue par la Core Team dont ils font partis. Wikipédia , Core Team .
CSS	Cascading Style Sheets est un langage décrivant la présentation des documents HTML et XML. Wikipédia
CSV	Comma-Separated Values est un format informatique de données tabulaires sous forme de valeurs séparées par des virgules. Wikipédia
Doctrine	Doctrine est un ORM pour PHP créé par Konsta Vesterinen en 2006. Wikipédia , Doctrine .
HighCharts	Conçu par Torstein Hønsi et son entreprise HighSoft en 2009. En 2011, l'entreprise développe Highstock destinée à générer des graphiques temporels. Wikipédia , HighSoft .
HTML	Hypertext Markup Language est un langage de balisage conçu pour représenter des pages web. Wikipédia
JAVA	JAVA est un langage de programmation orienté objet. Wikipédia
JavaScript	Conçu par Brendan Eich en 1995, JavaScript est un langage de programmation de scripts. Wikipédia
Maven	Maven est un outil de gestion et d'automatisation de production de projet logiciel Java. Wikipédia
MVC	Modèle, Vue, Contrôleur est un patron d'architecture logicielle destiné à répondre aux besoins des applications interactives. Wikipédia
MySQL	MySQL est un système de base de données relationnelle. MySQL
ORM	Object-Relational Mapping est une technique de programmation servant à donner l'illusion d'une base de données orientée-objet à partir d'une base de données relationnelle. Wikipédia
PhantomJS	PhantomJS est un navigateur internet sans interface graphique servant à automatiser des interactions avec des pages web. Wikipédia

PostgreSQL	PostgreSQL est un système de base de données open-source orienté objet relationnel. PostgreSQL
PHP	Hypertext-Preprocessor. Conçu par Rasmus Lerdorf en 1994, est un langage de programmation utilisé pour produire des pages web dynamiques via un serveur HTTP. Wikipédia
Responsive	Responsive web design est une technique de conception visant à offrir à l'utilisateur, un confort, quel que soit le support. Wikipédia
Symfony	Symfony est un framework MVC libre écrit en PHP développé par SensioLabs. Wikipédia , Symfony .
Tomcat	Tomcat est un serveur HTTP, conteneur web open source de servlets et JSP Java EE. Wikipédia
URL	Uniform Ressource Locator, désigne une chaîne de caractères utilisée pour adresser des ressources (html, images, son, courriel ...)

Définition d'un graphique

```
// définition des options du graphique
var options = {
  chart: {
    renderTo: 'container',
    zoomType: 'x',
    events: {
      load: function() {
        this.credits.element.onclick = function() {
          window.open(
            $(urlcreditsChart).val(),
            '_blank'
          );
        }
      }
    }
  },
  title: {
    text: $(titleChart).val()
  },
  exporting: {
    filename: $(titleChart).val(),
    chartOptions: {
      legend: {
        itemWidth: 250,
        itemStyle: {
          fontSize: '10px'
        }
      }
    }
  },
  {# necessite l'ajout de l'adresse du serveur distant iuem #}
  url: 'http://wapps.univ-brest.fr:8080/highcharts-export-web/'
},
  navigator: {
    series: {
      includeInCSVExport: false
    }
  },
  subtitle: {
    text: $(subtitleChart).val()
  },
  legend: {
    enabled: $(legendChart).is(':checked')
```

```

    },
    credits: {
        text: $(creditsChart).val()
    },
    plotOptions: {
        series: {
            turboThreshold: 1000,
            gapSize: gapSizeValue
        }
    },
    tooltip: {
        valueDecimals: 2,
        xDateFormat: {% if app.request.locale == 'fr' %}'%d %m
%Y',{% else %}'%Y %m %d',{% endif %}
    },
    xAxis: {
        title: {
            text : produceTitleWithUnit($(titleXAxis).val(),
$(unitXAxis).val())
        },
        type: null,
        ordinal: false
    },
    yAxis: {
        opposite: false
    },
    series: [{
        data: []
    }]
};

```

Fonction wichQuery

```
// fonction permettant d'identifier les bons paramètres à assigner à la
route en fonction du type de base, du type de requête demandé, de l'offset
et de la limit
function whichQuery(page, typebdd, type, offset, limit) {
    if (typebdd == "PostgreSQL") {
        if (type == "QB") {
            var url = '{{ path('data_list_query_brut_postgres',
{"typebdd": "postgres", "limit": "LIMIT", "offset": "OFFSET", "nbPage":
"nbPage"}) }}';
            url = url.replace("OFFSET", "OFFSET " + offset);
            url = url.replace("LIMIT", "LIMIT " + limit);
            url = url.replace("nbPage", limit);
        } else if (type == "QT") {
            var url = '{{
path('data_list_query_traitment_postgres', {"typebdd": "postgres"}) }}';
        }
        query(url, page, type, typebdd);
    } else if (typebdd == "MySQL") {
        if (type == "QB") {
            var url = '{{ path('data_list_query_brut_mysql',
{"typebdd": "mysql", "limit": "LIMIT", "nbPage": "nbPage"}) }}';
            if (limit > 0) {
                url = url.replace("LIMIT", "LIMIT " + offset
+ ", " + limit);
            } else {
                url = url.replace("LIMIT", "LIMIT " +
offset);
            }
            url = url.replace("nbPage", limit);
        } else if (type == "QT") {
            var url = '{{
path('data_list_query_traitment_mysql', {"typebdd": "mysql"}) }}';
        }
        query(url, page, type, typebdd);
    }
}
```

Requête avec pagination

```
/**
 * @Secure(roles="ROLE_ADMIN")
 */
public function queryBrutAction($limit, $offset, $nbPage){
    $idDataSource = $_POST['dataSource'];
    $request = $_POST['request'];
    $attributsSpatiaux = $_POST['attributsSpatiaux'];

    try {
        $dataList = $this->getDataList($idDataSource, $request,
$attributsSpatiaux, '', '');
    } catch (\Exception $e) {
        return new Response($e);
    }

    $listData = $dataList->getData();

    $label = $this->get('translator')->trans('request.label');
    $button = $this->get('translator')->trans('request.button');

    if ($nbPage) {
        $count = 0;
        foreach ($listData as $data) {
            $count = $count + 1;
        }
        $nbPages=ceil($count/$nbPage);

        try {
            $dataList = $this->getDataList($idDataSource,
$request, $attributsSpatiaux, $limit, $offset);
        } catch (\Exception $e) {
            return new Response($e);
        }

        $fields = $dataList->getFields();
        $listData = $dataList->getData();

        $html = '<div id="nbLigne" class="row"
style="display:none;">' . $count . '</div>';
        $html .= '<div style="margin:20px 0;" class="col-sm-5"><form
class="form-inline"><div class="form-group"><div class="input-group"><span
class="input-group-addon">';
        $html .= '<label class="chklabel"><input type="checkbox"
id="chk"><span style="vertical-align:super;"> ' . $label .
'</span></label></span><select id="selectlimit" class="form-
control"></select>';
```

```

        $html .= '<span class="input-group-btn"><button class="btn
btn-default" id="btncsvExp" type="button">' . $button .
'</button></span></div></div></form></div>';

        if ($count > $nbPage) {
            $html .= '<div class="pull-right"><nav><ul
id="paging" class="pagination"><li class="prev"><a href="#" aria-
label="Previous"><span aria-hidden="true">&laquo;</span></a></li>';
                for ($i = 1; $i <= $nbPages; $i++) {
                    $html .= '<li class="page pagehead"><a
href="#">' . $i . '</a></li>';
                }
                $html .= '<li class="next"><a href="#" aria-
label="Next"><span aria-
hidden="true">&raquo;</span></a></li></ul></nav></div>';
            }

            $html .= '<table class="table table-striped"><thead><tr>';

            foreach ($fields as $field)
                $html .= '<th>' . $field . '</th>';

            $html .= '</tr></thead><tbody>';

            foreach ($listData as $data) {
                $html .= '<tr>';
                foreach ($data as $item)
                    $html .= '<td>' . $item . '</td>';

                $html .= '</tr>';
            }
            $html .= '</tbody></table>';

            if ($count > $nbPage) {
                $html .= '<div class="pull-right"><nav><ul
id="paging" class="pagination"><li class="prev"><a href="#" aria-
label="Previous"><span aria-hidden="true">&laquo;</span></a></li>';
                    for ($i = 1; $i <= $nbPages; $i++) {
                        $html .= '<li class="page
pagefoot"><a href="#">' . $i . '</a></li>';
                    }
                    $html .= '<li class="next"><a href="#" aria-
label="Next"><span aria-
hidden="true">&raquo;</span></a></li></ul></nav></div>';
                }
                return new Response($html);
            } else {
                $fields = $dataList->getFields();
            }
        }
    }
}

```

```

        $html = '<div style="margin:20px 0;" class="col-sm-4"><form class="form-inline"><div class="form-group"><div class="input-group"><span class="input-group-addon">';
        $html .= '<label class="chklabel"><input type="checkbox" id="chk" checked><span style="vertical-align:super;"> ' . $label . '<span></label></span><span class="input-group-btn">';
        $html .= '<button class="btn btn-default" id="btncsvExp" type="button">' . $button . '</button></span></div></div></form></div>';
        $html .= '<table class="table table-striped"><thead><tr>';

        foreach ($fields as $field)
            $html .= '<th>' . $field . '</th>';

        $html .= '</tr></thead><tbody>';

        foreach ($listData as $data) {
            $html .= '<tr>';
            foreach ($data as $item)
                $html .= '<td>' . $item . '</td>';

            $html .= '</tr>';
        }

        $html .= '</tbody></table>';

        return new Response($html);
    }
}

```


Pagination dynamique

```
// fonction ajoutant le dispositif de navigation aux requêtes sans
traitement
function navigation(current, typebdd, type, limit, nbPage) {
    console.log('Page courante : ' + current);

    if (nbPage > 10) {
        var firstLi = $('#paging
li.page').first().children('a').html();
        var lastLi = $('#paging
li.page').last().children('a').html();

        $('#paging li.pagehead').hide();
        $('#paging li.pagefoot').hide();
        $('#paging li.pagehead').slice(0, 10).show();
        $('#paging li.pagefoot').slice(0, 10).show();

        if (current > 10 && current<=lastLi) {
            $('#paging li.pagehead').slice(0, current-
10).hide();
            $('#paging li.pagefoot').slice(0, current-
10).hide();
            $('#paging li.pagehead').slice(current-10,
current).show();
            $('#paging li.pagefoot').slice(current-10,
current).show();
        }
    }

    // au clic sur le bouton précédent
    $('#paging li.prev').on('click', function() {
        console.log('Demande d\'affichage de la page précédente');
        var page = current - 1;
        var offset = (page-1) * limit;

        if (current == 1) {
            console.log('Impossible d\'afficher une page
précédent la première');
            return;
        } else if (current >= 1) {
            console.log('affichage de la page ' + page);
            whichQuery(page, typebdd, type, offset, limit);
        }
    });
};
```

```

// au clic sur le bouton suivant
$('#paging li.next').on('click', function() {
    console.log('Demande d\'affichage de la page suivante');
    var page = parseInt(current) + 1;
    var offset = (page-1) * limit;

    if (current == nbPage) {
        console.log('Impossible d\'afficher une page suivant
la dernière');
        return;
    } else if (current <= nbPage) {
        console.log('affichage de la page ' + page);
        whichQuery(page, typebdd, type, offset, limit);
    }
});

// au clic sur un numéro de page
$('#paging li.page').each(function() {
    $(this).on('click', function() {
        var page = $(this).children('a').html();
        console.log('Demande d\'affichage de la page ' +
page);

        var offset = (page-1) * limit;

        console.log('affichage de la page ' + page);
        whichQuery(page, typebdd, type, offset, limit);
    });

    $(this).removeClass('active');

    if ($(this).children('a').html() == current) {
        $(this).addClass('active');
    }
});

setTimeout(function() {
    var limit = localStorage.getItem("limit");
    $("#select#selectlimit").val('LIMIT ' + limit);
}, 500);
}

```