[Date]

jeoffrey godart

[nom de la société]

[Titre du document]

[Sous-titre du document]

# Historique du document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Auteur | Modification | Date |
| 0.0.1 | Godart Jeoffrey | Création | 29/09/2015 |
| 0.1.0 | Godart Jeoffrey | Modification | 16/11/2015 |
| 0.2.0 | Godart Jeoffrey | Modification | 01/12/2015 |
| 1.0.0 | Godart Jeoffrey | Modification | 28/01/2016 |

Table des matières

[Historique du document 1](#_Toc451257266)

[La solution 3](#_Toc451257267)

[Architecture réseau 3](#_Toc451257268)

[Schéma réseau 3](#_Toc451257269)

[Descriptif 3](#_Toc451257270)

[Architecture logicielle 4](#_Toc451257271)

[Schéma des composants 4](#_Toc451257272)

[Descriptif 4](#_Toc451257273)

[Les méthodes de conception 6](#_Toc451257274)

[La méthode de développement 6](#_Toc451257275)

[Les patrons de conceptions 7](#_Toc451257276)

[Les bases de données 7](#_Toc451257277)

[La base de données serveurs 7](#_Toc451257278)

[La base de données client 7](#_Toc451257279)

[Le site web 8](#_Toc451257280)

[Architecture du projet 8](#_Toc451257281)

[Gestion de la connexion 8](#_Toc451257282)

[Accueil 8](#_Toc451257283)

[*Maquette fonctionnelle détaillée* 8](#_Toc451257284)

[Gestion des utilisateurs / groupe / Administrateur 8](#_Toc451257285)

[*Maquette fonctionnelle détaillée* 8](#_Toc451257286)

[Gestion des Questionnaires 8](#_Toc451257287)

[*Maquette fonctionnelle détaillée* 8](#_Toc451257288)

[Gestion des résultats 8](#_Toc451257289)

[*Maquette fonctionnelle détaillée* 8](#_Toc451257290)

# La solution

## Architecture réseau

### Schéma réseau

### Descriptif

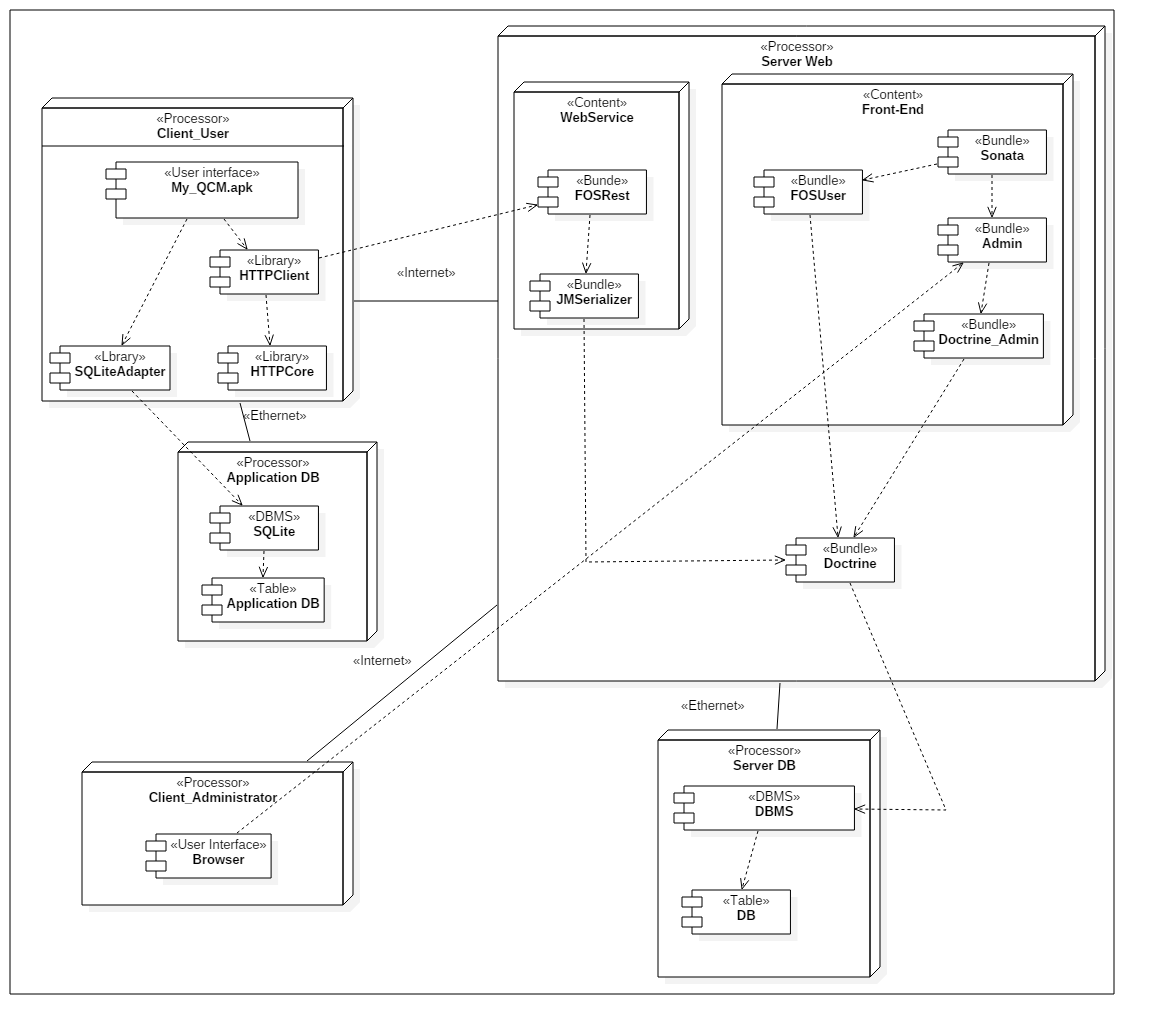
|  |  |
| --- | --- |
| Poste Client | |
| Paramètres | Composant |
| Nom du poste | Pc-Jeoff |
| Système d’exploitation | Windows 10 éditions familiale |
| Volume du disque Dur | 750 GB |

|  |  |
| --- | --- |
| Services du poste | |
| Nom du service | Utilité |
| Wamp Server 2.5 (Apache) pour le web | Hébergement de la partie Web |
| Serveur de base de données (MySQL 5.6.17) | Stockage des données |
| Service de gestion de base de données (PhpMyadmin 4.1.14) | Administration de la base de données |

|  |  |
| --- | --- |
| Configuration du réseau | |
| Paramètres | Valeur |
| Masque sous-réseau | 255.255.255.0 |
| DNS | Automatique |
| IP | \*.\*.\*.\* |

## Architecture logicielle

Schéma des composants



### Descriptif

#### Node client\_user

Le node client\_user représente la partie applicative de la solution et regroupe plusieurs composants :

- Une interface utilisateur représentant l'Apk de l'application, le programme exécuté sur le téléphone.

Cet Apk dispose de deux librairies :

-HttpClient qui est la librairie qui permet d'exécuter les requêtes de communication avec le WebService, elle hérite de HTTPCore.

- SQLiteAdapter qui elle est une libraire créer qui permet la communication avec la base de données en SQLite. Elle contient toute les requête de gestion de la base (Ajout, listé, suppression...).

#### Node Application DB

Ce node représente la partie base de donnée de l'application qui est composé de :

- Un système de gestion de base de données SQLite

- Ainsi que de la base de données

#### Node Client\_Administrator

Le node Client\_Administrator illustre la partie d'administration de la solution côté client, ses l'interface par navigateur donnant accès au front-end.

#### Node Server Web

Ce node exprime toute la partie serveur web et est divisé en 3 parties :

- Le node WebService

- Le node Front-end

- Le Bundle Doctrine qui permet à Symfony de communiquer avec le système gestion de base de données contenu dans le Node Server DB

#### Node WebService

Node permettant la réception et l'envoi de donnée entre l'application et le serveur Web.

Il est composé de deux Bundles :

- FOSRest qui permet de générer des urls renvoyant des flux Json pour permettre la circulation des infos entre l'applicatif et le serveur

- JMSerializer permet de sérialiser donc de mettre en forme les données envoyées via doctrine qui seront par la suite renvoyer par FOSREST Bundle

#### Node Front-end

Node qui permet d'administrer la solution au travers d'un portail web via plusieurs Bundle Symfony :

- Bundle Sonata permettant de mettre en place une interface d'administration facilement et rapidement. Qui est en fait un bundle regroupant d'autre bundle :

- Bundle Admin Fournissant à partir des informations reçu par le Bundle Doctrine-Admin une interface visuel auto-généré.

#### Node Server DB

A l'image du Node Application DB représente la partie base de données de notre serveur web.

En effet nous disposons également d'un système de gestion de base de données ainsi que de la base de données de l'application.

## Les méthodes de conception

### La méthode de développement

#### Convention de nommage

Le nommage devra respecter plusieurs conventions :

##### Générale

* Camel Case

##### Java

* Toutes les lettres de l’alphabet
* Les chiffres de 0-9
* Package :
  + Tout en minuscules
  + Root : com
* Classe et interface :
  + Première lettre en majuscule
  + Première lettre de chaque mot en majuscule
  + Nom simples et descriptifs
  + Pas d’acronymes
* Variable :
  + Première lettre en minuscule
  + Noms simples et descriptifs
  + Pour un usage local variable d’une seule lettre
* Constante :
  + Tout en majuscule
  + Mot séparer par un « \_ »
  + Noms simples et descriptifs

##### Windows Phone

* Classe, structures, interfaces, méthodes :
  + Débute par une majuscule
  + Toutes les lettres de l’alphabet
  + Evité les caractères spéciaux
* Variables, Locales :
  + Débute par une minuscule
  + Toutes les lettres de l’alphabet
  + Evité les caractères spéciaux

##### Objective-C

* Méthodes :
  + Première lettre en minuscule
  + A chaque nouveau mot en majuscules

#### Commentaires

Les commentaires devront :

• Formatés pour suivre les conventions internationales.

• Permettre la mise en place d’une documentation technique du code source.

• Être écrit en anglais pour faciliter le partage du projet quel que soit la langue.

• Une personne ne connaissant pas le projet aura la possibilité de comprendre et reprendre le code.

Ils pourront aussi illustrer une action à réaliser par la suite comme des notes.

Certains éléments sont obligatoirement à commenter :

• Méthodes

• Classes

• Variables de classe

Un exemple complet est fourni en Annexe de ce document.

#### Les tests unitaires

#### Versionning

Les patrons de conceptions

MVC

MVVM

# Les bases de données

La base de données serveurs

Diagramme classe détaillée

Dictionnaire des données

La base de données client

Diagramme classe détaillée

Dictionnaire des données

# Le front-end

Architecture du projet

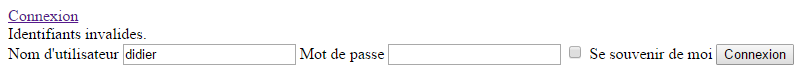
### Gestion de la connexion

#### Maquette fonctionnelle détaillée

##### Connexion



##### Erreur dans la connexion



#### Détails des fonctions

#### Connexion

La connexion est gérer par le BundleFOSUser.

En effet au moment de la création de mes entités j’ai attribué à la classe User un étendu d’un élément de FOSUser qui est BaseUser.



Ensuite une fois la configuration

##### Erreur dans la connexion

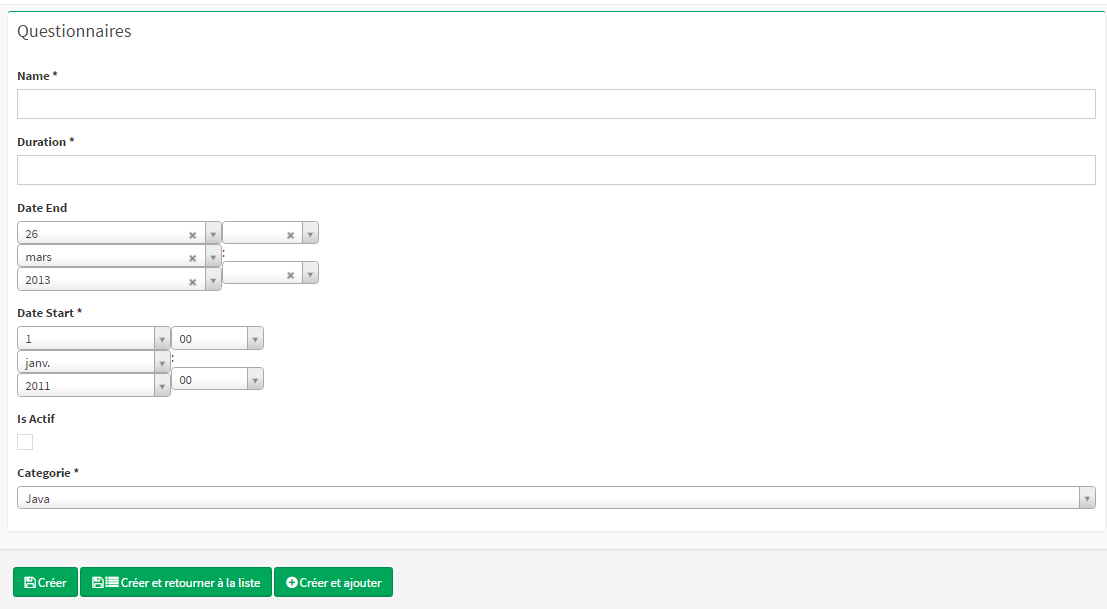
A partir du moment où nous quittons la connexion, le bundle Sonata prend le relais et génère automatiquement tout le processus de gestion des éléments. Afin de connaitre le flux de son fonctionnement il convient de se documenter sur le site officiel du Bundle Sonata.

Néanmoins je peux montrer un aperçu de son fonctionnement et du paramétrage nécessaire à la construction du Front-end selon les besoins de notre solution.

## Gestion des QCM

### Ajout d’un QCM

#### Maquette



|  |  |  |  |
| --- | --- | --- | --- |
| Element du formulaire | | | |
| Nom | Type | Valeur | Vérification |
| Name | TextField | Nom du QCM | La valeur du champ doit être unique, obligatoire |
| Duration | TextField | Durée du QCM | Obligatoire |
| Date End | DateTime | Date de fin du QCM |  |
| Date Start | DarteTime | Date à partir de quand le QCM va être disponible | Obligatoire |
| Is Actif | Boolean | Définis si le QCM est disponible | Obligatoire |
| Categorie | Category | Category du QCM | Obligatoire |

#### Détails de Configuration

|  |  |  |
| --- | --- | --- |
| Signature de la fonction | Paramètres d’entrée | Utilité |
| protected function configureFormFields | FormMapper $formMapper | Définis les éléments à afficher dans la page d’ajout (name, duration, dateEnd, dateStart, isActif,category) |

4- Le Webservice

4.0 Architecture du projet

4.1 Gestion du fux de connexion

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

4.2 Gestion du flux QCM

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

4.3 Gestion du flux des résultats

- Diagramme de séquence

- Diagrammme d'activité

- Cas de test

5- Application

5.0 Architecture du projet

5.1 Connexion sécurisée

- Maquette détaillée

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

5.2 L'accueil

- Maquette détaillée

- Diagramme de séquence

- Diagramme d'activité

5.3 Gestion des catégories

- Maquette détaillée

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

5.4 Gestion des qcms

- Maquette détaillée

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

5.5 Répondre aux qcms

- Maquette de détaillée

- Diagramme de séquence

- Diagramme d'activité

- Cas de test

5.6 Validation et envoie de résultat

- Flux détaillé

- Diagramme de séquence

- Fonctions implémentées

- Diagramme d'activité

- Cas de tests

5.7 Le mode hors connexion et la mise à jours

- Flux détaillé

+ Diagramme de séquence

- Fonctions implémentées

+ Diagramme d'activité

- Cas de tests

ANNEXE

----------

1- page d'exemple de code commenté

2- dictionnaires des données serveur

3- dictionnaire des données terminal mobile