Découverte du Framework Angular JS avec le contexte GSB

Partie 4 Création d'un rapport de visite

Description du thème

Propriétés	Description
Intitulé long	Découverte du Framework Angular JS avec le contexte GSB dans sa partie gestion des rapports de visite. Ceci est la partie 4
Formation concernée	BTS SIO option SLAM
Matière	SLAM 4
Présentation	Accompagnement dans la découverte d'Angular. Développement pas à pas d'une application à partir du contexte GSB
Notions	• D4.1 - Conception et réalisation d'une solution applicative • D4.2 - Maintenance d'une solution applicative Savoir-faire • Programmer un composant logiciel • Exploiter une bibliothèque de composants • Adapter un composant logiciel • Programmer au sein d'un Framework
Prérequis	Les principes du développement web, PHP, SQL, JavaScript
Outils	SGBD MySQL, un environnement de développement
Mots-clés	GSB, Angular JS, Ajax, MVVM
Durée	6 heures
Auteur(es)	Patrice Grand. Relectures de Cécile Nivaggioni, Yann Barrot, Luc Frebourg et Gaëlle Castel.
Version	v 1.0
Date de publication	novembre 2016

Présentation

La quatrième partie va aborder l'ajout d'un nouveau rapport de visite par le visiteur. Vous disposez d'une application de départ, dans le répertoire gsbAJSV4.0.

Pour ceux qui souhaitent partir des sources obtenues à la fin de la partie 3, vous devez, à partir des sources fournies dans le répertoire gsbAJSV4.0, récupérer :

- les fichiers ajax/*traiterajouterrapport.php* et ajax/*traiterrecherchemedicaments.php* ;
- la définition de la route */nouveaurapport* tel que cela est fait dans *js/app.js* ;
- les sources du contrôleur *nouveauRapportController* ;
- les vues *medicamentOffert.html* et *nouveauRapport.html*.

1 Les écrans attendus

Voici l'ensemble des écrans attendus pour l'ajout d'un nouveau rapport de visite :

Fig 1

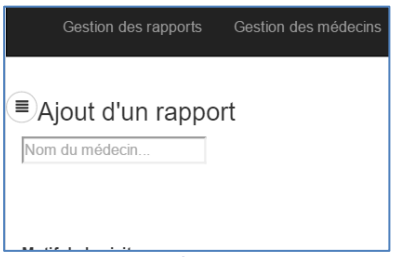


Fig 2

Fig 3

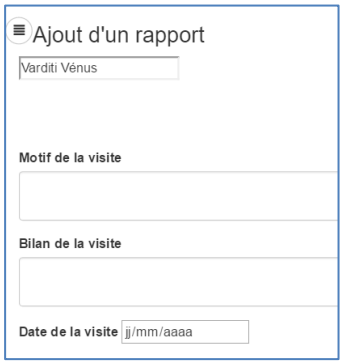


Fig 4

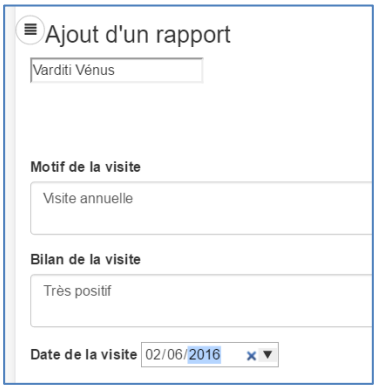


Fig 5

Fig 6

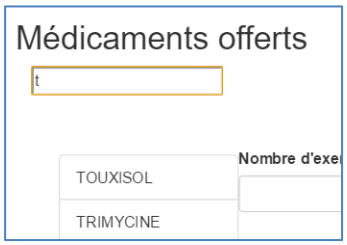


Fig 7

Fig 8

Fig 9

Fig 10

Fig 11

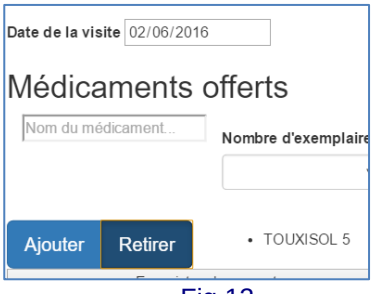


Fig 12

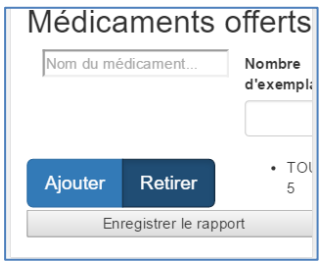


Fig 13

- le visiteur commence par sélectionner le médecin (figures 1 à 4) ; ce sont les mêmes fonctionnalités que sur la page de gestion d'un médecin,
- il saisit le motif, le bilan et la date (figure 5),
- il choisit un médicament à partir des premières lettres de son nom commercial (figures 6, 7 et 8),
- il sélectionne la quantité offerte de ce médicament (figure 9),
- il l'ajoute à la liste des médicaments offerts (qui apparaît) ; de nouvelles saisies sont possibles (figures 10, 11).
- il peut retirer de la liste le dernier médicament (figure 12),
- le visiteur termine par enregistrer le rapport de visite.

2 La partie recherche du médecin

Lors de l'ajout d'un nouveau rapport les principaux éléments utilisés seront :

- la route */nouveaurappport* définie dans js/app.js ;
- la vue *nouveauRapport.html* (qui est fournie) pour laquelle une description est donnée ci-après ;
- le contrôleur *nouveauRapportController* dont une partie du code vous est donnée.

Tout d'abord, commentons quelques éléments de la vue *nouveauRapport.html* :

```
1   <div ng-include src="'vues/medecins.html'" ></div>
2   <form class = "col-lg-6"  ng-submit="enregistrer()">
3       <div class="form-group">
4           <label for = "motif" > {{::lblMotif}} </label>
5           <textarea type="text"
6                 ng-model = "r.motif"
7                 class = "form-control"
8                 required="true" >
9           </textarea>
10      </div>
11      <div class="form-group">
12          <label for = "bilan" > {{::lblBilan}} </label>
13          <textarea type="text"
14                ng-model = "r.bilan"
15                class = "form-control"
16                required="true" >
17          </textarea>
18      </div>
19      <div class="form-group">
20          <label for = "date">{{::lblDate}}</label>
21          <input type ="date"
22                 ng-model="r.date"
23                 required="true"/>
24      </div>
25        <h2>{{::lblTitreMedicament}}</h2>
26      <div ng-include src="'vues/medicamentOffert.html'" ></div>
27      <input type ="submit" class="btn-group btn-group-justified" value ="{{::lblEnvoyer}}">
28  </form>
```

- ligne 1, insertion de la vue de recherche du médecin,
- lignes 2 à 24, un formulaire et les zones de saisie des informations du nouveau rapport,
- ligne 26, insertion d'une vue décrite dans un fichier distinct afin de gérer les médicaments.

Remarque : les propriétés *bindées* se réfèrent à un *objet r* qui devra être déclaré dans le contrôleur (vous pourrez vous appuyer sur ce qui a été vu dans la partie 3 – mise à jour des médecins) lors de l'implémentation de l'enregistrement d'un rapport (voir 4 – Enregistrement du rapport).

La première étape est la recherche d'un médecin (tel que cela a été traité dans la gestion des médecins.
La partie vue de la recherche du médecin (*medecins.html*) est entièrement réutilisée (grâce à la directive *ng-include*).
La partie contrôleur, identique à celle utilisée pour la gestion des médecins (voir contrôleur *medecinsController*), ne sera pas ici réutilisée. Nous aborderons dans la cinquième partie « Pour aller plus loin » une solution élégante pour réutiliser du code ; cela fera l'objet de la découverte des *services* sous Angular JS.

```
app.controller('nouveauRapportController',function($scope, $http,$rootScope){
    $scope.titre = "Ajout d'un rapport";
```

Travail à faire

- Compléter le code du contrôleur *nouveauRapportController* afin de :

- valoriser les propriétés utilisées par la vue *nouveauRapport.html* en veillant à respecter les écrans attendus ;
- gérer la recherche du médecin (figures 1, 2, 3 et 4).

- Tester.

3 Ajout des médicaments

La vue des médicaments offerts est un peu plus délicate à réaliser (*medicamentOffert.html*) ; on utilise un tableau pour disposer les éléments HTML (figure 11). Elle est disponible dans le projet fourni.

3.1 Choix du médicament

La partie haute reprend les principes de la recherche de médecin :

```
1  <table>
2      <tr >
3          <td>
4              <div class = "col-lg-6" >
5                  <div class="form-group">
6                      <input type="search"
7                         ng-model = "rechercheMedicament.nom"
8                         ng-change = "chargerMedicaments()"
9                         placeholder = "Nom du médicament..."
10                     />
11                 </div>
12             </div>
13                 <br><br>
14             <div class="list-group">
15                 <ul>
16                     <li class = "list-group-item"
17                         ng-repeat = "medicament in medicaments"
18                         ng-click = "choisirMedicament(medicament)" >
19                             {{medicament.nomCommercial }}
20                     </li>
21                 </ul>
22             </div>
23         </td>
```

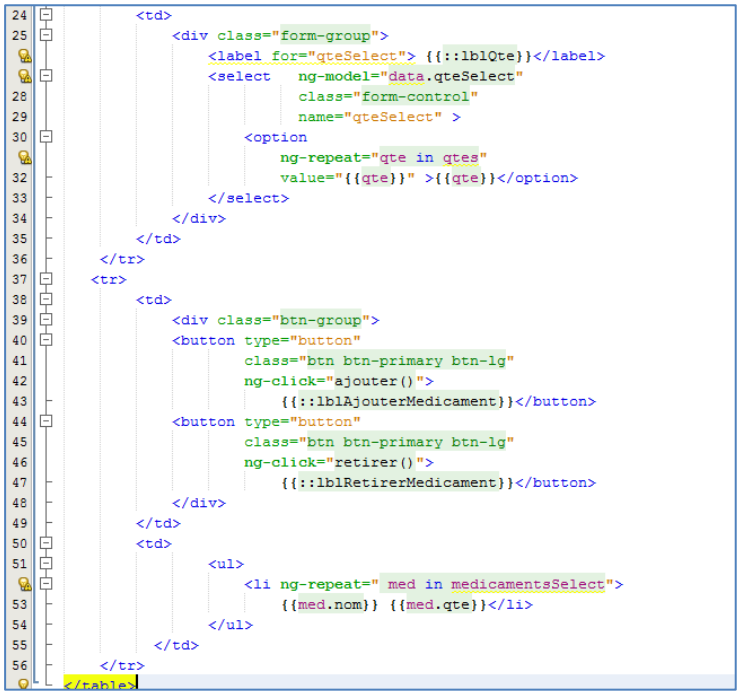
- ligne 7, la propriété bindée l'est à l'objet *rechercheMedicament,*
- le contrôleur devra définir le code de la méthode *chargerMedicaments*, ligne 8,
- ligne 17, la directive *ng-repeat* utilise *medicaments* qui devra être valorisé dans le contrôleur,
- ligne 18, appel de la méthode c*hoisirMedicament* avec le médicament courant.

Travail à faire

Compléter le contrôleur avec le code qui permet de gérer le choix du médicament (figures 6, 7 et 8) puis tester la recherche d'un médicament.

3.2 Gestion des médicaments

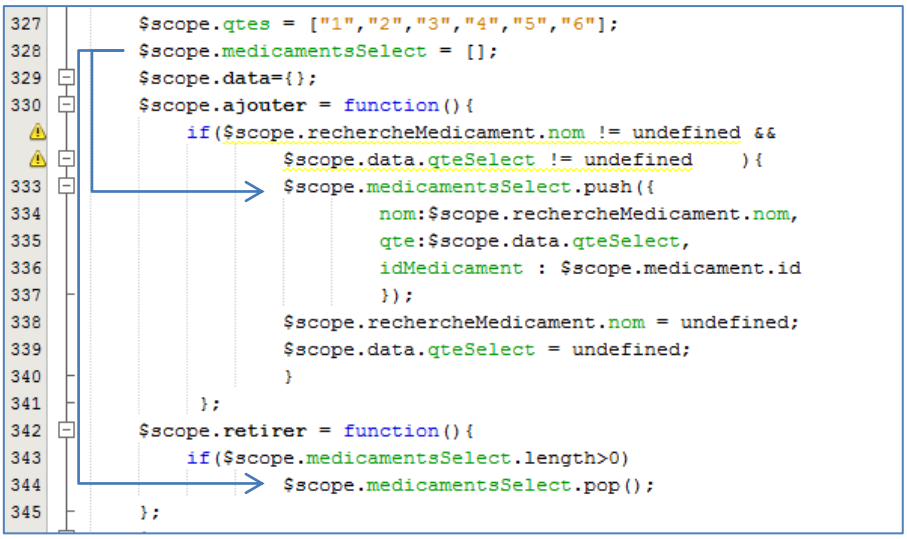
L'extrait de la vue *medicamentOffert.html* présentée ci-dessous correspond aux figures 9, 10, 11 et 12 :



```
24          <td>
25              <div class="form-group">
                    <label for="qteSelect"> {{::lblQte}}</label>
                    <select   ng-model="data.qteSelect"
28                            class="form-control"
29                            name="qteSelect" >
30                      <option
                            ng-repeat="qte in qtes"
32                          value="{{qte}}" >{{qte}}</option>
33                  </select>
34              </div>
35          </td>
36      </tr>
37      <tr>
38          <td>
39              <div class="btn-group">
40              <button type="button"
41                      class="btn btn-primary btn-lg"
42                      ng-click="ajouter()">
43                          {{::lblAjouterMedicament}}</button>
44              <button type="button"
45                      class="btn btn-primary btn-lg"
46                      ng-click="retirer()">
47                          {{::lblRetirerMedicament}}</button>
48              </div>
49          </td>
50          <td>
51              <ul>
                    <li ng-repeat=" med in medicamentsSelect">
53                      {{med.nom}} {{med.qte}}</li>
54                  </ul>
55            </td>
56      </tr>
</table>
```

- les lignes 27 à 33 définissent l'élément HTML *select* : la propriété *data.qteSelect* permettra au contrôleur de récupérer l'option sélectionnée. La directive *ng-repeat* permet d'ajouter les options ; notez l'attribut classique *value*. Le contrôleur devra, bien sûr, fournir les valeurs de *qtes*.
- les lignes 39 à 48 permettent de construire les deux boutons *Ajouter* et *Retirer.*
- les lignes 52 et 53 permettent d'afficher les médicaments choisis.

La partie qui gère l'ajout et le retrait de médicament, un peu délicate, vous est fournie ; commentons les points importants :



```
327     $scope.qtes = ["1","2","3","4","5","6"];
328     $scope.medicamentsSelect = [];
329     $scope.data={};
330     $scope.ajouter = function(){
            if($scope.rechercheMedicament.nom != undefined &&
                    $scope.data.qteSelect != undefined    ){
333                 $scope.medicamentsSelect.push({
334                         nom:$scope.rechercheMedicament.nom,
335                         qte:$scope.data.qteSelect,
336                         idMedicament : $scope.medicament.id
337                         });
338                 $scope.rechercheMedicament.nom = undefined;
339                 $scope.data.qteSelect = undefined;
340                 }
341         };
342     $scope.retirer = function(){
343         if($scope.medicamentsSelect.length>0)
344                 $scope.medicamentsSelect.pop();
345     };
```

- ligne 327, la propriété *qtes* (un tableau) va être utilisée dans la liste *select* grâce à la directive *ng-repeat,*
- la propriété *medicamentsSelect*, ligne 328, permet d'afficher les médicaments sélectionnés (ligne 52 de la vue précédente). Lorsque l'on clique sur *ajouter*, on ajoute, au tableau

medicamentSelect, un objet avec 3 champs grâce à la méthode push (ligne 333). On en retire le dernier élément grâce à la méthode pop (ligne 344).

- les lignes 338 et 339 mettent à « blanc » le champ de saisie du médicament et la liste des quantités. Elles utilisent la constante *undefined* qui avec JavaScript se distingue de *null* ; ce qui n'est pas sans troubler bien des développeurs habitués à d'autres pratiques…

4 Enregistrement du rapport

C'est la dernière étape ; elle utilise ce que nous avons vu précédemment :

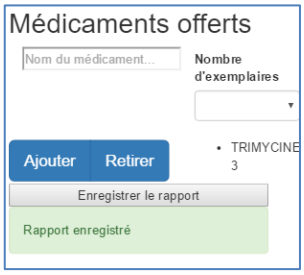
- récupération de la date (partie 2 et la mise à jour des rapports),
- les appels Ajax.

Pour mener à bien votre appel Ajax, il vous faudra respecter le format attendu des données passées dans le fichier *traiterajouterrapport.php* fourni :

```
2   session_start();
    $data = json_decode(file_get_contents('php://input'),true);
4   if(isset($_SESSION['visiteur'])){
5       require_once '../data/pdogsbrapports.php';
6       $idMedecin = $data['idMedecin'];
7       $bilan = $data['bilan'];
8       $motif = $data['motif'];
9       $date = $data['date'];
10      $lesMedicaments =  $data['lesMedicaments'];
11      $pdo = PdoGsbRapports::getPdo();
12      $idVisiteur = $_SESSION['visiteur']['id'];
13      $ret = $pdo->ajouterRapport($idMedecin,$idVisiteur, $bilan, $motif, $date, $lesMedicaments);
14      echo $ret;
15  }
16  else{
```

Travail à faire

Ecrire la méthode *enregistrer* dans le contrôleur et modifier la vue afin d'obtenir les écrans suivants :



Ceci termine cette quatrième partie. Le corrigé est disponible dans le répertoire **gsbAJSV4.1.**