Conception et création d'une application web de gestion de sondages : *inQuire*

Groupe 5 - Thiry Jérôme & Vermeiren Rémy

17 mai 2015

Table des matières

Introduction

Ce rapport concerne le projet d'approfondissement en sciences informatiques, mis en route dans le contexte du cours LSINF1212 dispensé à l'Université Catholique de Louvain. Il nous a été demandé de concevoir une application web dont le sujet était laissé au choix. Nous avons donc décidé de créer une application de gestion de sondages qui permettrait, entre autres, de créer, remplir et gérer des sondages en ligne, ainsi que d'en voir les résultats. Ce rapport a pour but d'expliquer les différentes étapes de la création de cette application, de la phase d'analyse, en passant par sa conception jusqu'à son implémentation en elle-même. Le code complet de l'application est disponible sur https://github.com/jerthiry/inquire.

1 Phase d'analyse

Comme dans tout projet, celui-ci a commencé par une phase d'analyse. Celle-ci a été entre autres réalisée en utilisant le développement par comportements, et, en particulier, en accord en accord la convention *cucumber* dont l'algorithme 1 donne un extrait d'un exemple décrit en début de projet, et qui concerne la création de sondage. Ces différents travaux nous ont permis de cerner plusieurs besoins.

On peut par exemple citer le fait que l'utilisateur doive être connecté afin de pouvoir créer un sondage. En effet, les résultats d'un sondage n'étant accessibles que par son créateur, il était nécessaire de faire ce choix.

Citons encore le fait de pouvoir avoir des sondages publics mais aussi des sondages privés. Cela nous semblait important car les besoins sont multiples. Une enquête entre amis n'a pas la même ampleur qu'une enquête pour une étude, par exemple. Cependant, nous avons été amené, après réflexion, à autoriser les utilisateurs non connectés à pouvoir remplir un sondage. Cela est moins contraignant pour l'utilisateur et permet donc plus de flexibilité. Un bon compromis entre ces deux choix a été de ne pas reprendre les sondages privés sur la page principale. Ils ne sont donc accessibles qu'à partir du permalink leur correspondant. Il en va de même si la date de clotûre du sondage est dépassée.

Il nous a aussi semblé important de permettre aux utilisateurs de modifier leurs données personnelles, de pouvoir avoir une vue d'ensemble sur leurs sondages, ainsi que de proposer à tous les visiteurs du site une liste des sondages publics les plus récents.

Algorithme 1 BDD - Création de sondage

```
1  Feature: Survey creation
2
3    In order to collect data
4    As a user
5    I want to create a survey
6
7    Scenario: User can create a survey
8        Given the user is registered
9        When the user requests to create a survey
10       Then the application proposes different settings
11       And the user fills in the settings
12       And the user confirms his choices
13       And the application creates the survey
```

2 Choix de conception

Cette section décrit les choix de conception, et entre autres l'architecture du système. Le système se base sur plusieurs technologies qui s'occupent de différentes parties de celui-ci.

L'application se divise en trois couches importantes. En premier lieu, nous trouvons la couche utilisateur, qui gère l'affichage et envoie des requêtes au serveur. Ensuite vient le noyau de l'application, qui gère les requêtes et, enfin, la couche base de données, implémentée en MongoDB, un type de bases de données non-relationnelles, qui stocke les données sous forme d'objet en Javascript.

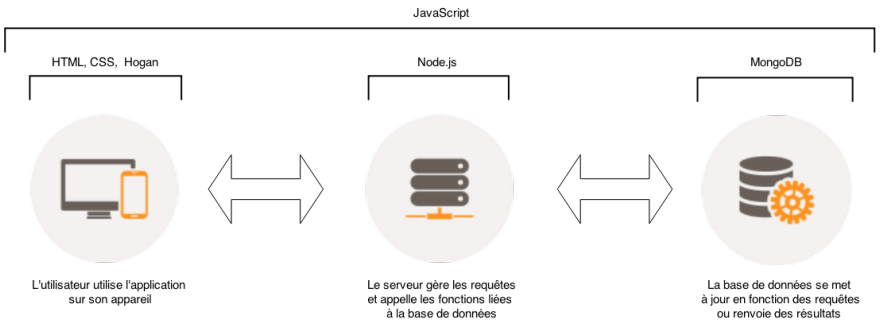

FIGURE 1 – Couches de l'application

2.1 Couche utilisateur : HTML, CSS, Javascript, Hogan

Le dossier *views* contient des fichiers qui gèrent l'affichage de l'application dans le navigateur, entre autres les différentes pages HTML affichées sur le site. Ces pages utilisent le CSS de Bootstrap. Certaines pages HTML ne sont pas des pages complètes puisqu'elles ne servent qu'à venir s'insérer dans une autre page HTML. On trouve aussi les images (logos, background, ...) dans le dossier images, ainsi que des scripts Javascript, utiles pour changer de contenu dynamiquement en fonction des choix de l'utilisateur, notamment lors du choix du type de question à créer lors de l'initialisation d'un questionnaire.

Lorsque le serveur désire interpréter des pages HTML, ils leur passe des argu-

ments, qui sont alors traités grâce à un outil de *templating*, Hogan.js.

2.2 Noyau de l'application : Node.js

Le noyau de l'application a été implémenté grâce à Javascript et Node.js. Celui-ci gère plusieurs opérations, dont entre autres :

- Dans app.js :
  - Le lancement du serveur ;
  - La liaison avec la base de données.
- Dans le dossier routes :
  - Le routage des différentes pages de l'application, implémenté avec l'aide du module *Express* ;
  - La gestion des requêtes GET et POST sur le serveur, c'est-à-dire les fonctions appelées lorsque l'on afficher une page (on peut alors l'interpréter et notamment lui passer des variables grâce à Hogan), où que l'on clique sur tel ou tel bouton par exemple. C'est ici aussi que l'on trouvera les appels aux fonctions dédiées à la base de données.

2.3 Couche base de données

Les fichiers constituant la couche base de données se trouvent dans le dossier *persistence*. Ceux-ci contiennent les fonctions qui interagissent directement avec la base de données. Il existe un fichier par type d'objets que l'on peut retrouver dans la base de données (utilisateurs, sessions, questionnaires et réponses). On y trouve aussi bien des fonctions d'insertion que de récupération ou de mise à jour.

3 Choix d'implémentation du système

Dans cette section sont expliqués nos choix d'implémentation. Nous ne reprenons évidemment que les choix importants, ou ceux qui nous semblent les moins évidents.

3.1 Base de données

La base de données étant implémentée sur base d'objets Javascript, elle est très flexible d'utilisation. Nous avons cependant du faire certains choix qui nous semblaient pertinents.

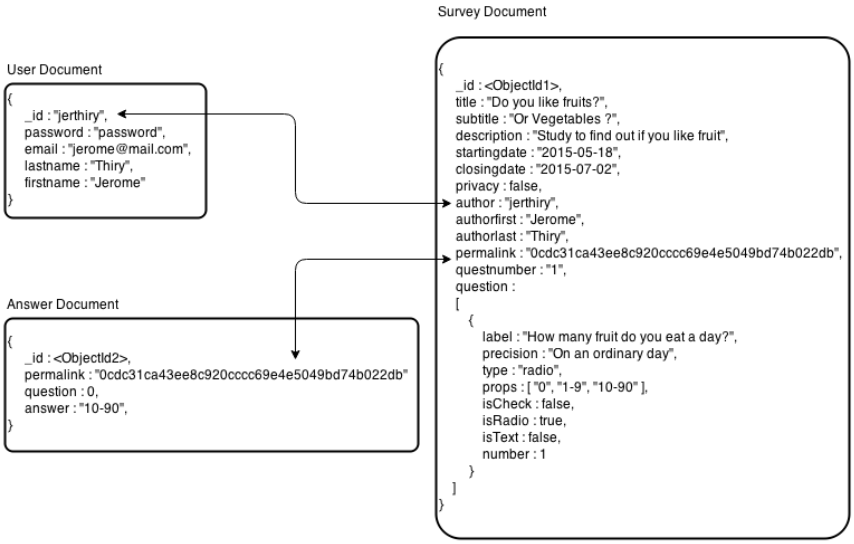

FIGURE 2 – Diagramme de la base de données

En premier lieu, nous avons décidé de séparer les objets questionnaires et réponses. Nous aurions pu rajouter les réponses à l'objet questionnaire à chaque nouvelle soumission, mais cela n'aurait pas été idéal. En effet, si le sondage est un minimum populaire, cela signifie qu'il faudrait pouvoir écrire sur l'objet constamment, ce qui engendrerait des risques de corruption ou, au mieux, de ralentissement. De plus, la suppression des réponses (pas encore implémentée mais imaginable dans le futur) ne s'en trouve que simplifiée.

Ensuite, le fait que le *templating* HTML se fasse avec Hogan.js nous a contraints à rajouter des propriétés à l'objet questionnaire. En effet, Hogan.js ne peut gérer que des conditions binaires (booléennes). Comme l'affichage d'une question diffère selon son type, il a fallu rajouter des champs *isCheck*, *isRadio* et *isText* afin de savoir, par exemple, s'il faut afficher des boutons radio ou des cases à cocher (choix simple ou multiple).

3.2 Cookies

Nous avons décidé d'utiliser des cookies afin de simplifier l'implémentation mais aussi d'alléger la tâche du serveur. Ces cookies contiennent entre autres la session courante et les informations de l'utilisateur. Cela permet d'avoir un accès direct à ses nom et prénom, qui sont affichés sur toutes les pages, plutôt que de rechercher dans la base de données. Cela permet donc d'être plus efficace.