B3431 : Célia Djouadi et Mohamed Nour El-Jadiri + Gauthier Hembise

Analyse critique des livrables

Dans cette deuxième partie de projet nous avons reçu le livrable du Binôme 3231 constitué de Raoul El Mir et Sarah Pignol, décrivant l'application Instructif à implémenter.

I. Critique de l'IHM

Globalement le niveau de difficulté de l'implémentation de l'interface est très élevé étant donné les nombreux icônes présents. Pour pallier à cette difficulté nous avons utilisé la framework Bootstrap qui offre un ensemble de composants et de styles prédéfinis. Le dossier de spécification fournit aurait pu comprendre une indication nous incitant à utiliser ce genre de framework afin d'optimiser notre temps de travail et nous permettre de concentrer notre effort sur l'appel des services.

Néanmoins, malgré le recours à Bootstrap nous n'avons pas réussi à reproduire le layout exacte pour certaines pages. C'est le cas pour la page statistiques pour laquelle nos icônes ne correspondent pas tout à fait à ceux attendus (cf figure 1).

Figure 1 : comparaison de nos layouts et ceux demandés avec à gauche un exemple d'icône attendu et à droite notre implémentation du dit icône

Cette problématique nous amène à analyser le livrable qu'on a rendu pour l'application Prédictif qui en effet demande lui aussi un grand effort de développement esthétique et réactif pour certains affichages comme celui du menu déroulants listant les médiums. Nous aurions donc pu également prendre en compte cette difficulté et la solutionner en proposant l'utilisation de Bootstrap.

Aussi la fonctionnalité permettant de visualiser les statistiques des intervenants sur une carte de France nous semble compliquée à implémenter en css uniquement. C'est pourquoi nous avons trouvé plus interessant d'avoir recours à une API. Celle de google est payante et difficile à mettre en place, nous avons donc fait le choix d'utiliser OpenLayers. Là encore il aurait été judicieux de prendre en compte la difficulté liée à ce choix d'affichage afin d'en fournir un plus simpliste ou au moins nous proposer l'usage d'une API.

Retrospectivement, bien que nécessitant l'usage de frameworks, notre projet n'implique pas le recours à une API ce qui aurait ajouté de la complexité au projet. Globalement cette analyse nous permet de nous rendre compte de la difficulté qu'entraine certains de nos choix esthétiques sur le travail à fournir. En prenant davantage en compte le délai de rendu et aussi le niveau de compétences acquis en 3IF, nous aurions pu adapter la complexité esthétique de notre interface laissant ainsi plus de disponibilité pour la performance du projet.

Enfin, bien que difficile à implémenter, cette interface est très complète et offre un visuel professionnel et épuré ce qui nous incite à réaliser un travail de qualité. De plus le dossier de spécification détail bien les attendus notamment avec des ICARs très bien renseignés. Ainsi le choix a été fait de proposer de nombreuses fonctionnalités mais nous nous laissons le droit de les adapter/ simplifier. Concernant notre dossier de spécification Prédict'if, nous avons quelques fois souligné que certains affichages étaient optionnels tel que les liens du footer. Nous aurions pu d'avantage

spécifier ce qui était prioritaire afin de faciliter l'organisation de l'équipe chargée du développement.

II. Critique de l'application

Concernant l'aspect plus pratique de l'application, nous constatons une incohérence au sujet de la mise en relation élève - enseignant. En effet lorsque l'élève fait une demande, il est indiqué qu'il accède directement à la Visio tandis que d'après le dossier de spécification l'enseignant aurait une demande en attente. Pour répondre à cette problématique dans notre projet Prédictif nous avons veillé à ce que la demande et l'acceptation se fassent de manière synchrone. Pour ce faire notre consultation a un état « en attente », « en cours » ou « terminé » (cf figure 2). Ainsi leur choix d'affecter un coursActuel à un élève nous semble pas judicieux et nous recommandons plutôt d'affecter un état aux cours lui même et de prendre en compte l'état « en attente ».

Consultation demanderConsultation(Client client, Medium medium)	Le client peut demander une consultation avec un médium de son choix. La consultation est marquée *EN_ATTENTE* jusqu'à ce qu'un employé décide de la prendre.	Appel de la méthode *employeDao.findAvailableEmploye()* (cf. partie 5 pour plus de clarifications).
void accepterConsultation(Consultation consultation) .	Permet d'accepter une consultation en attente.	Envoie une notification à l'employé chargé de la consultation en paramètre puis change l'état de la consultation à *EN_COURS*. Change aussi l'état de l'employé à *INDISPONIBLE* dans la base de donnée (appel de employeDao.updateDisponibilite(disponibilite)).
void finConsultation(Consultation consultation, final String commentaire)	Met fin à la consultation en cours Associe un commentaire que l'employé renseigne à la consultation dans la base de données. Update le tuple correspondant sur la table Consultation.	Set l'état de la consultation en paramètre à *TERMINEE* et lui rajoute le commentaire passé en paramètre. Set l'état de l'employé charge de la consultation à *DISPONIBLE*. Tous les updates sont ensuite persistés.

Figure 2 : extrait de notre rapport prédictif avec mise en évidence des éléments que nous aurions aimé retrouver dans le document instructif

Un autre problème que nous avons rencontré lié aux services fournis concerne l'inscription d'un lycéen. En effet le service inscriptionEleve n'est pas fonctionnel dans le cas où l'on voudrait inscrire un étudiant de niveau lycée. En effet, l'étudiant ainsi créé n'avait pas d'établissement et n'était pas persisté ce qui mettait la base de donnée dans un état incohérent. Cette contrainte nous a amené à proposer un correctif prenant en compte le statut de lycéen en le persistant et en lui affectant un établissement .

Enfin dans un but d'optimisation nous aurions apprécié avoir un service spécifique chargé de lister les cours d'un élève. Ainsi lors de la connexion d'un client on pourrait faire appel à ce service afin d'accéder à l'information si l'élève a un enseignement en cours ou non. Or dans le cas présent en l'absence de services dédiés on doit avoir recours a des getters pour parcourir la liste des interventions et récupérer celles en cours. Enfin nous regrettons l'absence de commentaires dans les codes fournis ce qui nous fait perdre un temps précieux dans la compréhension et la prise en main de ceux-ci.

Il est néanmoins important de souligner que globalement tout les services nécessaires à la mise en place de l'interface telle qu'elle est demandée dans le dossier de spécifications sont bien présents et fonctionnels. De plus nous avons apprécié la mise en place des services d'initialisation qui ont grandement facilité le début de notre projet.

En conclusion, les livrables fournis par nos camarades nous ont permis de mener à bien l'implémentation de l'interface demandée bien que comprenant quelques incohérences que nous avons soulignées et solutionnées. Le travail effectué à partir de leurs livrables nous a permis de nous aussi prendre du recul et avoir un avis critique sur ceux que nous avons fournis pour implémenter l'application Prédictif.