07/01/2016

Pablo Gabolde, Amaury Petrini, Alexandre Decraene

G3S3

Rapport de programmation

CPOA

Mots de passes nécessaires :

Vous pouvez retrouver le site web à l’adresse suivante :

<http://iutdoua-webetu.univ-lyon1.fr/~p1406314/AppliWebCPOA/billeterie/controller/index.php>

Pour vous connecter à l’espace administrateur, il faut utiliser les logs suivants :

* Login : PETRINI
* Mdp : 0000

Eléments techniques

# Etat d’avancement

* **Pour la billetterie :**
* *Ce qui fonctionne* :
* Vérification du numéro de licence du client
* Gestion des erreurs (aucun numéro saisi, numéro incorrect)
* Achat de plusieurs places, choix des emplacements libres
* Saisi d’un code promotionnel, et nouveau calcul de prix
* Connexion d’un administrateur
* Gestion des erreurs (aucun login ou mot de passe saisi, mot de passe incorrect)
* Affichage des billets vendu, ainsi que leur prix, leur emplacement, le match associé
* Possibilité d’ajouter ou de supprimer un billet
* Affichage des promotions, avec leur code associé et leur pourcentage de réduction
* Possibilité d’ajouter ou de supprimer un code promotionnel.
* *Ce qui ne fonctionne pas :*
* Pas d’achat plusieurs place à la fois, seulement une par une
* Pas de promotion pour un client licencié
* **Pour le planning des matchs :**
* *Ce qui fonctionne :*
* *Ce qui ne fonctionne pas :*

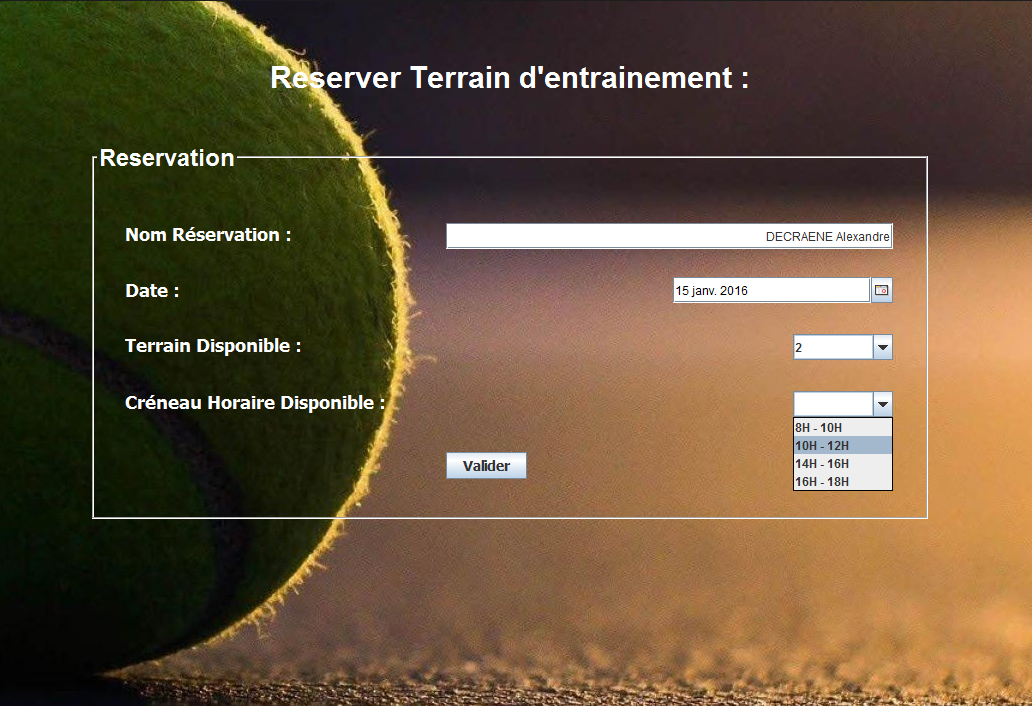
# Description de l’architecture

Notre méthode utilisée pour coder a été le développement itératif incrémental d’UML car nous avons suivi nos diagrammes et notre IHM bien qu’il ait fallu les modifier et les repenser pendant le développement.

Ensuite, nous avons organisé (au départ) le projet (WEB et JAVA) à l’aide des couches MVC. Suite au manque de temps, nous avons délaissé la couche « Model » pour parvenir à un MVC « personnalisé ». En effet, il y avait les contrôleurs et les vues.

# Retour à l’analyse

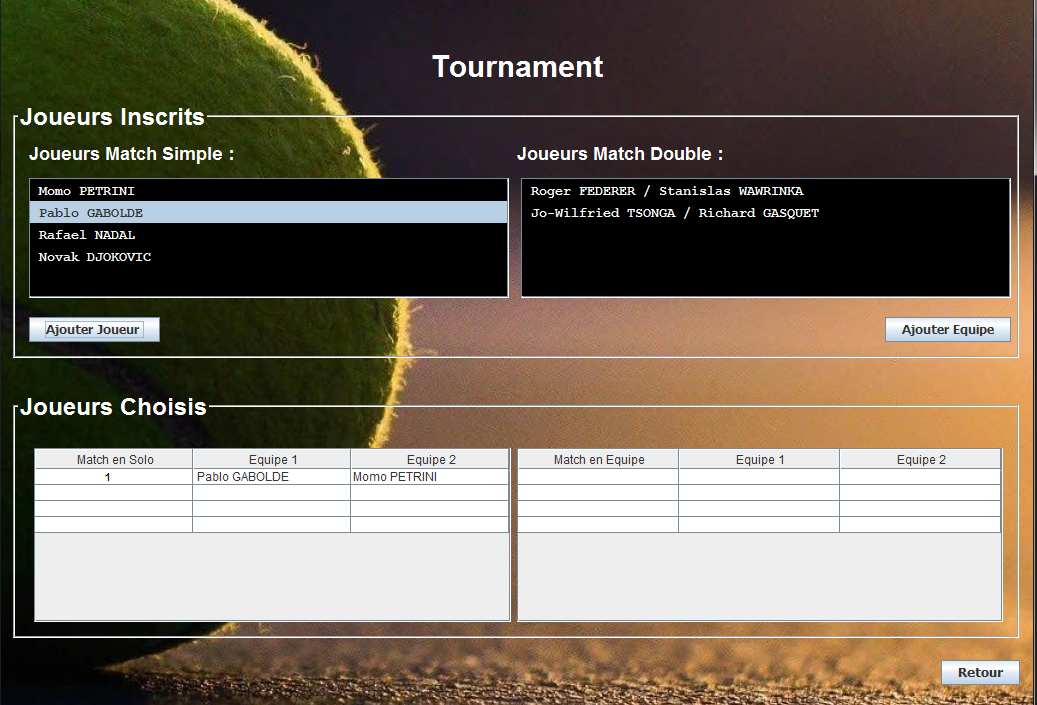
  
Nous avons utilisé ce diagramme de classe afin de créer une base de données (MySQL) qui soit cohérente avec notre projet. Nous avons tout de même dû modifier quelques éléments car notre diagramme était au final incomplet.



Dans cet exemple, on récupère les différentes dates, les terrains possibles et les créneaux horaires dans la base de données à l’aide de la classe Match.

*Un autre exemple de l’interface JAVA :*

*Ici on à l’interface qui permet de Gérer le premier tour du tournois, elle n’est pas fonctionnelle à 100%.*





Pour donner un exemple pour le système de billetterie : nous avons utilisé la classe « Match » pour récupérer la liste des matchs dans la base de donnée, pour ensuite l’afficher. Cela sert à celui qui veut acheter une place de sélectionner le bon, et éventuellement de voir les détails de celui-ci :



# Points d’amélioration

1. Tout d’abord, pour la billetterie, on pourrait rediriger le client vers un site de paiement sécurisé une fois ses achats terminés, avec un mail de confirmation et une copie des billets.
2. De plus, au lieu de vérifié le numéro de licence par rapport à une base de données, on pourrait vérifier par rapport au fichier national des licences.
3. Au moment du choix de l’emplacement des places, on aurait pu ajouter une carte des tribunes interactive, comme sur le vrai site de l’open sud de France.
4. On aurait également pu ajouter des promotions pour les étudiants, avec vérification, ou pour les personnes à mobilités réduites avec un choix adéquat des places.
5. La génération des places automatiques, c’est-à-dire que l’administrateur n’a pas besoin de les créer tous manuellement.
6. La génération du premier tour du planning des matchs automatique.

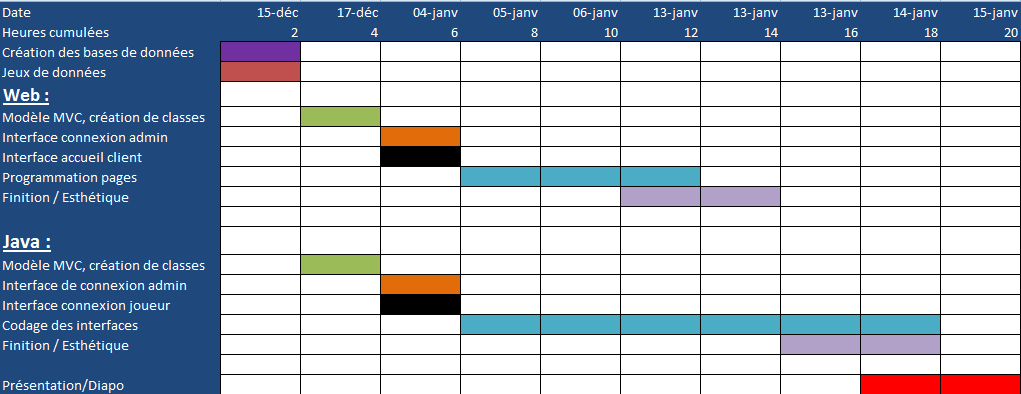
# Rétro-conception

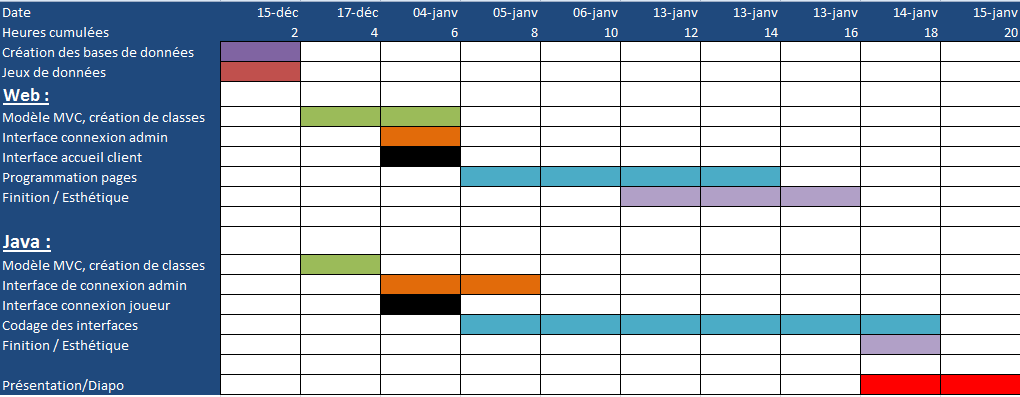
Notre phase d’analyse n’était pas assez poussée. En effet, au fur et à mesure de l’avancement de notre projet, nous avons dû revoir nos IHM ainsi que nos diagrammes, car ceux-ci n’étaient pas adaptés. En revanche, les diagrammes de cas d’utilisation nous ont aidés pour cerner les besoins des utilisateurs.

Au niveau développement, nous avons tout d’abord eu une phase de réflexion entre nous trois afin de discerner les différentes fonctions de l’application. Ensuite, nous avons eu une phase de pratique où nous avons codé à la fois de façon individuelle et collective puisque nous savions quoi faire même s’il a fallu se mettre d’accord sur certains points d’ombre.

Eléments de gestion de projet

1. Planning

Planning prévisionnel :

Planning réel :

Répartition des tâches

Amaury et Pablo se sont mis d’accord pendant toute la durée du codage pour savoir comment réaliser le site, avec quelle fonctionnalités, tout en restant en contact avec Alexandre qui développe la partie java pour que ce soit en harmonie.

Pablo s’est occupé de la partie client, qui concerne principalement l’achat des billets, tandis qu’Amaury a réalisé la partie administrateur, qui concerne la gestion des billets et des codes promotionnels.

Lorsque le projet Web était finalisé, Amaury et Pablo ont terminé l’application JAVA avec Alexandre.

Nous avons également créé ensemble la base de données commune à l’application et au site Web pour être en cohérence.

Afin de se mettre d’accord sur nos deux applications et que tout soit cohérent il y a eu un échange permanent entre les 3 membres du groupe.

1. Système de gestion de version et test

En ce qui concerne le CVS, nous avons utilisé Dropbox, qui a été très utile pour partager notre travail, en revanche, nous reconnaissons que l’utilisation de GIT aurait été plus approprié pour travailler en collaboration.

Enfin, en ce qui concerne les tests, nous avons fait tester les applications (WEB et JAVA) à 10 personnes de notre entourage. Cela nous a permis d’améliorer l’ergonomie et l’efficacité de celles-ci.

1. Bilan personnel

Ce système de répartition des tâches nous a fait penser à ce que l’on peut voir dans une entreprise, ce qui pour nous a été très enrichissant car nous avons appris beaucoup de nouvelles choses aussi bien en terme de gestion du temps qu’en terme de notions.

Si cela était à refaire nous nous impliquerions beaucoup plus sur l’analyse pour ainsi ne pas perdre de temps dans le développement et ainsi optimiser notre temps. De plus, il est clair qu’il aurait fallu se mettre à plusieurs sur l’application JAVA dès le début car celle-ci était la partie la plus concise du projet.

Au final, nous avons déterminé une technique d’analyse propre à nous : il s’agirait de faire les maquettes de l’IHM de notre côté, pour ainsi faire une analyse UML plus complète que l’on proposera au client. Celle-ci s’avère plus flexible pour le client plutôt qu’une analyse IHM qui figerait les choses.