



Rapport Final de Projet

Réalisé par

**MATTHIEU GRIMAUD – ALEXIS NIZARD - ADRIEN PAPIN- BENJAMIN
SERVA**

Introduction:

L'objet de ce rapport est de présenter synthétiquement le travail réalisé lors de notre projet informatique. Nous avons pour objectif de réaliser un site web permettant de gérer des tournois. Nous avons très naturellement choisi de nous orienter vers les tournois sportifs, certains de nous ayant un intérêt plus important pour le sport qu'un autre domaine.

Ce projet informatique avait pour objectif la réalisation d'un site web permettant de gérer des tournois. Comme certains d'entre nous avaient un intérêt plus important pour le sport qu'un autre domaine, nous avons très naturellement choisi de nous orienter vers les tournois sportifs.

Ce site a été pensé et construit avec l'idée d'offrir un maximum de flexibilité à l'utilisateur, en lui permettant par exemple d'avoir plusieurs équipes, de participer à plusieurs tournois en même temps ou encore d'avoir le même nom qu'un autre utilisateur. Il fallait s'assurer que celui-ci ne soit pas limité dans ses possibilités. sans pour autant que cela pose problème dans la conception technique du site.

Nous avons également pour volonté d'avoir un site ayant une navigation aussi intuitive que possible. Pour permettre cela, nous avons cherché à réduire au maximum le nombre de clics nécessaires à son fonctionnement mais aussi, dans la mesure du possible, à réduire la latence que pouvaient engendrer certaines fonctionnalités du site.

Dans la suite de ce rapport, nous vous expliquerons dans un premier temps comment l'organisation du travail s'est faite au sein du groupe. Puis, pour vous permettre d'avoir une vision abstraite du site, nous vous exposerons son architecture. Après cela, la base de données sera étudiée en détail et, pour finir, nous nous pencherons sur quelques détails techniques du site important à nos yeux.

La suite de ce rapport expliquera dans un premier temps l'organisation du travail au sein du groupe. Puis, afin d'obtenir une vision abstraite du site, son architecture sera exposée. Après cela, la base de données sera étudiée en détail. Enfin, quelques détails techniques importants à nos yeux seront expliqués.

Pour finir cette introduction, nous tenons à vous faire savoir qu'une visualisation de notre site est possible à l'adresse suivante :

<https://tournoiabam.000webhostapp.com/>

Un compte “admin” et un compte “membre” y sont également à votre disposition :

relecteur@umontpellier.fr mdp000

mortier.cris@etu.umontpellier.fr mdp362

Gestion de groupe:



































Démarrer l'architecture du site	 matthieu.grimaud@etu.umontpellier.fr		10 – 17 fév
Modéliser et implémenter la BDD	 alexis.nizard@etu.umontpellier.fr		10 – 17 fév
Créer son compte	 adrien.papin@etu.umontpellier.fr		17 fév – 3 mar
Gestion de la connexion avec compte	 benjamin		17 fév – 3 mar
synchroniser le travail précédent	 benjamin		3 – 17 mar
partie paramètres	 matthieu.grimaud@etu.umontpellier.fr		3 – 17 mar
affichage des différents tournois	 adrien.papin@etu.umontpellier.fr		3 – 17 mar
avancement de l'affichage du tournoi pour le gestionnaire		tout le monde	17 – 31 mar
rectification css sur le site (amélioration design)	 adrien.papin@etu.umontpellier.fr		31 mar – 7 avr
saisit des matchs/date/vainqueur	 alexis.nizard@etu.umontpellier.fr		31 mar – 7 avr
création équipe et gestion/inscription à un tournoi	 matthieu.grimaud@etu.umontpellier.fr		31 mar – 14 avr
réduction du nombre de cliques dans le site		tout le monde	14 – 21 avr
gestion modérateur (admin/joueur)/bdd à modifié	 matthieu.grimaud@etu.umontpellier.fr		21 – 28 avr
finition de la gestion tournoi	 alexis.nizard@etu.umontpellier.fr		29 avr
gestion des pré-inscriptions	 benjamin		29 avr
ajout des likes	 adrien.papin@etu.umontpellier.fr		29 avr
création de la page profil	 matthieu.grimaud@etu.umontpellier.fr		29 avr
compléter la page profil		tout le monde	30 avr
Démarrer l'architecture du site	 matthieu.grimaud@etu.umontpellier.fr		10 – 17 fév
Modéliser et implémenter la BDD	 alexis.nizard@etu.umontpellier.fr		10 – 17 fév

Diagramme de Gantt pour l'organisation du travail

Concernant l'organisation, au début du projet nous voyons deux parties importantes. D'une part la conception et la réalisation de la base de données et d'autre part la fabrication d'un squelette en HTML/CSS afin d'avoir une première version du site.

Nous avons d'abord décidé de diviser les quatres membres en deux équipes. Alexis et Adrien étaient chargés de mettre en place la base de données (de sa conception à son implémentation). Pour ce faire, ils ont d'abord travaillé sur sa modélisation en réalisant un diagramme. Cela nous a permis d'en discuter avec notre encadrant. Enfin, après que la modélisation ait été validée par tous, son implémentation a pu être effectuée. En parallèle, Benjamin et Matthieu avaient eux pour tâche de mettre en place le site (HTML et CSS) avec les éléments de base d'un site web (titre, bannière, footer, premiers boutons nécessaires...).

Une fois ces tâches accomplies de part et d'autre, c'est lors des réunions hebdomadaires prévues entre les membres du groupe et des réunions bihebdomadaires avec notre tuteur que nous avons planifié les tâches suivantes à réaliser. Nous les répartissions de telle sorte à ce que chaque membre ait quelque chose à faire. À noter qu'ils nous arriveraient aussi régulièrement de travailler ensemble sur une même tâche.

Durant tout le projet, Alexis et Adrien ont surtout travaillé sur les fonctionnalités d'admin et de gestionnaire, notamment sur la partie gestion des tournois et son avancement. Tandis que Benjamin et Matthieu se sont concentrés sur la partie simple joueur et modérateur, comme la création et la gestion de son équipe.

Outils utilisés:

Concernant les outils utilisés pour la communication entre les membres du groupe, nous avons mis en place un serveur Discord pour discuter des tâches et de l'avancement du projet en général, mais aussi pour que chacun puisse discuter des différents problèmes et/ou solutions lorsqu'il y en avait. Les réunions entre nous avaient également lieu sur Discord.

Les réunions avec notre tuteur avaient lieu quant à elles toutes les 2 semaines sur BigBlueButton.

Afin de partager notre code, après avoir discuté des différentes alternatives possibles pour cela, nous avons choisi d'utiliser Gitlab pour stocker le code et le rendre consultable entre chaque membre du groupe.

Nous avons également utilisé Repl.it, un éditeur de code en ligne qui nous a été très utile lorsque l'on souhaitait travailler ensemble en temps réel sur un même morceau de code.

Architecture du site :

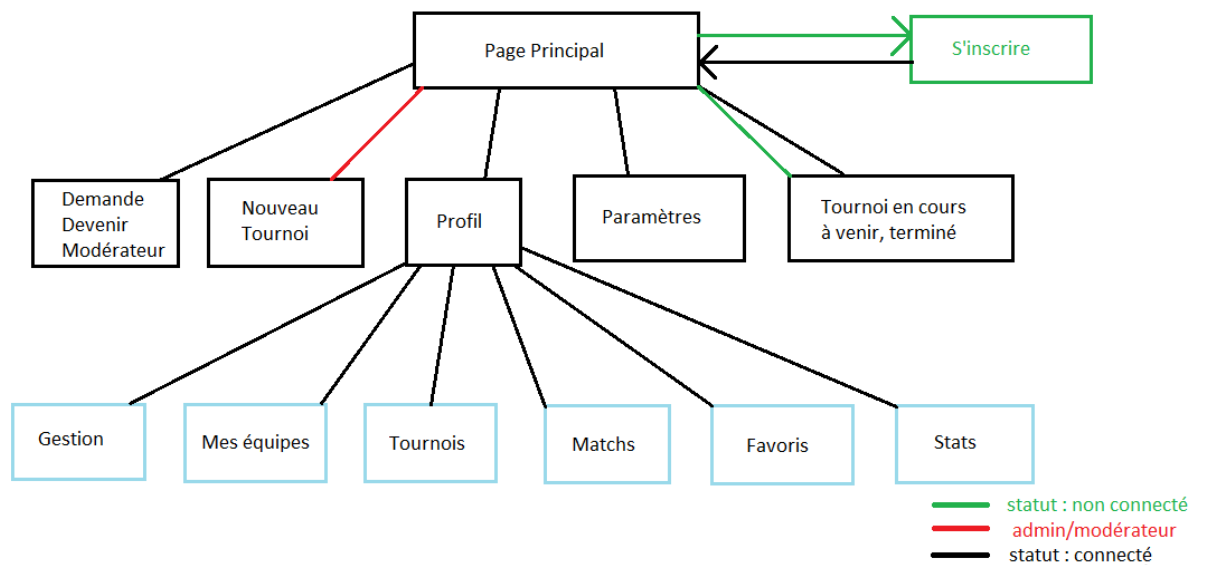


Schéma représentant la structure du site

Avant d'aborder l'architecture du site, il semble essentiel de mentionner les différents droits qu'un utilisateur peut avoir sur le site. Ces droits sont les suivants :

- **Admin:** L'utilisateur a le contrôle sur tout mais n'a pas la possibilité de créer son équipe et s'inscrire à un tournoi.
- **Modérateur:** L'utilisateur a les mêmes droits qu'un membre mais aura la possibilité supplémentaire de créer des tournois.
- **Membre:** L'utilisateur gère ses équipes et ses inscriptions aux tournois.
- **Gestionnaire:** Ce n'est pas réellement un droit. Un utilisateur peut être amené à être gestionnaire c'est-à-dire gérer l'avancement d'un tournoi. Il dispose également de quelques fonctionnalités supplémentaires, comme d'un bouton lui permettant d'imprimer son tournoi, la possibilité de changer la couleur des cases de son tournoi ou encore de boutons lui permettant de gérer plus simplement son tournoi (remplir aléatoirement les premières rencontres, remise à zéro, affectation de la même date et lieu à toutes les cases).

Tout compte peut être gestionnaire.

Pour commencer, toute personne souhaitant accéder au site [sera redirigée](#) sur la page d'accueil qu'il soit connecté ou non.

Si un utilisateur souhaite naviguer sur le site sans être connecté, il pourra seulement accéder aux informations des différents tournois présents sur la page d'accueil (qu'ils soient en cours, à venir ou terminés).

S'il s'inscrit ou se connecte, il aura alors le statut connecté et aura la possibilité d'interagir avec les différentes fonctionnalités suivantes :

Tous les utilisateurs, et peu importe le droit qu'ils possèdent, ont la possibilité d'accéder à leurs paramètres et à leur profil.

Lorsqu'un utilisateur clique sur son profil, il verra apparaître un menu composé de différents onglets.

Ci-dessous les différents onglets :



Représentation des différents onglets de la page profil

1. **Gestionnaire:** Si l'utilisateur est gestionnaire d'un ou plusieurs tournois, alors ces derniers sont répertoriés dans cet onglet.
L'utilisateur peut trier ses tournois par ordre alphabétique, dates, types de tournois.

2. **Mes équipes:** Sont affichées les équipes dont l'utilisateur est le capitaine, le cas échéant, la liste des joueurs pour chacune de ses équipes. Mais également un emplacement supplémentaire pour créer d'autres équipes s'il le souhaite.
3. **Tournois:** Sont affichés les tournois auxquels l'utilisateur s'est inscrit (seul ou en équipe).
A l'instar de l'onglet Gestionnaire, l'utilisateur peut trier ses tournois par ordre alphabétique, **dates**, types de tournois.
4. **Matches:** Est affiché l'historique des matchs réalisés par les différentes équipes de l'utilisateur mais également les matchs à venir.
5. **Mes favoris:** Sont affichés les tournois que l'utilisateur a liké.
Une nouvelle fois, l'utilisateur peut trier ses tournois par ordre alphabétique, date, types de tournois.
6. **Mes statistiques:** Sont affichées les statistiques d'un utilisateur, ses ratios de victoires, défaites.



- 1) La partie paramètres permet de consulter l'ensemble de ses informations personnelles et de les modifier si nécessaire.
- 2) De plus, un admin possède une fonctionnalité supplémentaire qui est celle de créer des tournois

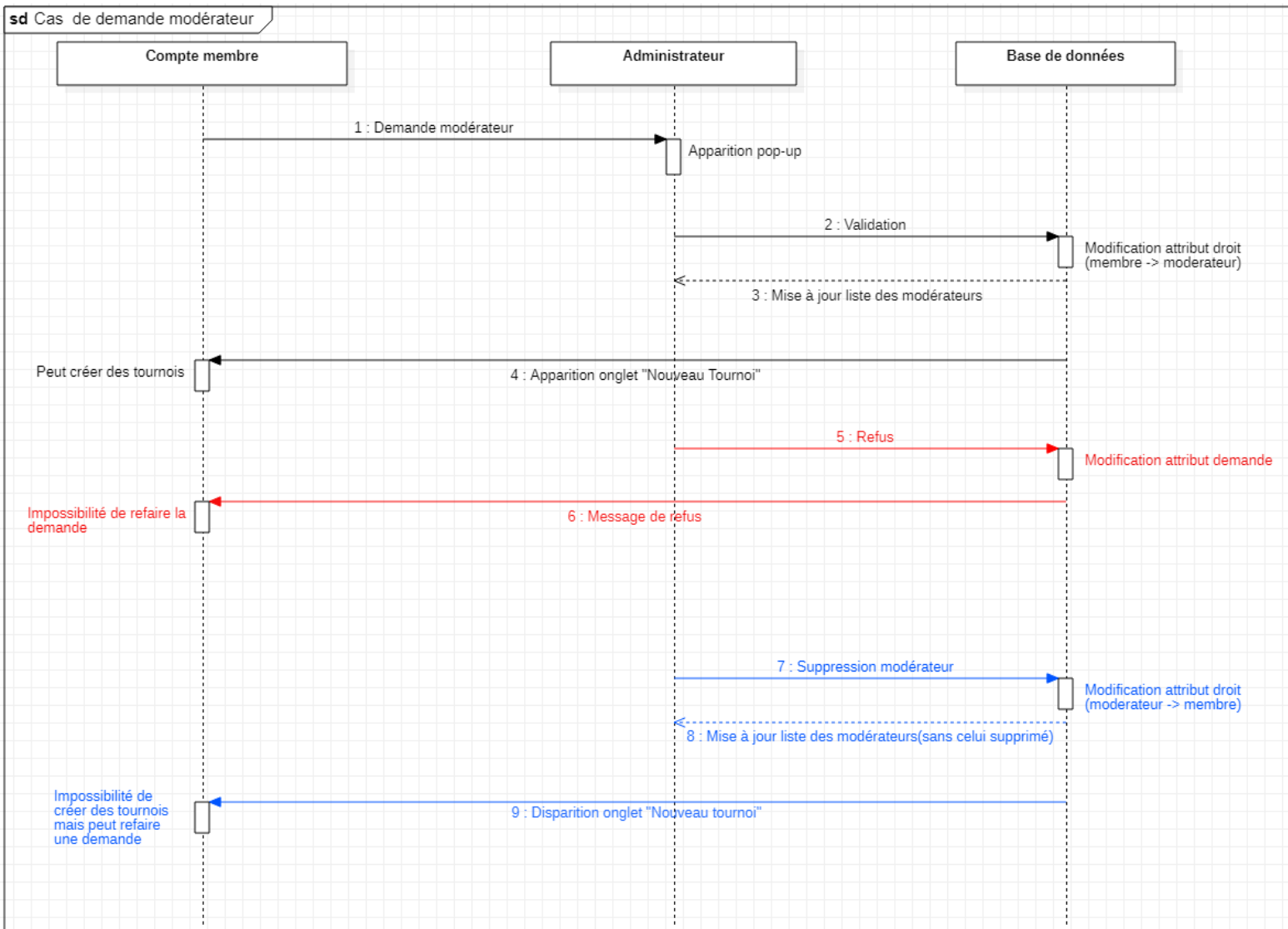
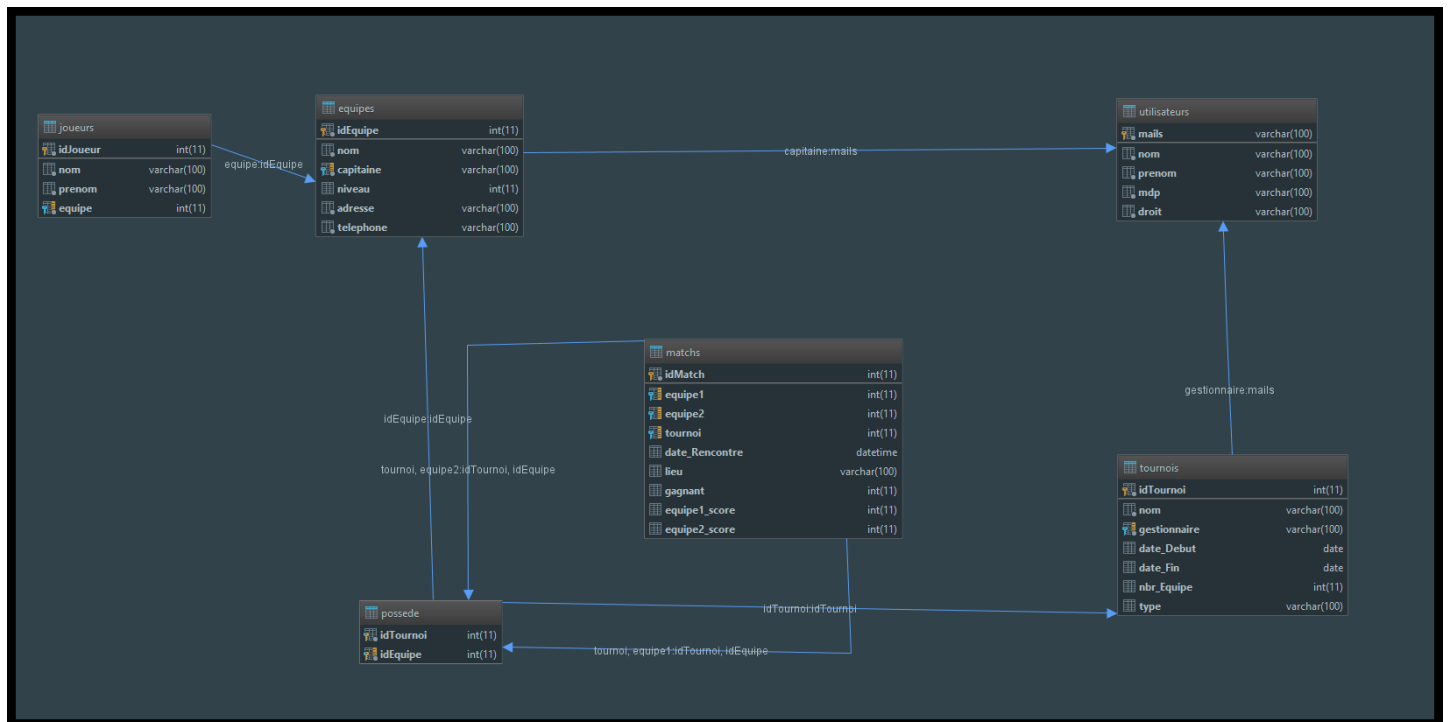


Diagramme de séquence représentant le cas d'une demande de modérateur

Les comptes lambda, une fois connectés, ont la possibilité de faire des demandes pour pouvoir également créer des tournois (être modérateur). S'ils sont acceptés, l'onglet de création de tournoi apparaîtra sur leur compte.

L'admin peut gérer ces différentes demandes sur son profil par le biais d'un onglet "Consulter les demandes" que seul ce droit possède.

Base de données



Base de donnée de départ

Ce diagramme ci-dessus représente la base de données que nous avons conceptualisée et décidé d'implémenter au début du projet.

Cette base contenait 6 tables, chacune reliées entre elles par différentes clés primaires et étrangères.

La première table était la table *utilisateurs*. Celle-ci possédait comme clé primaire un mail. Les autres attributs sont le nom, le prénom, téléphone, adresse, droit et mdp d'un utilisateur.

Cette table a été créée pour stocker les différentes données d'un utilisateur.

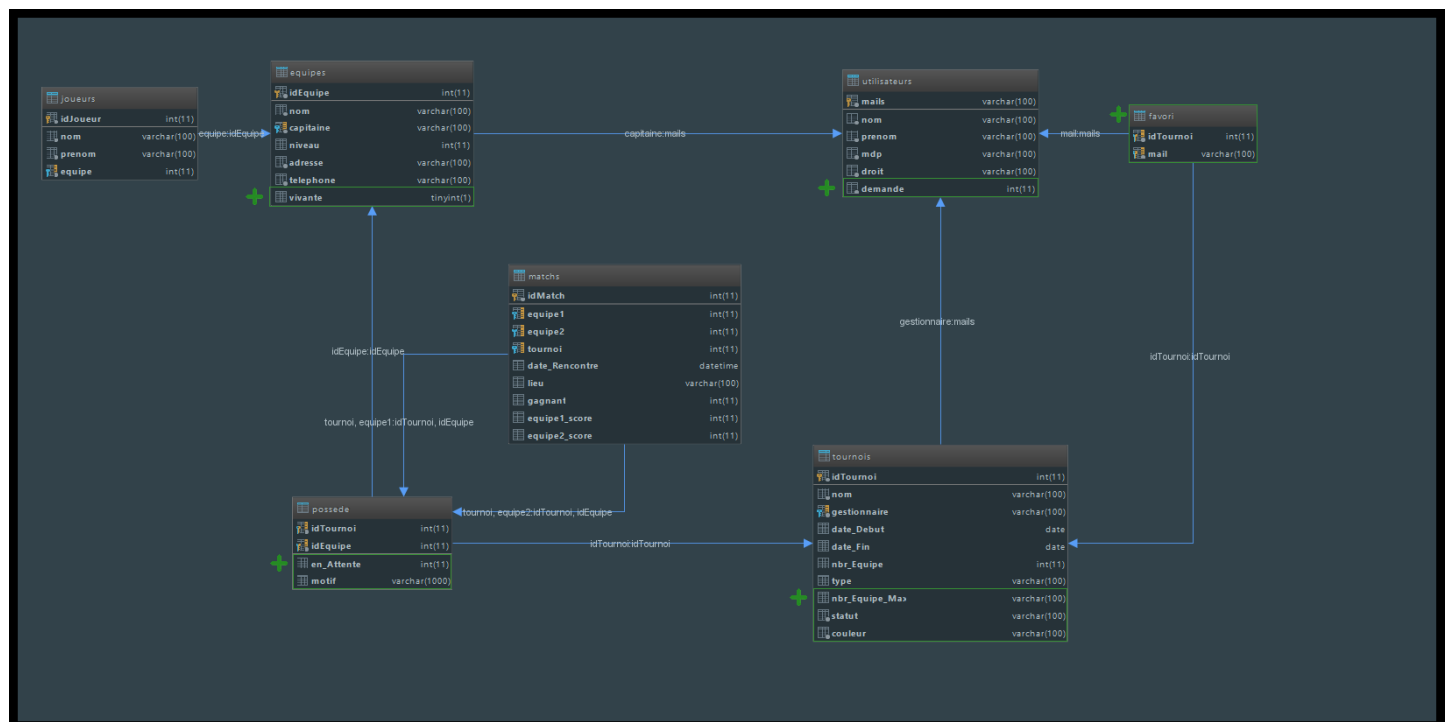
La deuxième table était la table *tournois*. Celle-ci possédait comme clé primaire l'id d'un tournoi. Les autres attributs étaient le nom du tournoi, son gestionnaire, la date de début, la date de fin du tournoi, le nombre d'équipes et le type de tournoi (Coupe, Championnat...)

La troisième table *equipes*. Celle-ci permettait de stocker les informations d'une équipe. La clé primaire est un id. Les autres attributs sont le nom de l'équipe, son capitaine (mail du joueur, clé primaire de la table *utilisateurs*), le niveau de l'équipe, son adresse et son téléphone.

La table suivante *joueur* qui permet de stocker un joueur et dont la clé primaire est un idJoueur. On a ensuite le nom et prénom du joueur et son équipe (qui est une clé étrangère faisant référence à l'*idEquipe* présent dans la table Equipes).

On a également la table *possède* qui permet d'associer la table équipe et tournois, quand celle-ci est inscrite dans un tournoi et dont les deux attributs sont l'idTournoi et idEquipe (deux clés étrangères).

Et enfin la table *match*, qui permet de stocker les matchs pour chaque tournoi. Celle-ci possède comme clé primaire un idMatch. On ensuite comme attributs l'équipe1, l'équipe2, le tournoi concerné, la date, le lieu le gagnant du match, le score pour la première équipe, le score de chaque équipe.



Base de donnée finale

Au fur et à mesure du projet, nous avons dû ajouter des attributs dans certaines tables de la base voire de nouvelles tables lors de l'implémentation de fonctionnalités avancées pour que celles-ci puissent fonctionner. Voici les différentes données que nous avons pu ajouter.

La table *favori* qui permet de gérer et stocker les différents tournois likés par un utilisateur. Cette table fait le lien avec la table *tournois* par l'id du tournoi et utilisateurs par les *mails*. Les attributs sont des clés étrangères.

L'attribut *demande* situé dans la table *utilisateurs*. Cet attribut permet de traiter le cas des demandes d'un simple membre pour pouvoir créer des tournois (modérateur). L'attribut prendra la valeur 0 si aucune demande est faite, 1 si une demande est réalisée par un membre, -1 si la demande est refusée par un admin.

L'attribut *vivante* dans la table *équipe*. C'est un booléen nécessaire dans le cas où une personne supprime son équipe et que l'on veut garder les infos des équipes qui ont participé à un tournoi terminé.

L'attribut *en_Attente*. Cet attribut a été ajouté dans la table *possede* afin de pouvoir gérer le système de pré-inscriptions c'est-à-dire lorsque qu'un capitaine d'équipe souhaite pré-inscrire son équipe à un tournoi donné. L'attribut prendra la valeur 1 tant que l'équipe est en attente. 0 si le gestionnaire accepte l'équipe, -1 s'il la refuse. **dans possede, utile pour traiter le cas des pré-inscriptions.**

L'attribut *motif* a également été ajouté dans la table *possede* afin de stocker le message de refus dans le cas où un gestionnaire refuse l'équipe.

L'attribut *couleur* dans la table *tournois*, permettant au gestionnaire de modifier l'esthétique de son tournoi en changeant la couleur des cases contenant les différentes rencontres.

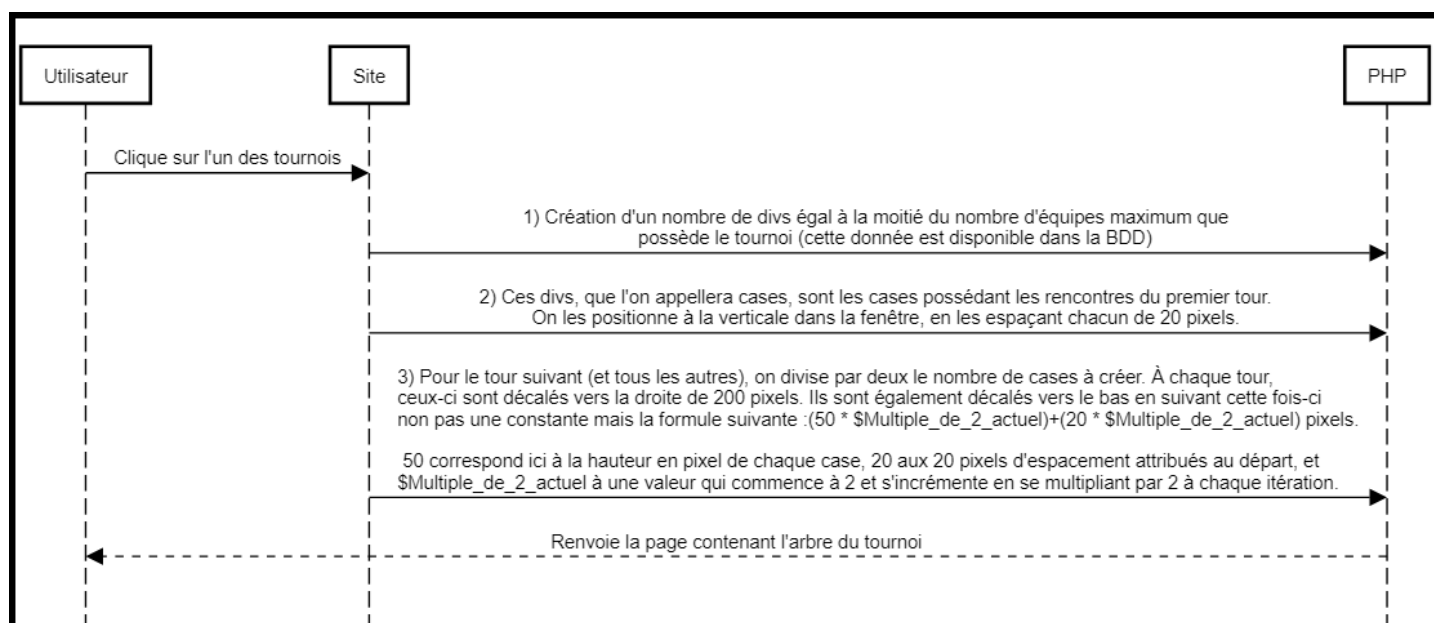
Et enfin, un attribut nommé *statuts* répertoriant le statut des tournois. Les différentes valeurs sont donc en cours, à venir, terminé.

Particularités techniques de notre site :

- **L'utilisation de PHP** pour générer l'arbre du tournoi

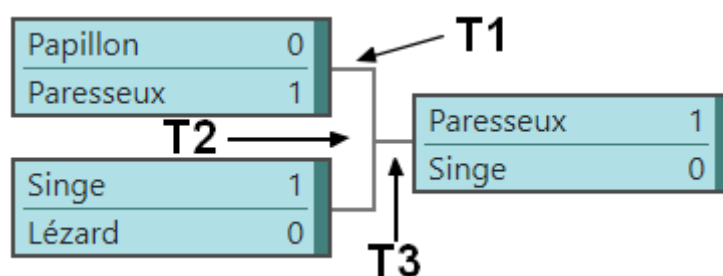
Cette partie du projet ne fut pas une mince affaire, et à vrai dire nous avons dû nous y prendre à plusieurs reprises avant d'arriver à un résultat concluant.

Voici un diagramme de séquence expliquant le processus

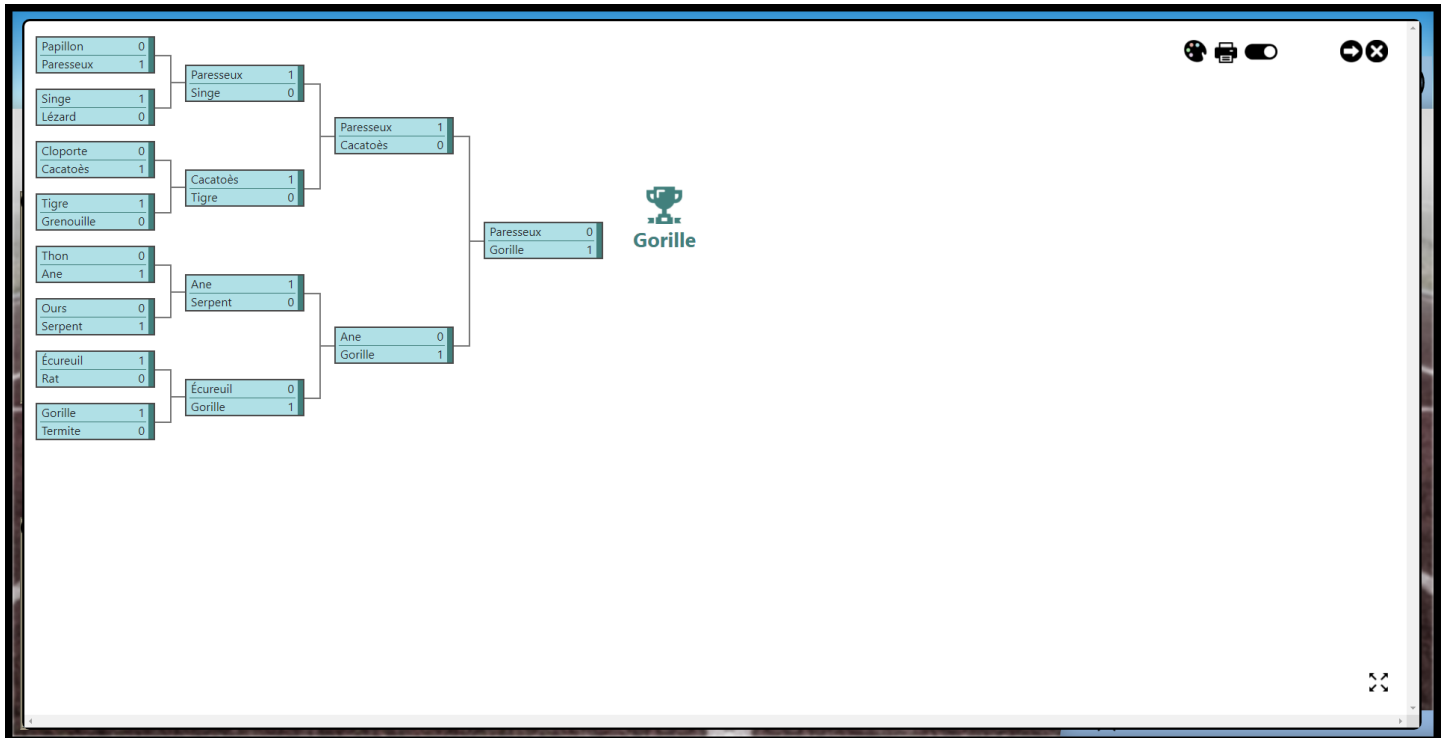


Les cases sont également reliées entre elles par des branches. Pour les afficher, le principe reste le même : à l'extrémité de toutes les cases, sauf la dernière, un trait, que l'on appellera T1, de 25 pixels de longueur est ajouté. Deux par deux, les cases sont alors reliées par un trait horizontal, T2, partant de l'extrémité du trait T1. La taille de ce nouveau trait augmente constamment à chaque nouveau tour. Il faudra enfin créer un dernier trait, T3 (identique au trait T1) qui partira cette fois du milieu de chaque trait T2 créés.

Les différents traits servant à créer les branches du tournoi



Arbre d'un tournoi à 16 équipes



- **L'utilisation de jQuery** à deux reprises nous a servi à embellir notre site, en permettant :
 - D'une part, la création d'un carrousel d'images dans le header.
 - D'autre part, de permettre l'affichage de diverses graphiques dans l'onglet "Mes statistiques" à l'intérieur de la page "Mon profil".
 - La création d'un carrousel d'images dans le header permettant de faire arriver les images avec une plus grande vélocité, en plus de les faire défiler de façon récursive. Choses que nous n'arrivons pas à reproduire avec Javascript uniquement.
 - D'autre part, de permettre l'affichage de divers graphiques dans l'onglet "Mes statistiques" à l'intérieur de la page "Mon profil".

Note : Utiliser jQuery pour le carrousel nous a permis de faire arriver les images avec une plus grande vélocité, en plus de les faire défiler de façon récursive. Choses que nous n'arrivons pas à reproduire avec Javascript uniquement.

Nous avons vite vu les limites de PHP en travaillant sur la création du site web. En effet, avec PHP, chaque fois que l'utilisateur souhaite interagir avec le site, cela nécessite un rafraichissement de la page.

Pour faire face à cela, nous avons eu recours à différentes techniques :

- **L'utilisation de Javascript** pour modifier les propriétés Css d'un objet sans rafraichir la page :

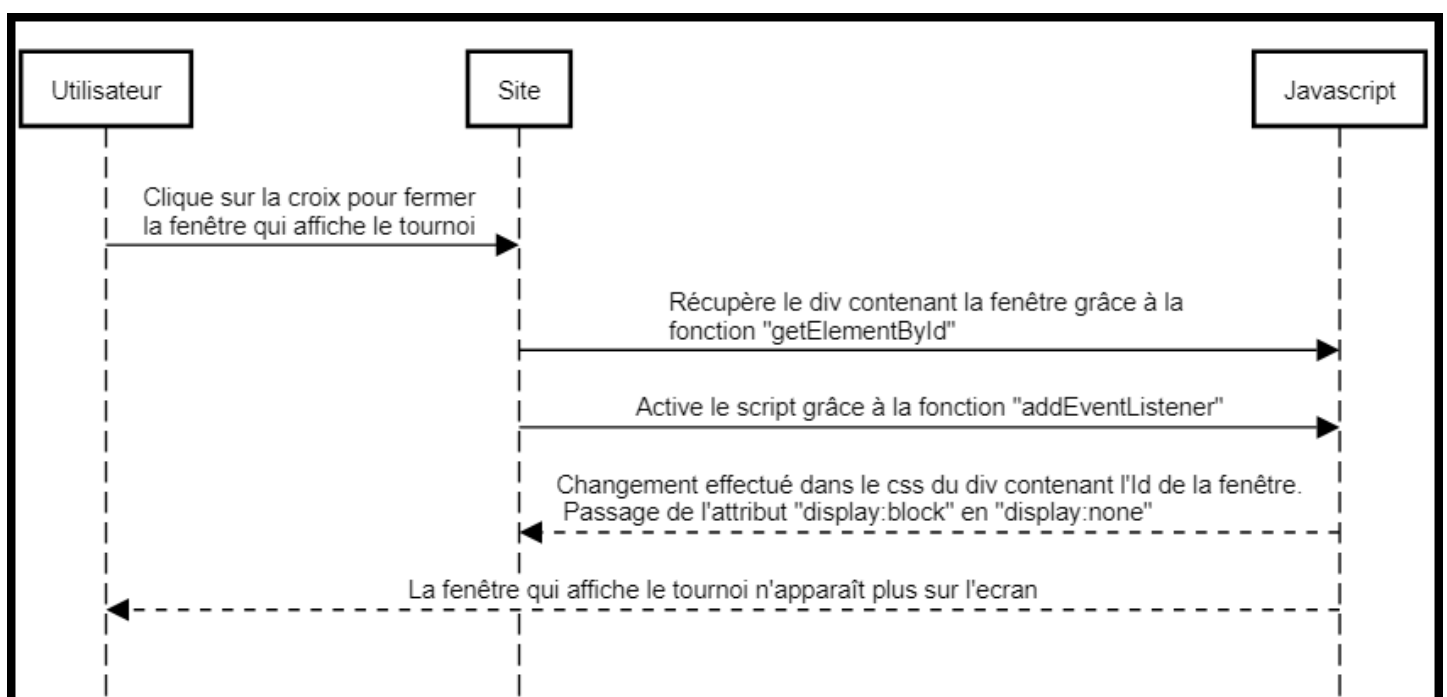
Nous avons vite vu les limites de PHP en travaillant sur la création du site web. En effet, avec PHP, chaque fois que l'utilisateur souhaite interagir avec le site, cela nécessite un rafraichissement de la page.

Pour faire face à cela, nous avons eu recours à différentes techniques :

L'utilisation de Javascript, qui fut principalement utilisé pour faire apparaître et disparaître une fenêtre de l'écran. Voici quelques exemples concrets d'utilisation :

- Faire apparaître et disparaître la fenêtre permettant de créer un tournoi.
- Naviguer entre la page 1 (qui affiche le tournoi) et la page 2 (qui affiche les informations relatives au tournoi)
- Fermer la fenêtre qui s'ouvre lorsqu'on clique sur un tournoi
- Masquer toutes les icônes d'éditations qui s'affichent lorsque l'on est gestionnaire

Voici un diagramme de séquence expliquant le déroulement de l'opération



Nous retrouvons ce même principe avec le bouton agrandir / réduire la taille de la fenêtre qui apparaît lorsque l'on clique sur un tournoi, à la différence qu'au lieu de modifier l'attribut "display", ce sont les "width" et "height" qui sont modifiés.

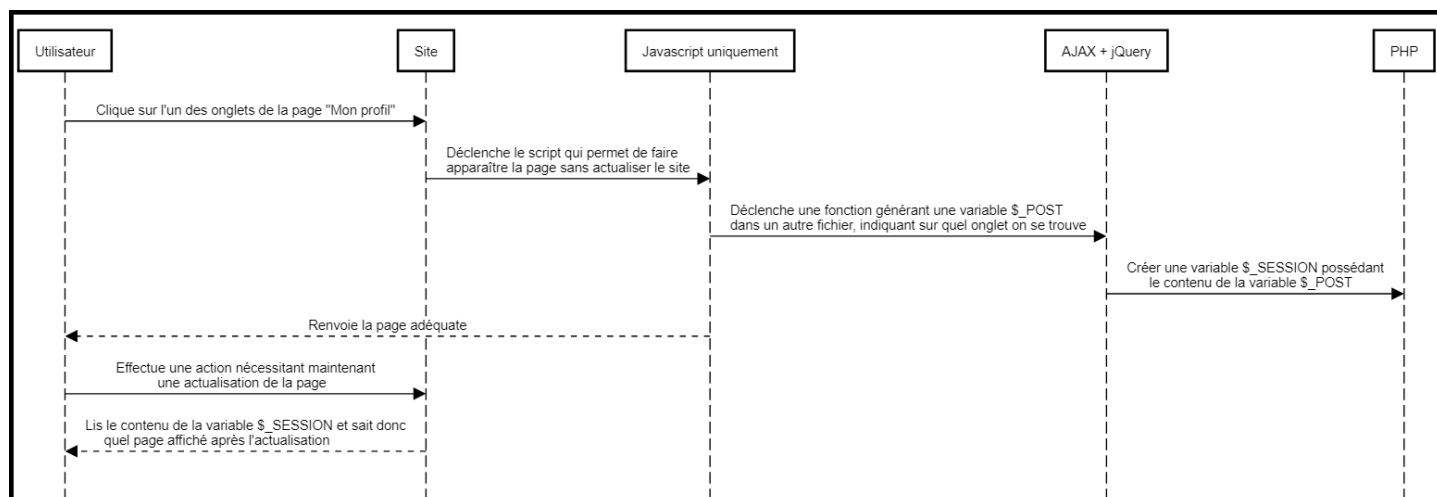
- **L'utilisation de l'AJAX avec jQuery** pour échanger et récupérer des données sans rafraichir la page :

Javascript **seul** nous a été d'une grande aide pour la création du site, mais une fois encore, son utilisation avait ses limites. En effet, une fois que la page était actualisée, le code revenait à son état initial et les modifications faites disparaissaient.

Pour faire face à cela, nous avons utilisé l'AJAX à deux reprises:

- **Premièrement**, Lorsque le gestionnaire saisit une rencontre, **il** déclare un gagnant ou affecte une date et un lieu à un match, la page est rafraîchie, il fallait donc réfléchir au cas de figure suivant :
Si l'utilisateur a cliqué sur le bouton permettant d'agrandir la fenêtre, alors, après un rafraichissement de la page, la fenêtre devra conserver ses dimensions.
- **Deuxièmement**, Lorsque l'utilisateur navigue entre les différents onglets de son profil, il n'y a pas de rafraichissement de pages effectué. Or, si celui-ci souhaite interagir avec un des onglets, dans la plupart des cas, une actualisation sera nécessaire. Il fallait donc faire en sorte que même après cette actualisation, il puisse rester sur le dernier onglet cliqué.

Voici un diagramme de séquence montrant ce qui se passe pour le cas de l'onglet



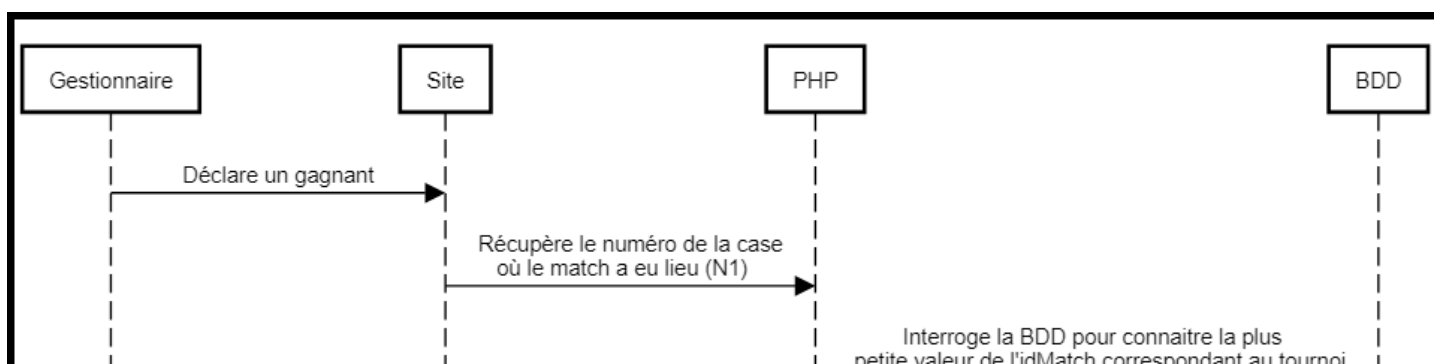
Pour conserver les dimensions de la fenêtre après l’actualisation, c’est le même principe. La variable `$_SESSION` indiquera cette fois-ci si la fenêtre a été agrandie ou non.

• Réduction du délai générées par une requête SQL

Nous avons également remarqué que le temps d’attente avant l’arrivée d’une nouvelle page était plus long lorsque la base de données était impliquée. Pour comprendre comment nous avons optimisé notre site de telle sorte à ce que cette latence soit le moins visible possible, il faut d’abord connaître comment l’avancée d’un tournoi fonctionne :

Dans un tournoi, chaque case créée pour représenter un match possède un numéro qui s’incrémente de 1 à chaque fois. Dans la BDD, la table `matches` quant à elle, possède une colonne `idMatch` contenant un int qui s’auto incrémente à chaque nouvelle ligne ajoutée.

Prenons maintenant le cas où un gestionnaire de tournoi déclare le gagnant d’une rencontre :



*En numérotant les cases de chaque match, à la façon d'un arbre binaire représenté avec un tableau, nous nous retrouvons avec une relation entre les cases pères et les cases fils, permettant ainsi de déterminer l'idMatch du **prochain match** [sans entrer dans les détails, celle-ci équivaut à $N1 + \text{floor}(\text{nbr de cases totales pour le tour où se situe le tournoi} / 2)$]

Maintenant que nous savons ça, revenons-en à la réduction du délai générée par une requête SQL. La requête qui prenait le plus de temps était celle qui servait à créer toutes les lignes initialisées avec la valeur NULL (lignes vierges) de la table matches, lors de la première saisie d'une rencontre, d'une date ou d'un lieu. Plus les tournois étaient gros (au-delà d'une soixantaine d'équipes environ), plus un délai était perceptible à l'activation de cette requête.

En effet pour chaque tournoi, il fallait créer (nombre de cases * 2) - 1 lignes.

La solution trouvée fut plus conceptuelle qu'autre chose : le but étant d'améliorer au maximum l'utilisation du site pour les membres. Ainsi, la requête se chargeant de créer toutes les lignes vierges d'un tournoi ne se fait plus par le gestionnaire mais par l'admin lui-même lors de la création du tournoi. Il est donc le seul à percevoir le délai.

Cette décision implique un autre changement, en effet le gestionnaire possède à sa disposition un bouton "Remise à zéro du tournoi" . Au départ, ce bouton avait été codé ainsi : il supprime de la BDD toutes les lignes liées au tournoi, puis recrée les lignes vierges. L'utilisation de ce bouton nous ferait un peu revenir au point de départ... Heureusement, une solution assez ingénieuse fut trouvée, plutôt que d'utiliser les requêtes DELETE puis INSERT, l'astuce était de modifier directement les lignes déjà créées par l'admin avec la requête UPDATE, cela offrait un gain de temps considérable¹, réduisant ainsi grandement le délai.

Pour la suite nous avons commencé l'intégration de la gestion de tournois de type "poule" et "chacun pour soi". L'objectif était de garder l'interface de l'affichage d'un arbre et de la modifier pour obtenir un tableau avec une liste d'équipe associé à leur score pour le "chacun pour soi" et d'ajouter une grille des rencontres ainsi qu'un détails de chaque match dans le cas de "poules".

note : je ne suis pas sûr des temps que j'utilise et je sais pas si c'est trop ou pas assez

Conclusion :

Une application de base et une base de données spécifique étaient nécessaires dans ce projet. Nous les avons implémentées, nous avons également pu mettre en place plusieurs fonctionnalités supplémentaires (profil, gestion des informations personnelles, statistiques...). Néanmoins, dans le futur, il serait possible de rendre le site encore plus polyvalent en ajoutant des fonctionnalités supplémentaires.

La première que nous pouvons évoquer, celle à laquelle nous avons beaucoup discuté entre nous et avec notre encadrant, est la création de deux autres types de tournois : les championnats et les poules. Malheureusement, par manque de temps cela n'a pas pu être mis en place, cela dit la possibilité de créer un tournoi de ces deux types existe dans notre site, et nous permet de donner une utilité à un bouton qui fonctionne : le bouton "trie par type de tournoi".

Nous aurions également pu modifier la fonction des demandes de modération pour la rendre plus flexible, notamment la possibilité de réintroduire une demande en cas de refus, mais seulement après un certain nombre de jours.

Enfin, une autre amélioration future aurait été l'intégration d'une gestion des matchs en plusieurs sets pour des sports comme le tennis par exemple.

Concernant les difficultés rencontrées, nous pourrions citer Git, que nous avons découvert au cours de ce projet. Sa prise en main fut compliquée au début, mais une fois celui-ci maîtrisé, il fut indispensable pour l'avancement du projet. Au fil des semaines, nous avons également appris à mieux nous organiser et à communiquer entre nous, ce qui nous a permis d'optimiser notre travail et de réduire de nombreux conflits liés au code.

De plus, nous avons souvent dû nous réadapter suite à des changements que nous n'avions pas anticipés sur le moment. C'est le cas lorsqu'il a parfois fallu ajouter des attributs supplémentaires dans la base de données pour certaines fonctionnalités. Cela pouvait poser problème en créant des conflits entre les clés primaires et secondaires déjà existantes. C'est pourquoi, au fur et à mesure des réunions, nous nous sommes posé davantage de questions sur ce que nous

devions faire, afin de ne pas avoir à modifier sans cesse des choses qui entraîneraient toujours des erreurs sur l'ensemble du site.

La réalisation de ce projet nous a tout d'abord appris à gérer le travail en équipe, ce qui était essentiel dans l'avancement du projet. Grâce à quelques erreurs faite au début, nos réunions sont devenues de plus en plus efficaces, ce qui nous a permis de mieux anticiper les problèmes futurs.

Remerciements

Nous tenons à remercier notre encadrant François Suro qui nous à accompagner et conseiller tout au long du projet .

Source :

¹<https://stackoverflow.com/questions/1271641/in-sql-is-update-always-faster-than-delet-einsert#:~:text=Obviously%2C%20the%20answer%20varies%20based,implemented%20faster%20than%20DELETE%2BININSERT.&text=The%20other%2C%20minor%2C%20issue%20is,that%20row%20remain%20the%20same>