Groupe 3 - MAKY
ALLIOUX Yvan
ARGENTIN Margaux
CHENAIS Apolline
QUEMENEUR Katell

Projet développement java web

Rapport d'analyse UML

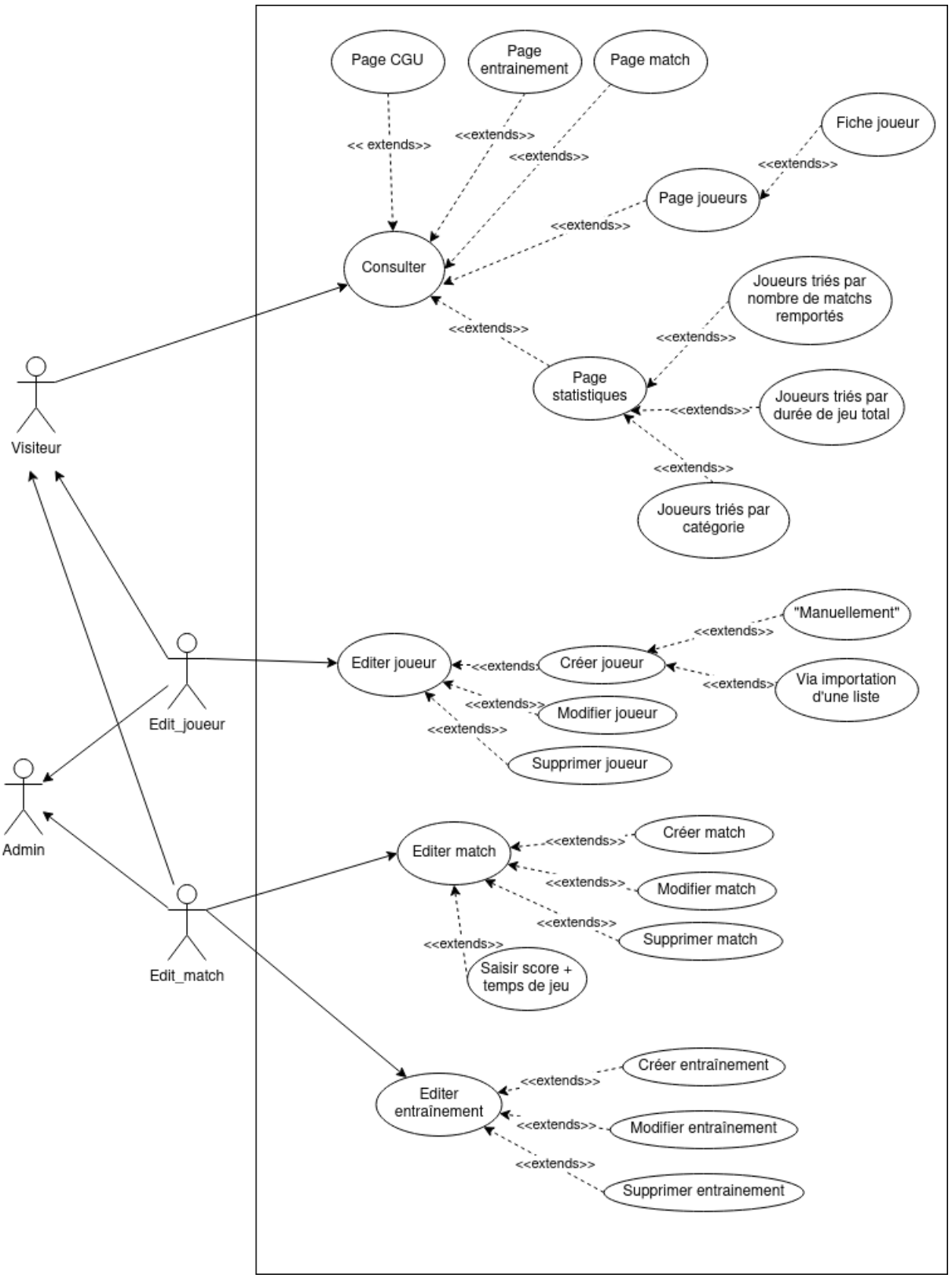
Sommaire

I.Diagramme de cas d'utilisation

Le site web est composé de deux parties. Une partie publique, accessible à tous les visiteurs du site, et une partie privée, disponible uniquement après authentification. Les administrateurs et les éditeurs ont également accès à la partie publique car ce sont des visiteurs particuliers. Les visiteurs ne peuvent que consulter les différentes pages sans pouvoir modifier aucune information. Ils ne peuvent donc accéder qu'à la page principale de chaque objet (joueur, match, entraînement). Il peut également trier l'ordre d'apparition des informations tels que la liste des joueurs par nom ou classement. Les visiteurs ont également accès à la page de statistiques sur laquelle ils peuvent voir les joueurs triés par nombre de matchs remportés ou par durée de jeu totale.

Un administrateur est à la fois un éditeur de joueur et de match. Un éditeur de joueur peut créer, modifier et supprimer un joueur sur la page de la liste de joueurs. Un éditeur de match peut créer, modifier et supprimer un match et un entraînement sur les pages de la liste de matchs et d'entraînements. Il peut également ajouter un score et une durée à un match joué. A chaque édition d'un éditeur, celui-ci est envoyé vers une nouvelle page lui permettant d'effectuer l'action voulue.

Dans les différents cas d'utilisation présentés ci-dessous, nous considérons que les administrateurs, les éditeurs de joueurs et les éditeurs de matchs sont déjà authentifiés et peuvent donc déjà accéder aux options et aux pages de modifications.


Page CGU
Page entrainement
Page match
<< extends>>
<<extends>>
<<extends>>
Fiche joueur
<<extends>>
Page joueurs
<<extends>>
Consulter
Joueurs triés par nombre de matchs remportés
<<extends>>
<<extends>>
Page statistiques
Joueurs triés par durée de jeu total
<<extends>>
<<extends>>
Joueurs triés par catégorie
Visiteur
"Manuellement"
<<extends>>
Editer joueur
<<extends>
Créer joueur
<<extends>
Via importation d'une liste
<<extends>>
Modifier joueur
<<extends>>
Supprimer joueur
Edit_joueur
Admin
Editer match
<<extends>>
Créer match
<<extends>>
Modifier match
<<extends>>
Supprimer match
<<extends>>
Saisir score + temps de jeu
Edit_match
Editer entraînement
<<extends>>
Créer entraînement
<<extends>>
Modifier entraînement
<<extends>>
Supprimer entrainement

II. Uses Cases

1. Consulter la fiche d'un joueur

La fiche d'un joueur contient toutes les informations reliées à un joueur choisi. Pour accéder à la fiche d'un joueur, on accède à la page "joueur" contenant la liste de tous les joueurs. Pour chaque diagramme de ce use-case, on considère que le visiteur est déjà sur cette page des joueurs.

Diagramme d'activité

Lorsque l'on souhaite se rendre sur la fiche d'un joueur, on sélectionne un joueur puis on clique sur le bouton "+". Une requête est envoyée au servlet pour récupérer les données à afficher. Une fois les données récupérées, la fiche du joueur s'affiche.

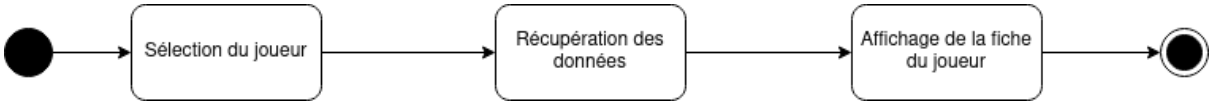


Diagramme de séquence

La consultation des fiches des joueurs peut se faire pour tous les visiteurs. Ces derniers vont donc sélectionner un joueur de la liste pour afficher ces informations. Une demande d'affichage est donc envoyée vers le servlet, qui est donc le contrôleur, qui va récupérer les différentes données auprès du modèle qui est notre base de données. La base de données renvoie les informations vers le contrôleur, puis la fiche joueur s'ouvre, affichant les données récupérées. Lorsque le visiteur clique sur le bouton "fermer" de la fiche du joueur, la fiche se ferme et le contrôleur retourne sur la page de la liste des joueurs.

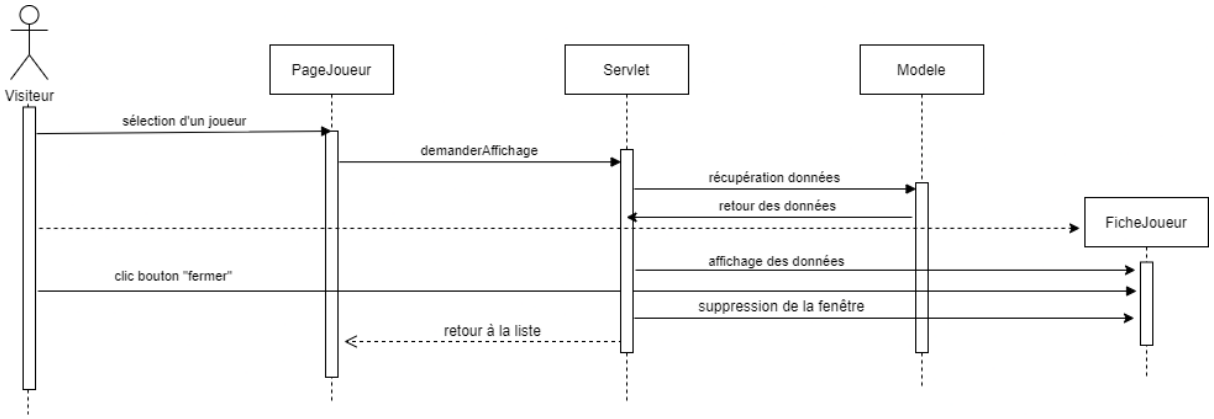


Diagramme d'état-transition

Lorsque l'on est sur la page de la liste des joueurs, la fiche joueur n'est pas chargée. Lorsque le visiteur choisit un joueur, sa fiche se charge et s'affiche. Celle-ci se ferme lorsque l'utilisateur clique sur l'icône de fermeture et celle-ci n'est donc plus chargée. La fiche joueur reste non chargée si le chargement de la page ne s'effectue pas correctement comme lors d'un échec de récupération de données vers la base de données.

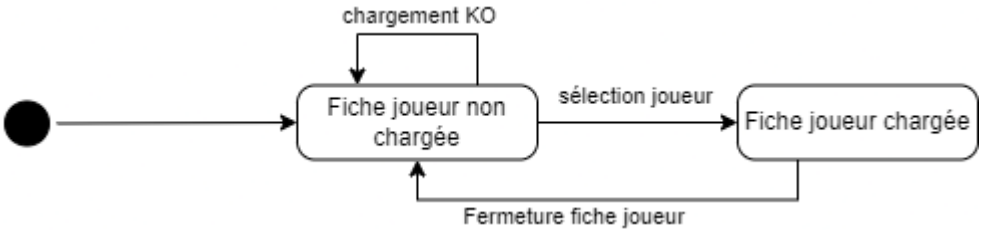
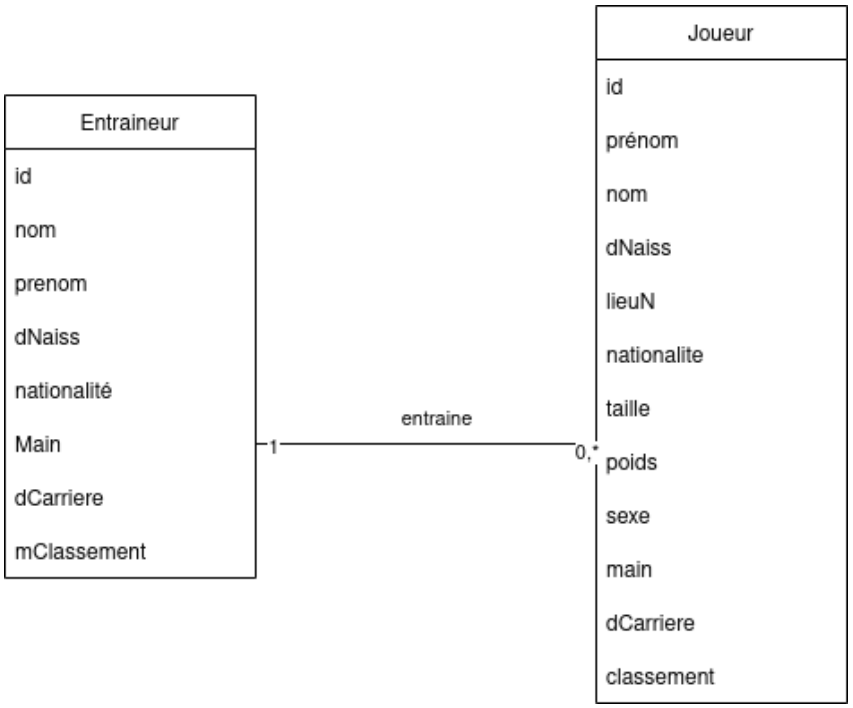


Diagramme de classe

Pour afficher les différentes informations concernant un joueur sur sa fiche, nous avons besoin de la classe "Joueur", qui contient toutes les valeurs du joueur telles que son nom, son prénom ou encore son classement. Nous avons également besoin de la classe "Entraîneur" nous permettant de donner les informations sur l'entraîneur du joueur sélectionné comme son nom. Un joueur de peut être entraîné que par un seul entraîneur, tandis qu'un entraîneur peut entraîner zéro ou plusieurs joueurs.



2. Créer un joueur manuellement

Pour ajouter un joueur, il faut se rendre sur la page des joueurs. Il y a deux manières d'ajouter un joueur :

- manuellement en remplissant un formulaire
- grâce au téléchargement d'un fichier CSV.

Dans ce cas d'utilisation, nous nous concentrerons sur l'ajout manuel. Pour chaque diagramme de ce use-case, on considère que l'administrateur ou l'éditeur de joueur est déjà sur cette page des joueurs.

Diagramme d'activité

Lorsqu'un administrateur ou un éditeur de joueurs reçoit une demande d'ajout d'un joueur, celui-ci choisit l'option de création manuelle sur la page de la liste des joueurs. Une nouvelle fenêtre s'affiche et l'administrateur doit ensuite saisir les informations du joueur à ajouter puis il enregistre le joueur en cliquant sur "enregistrer". Lors de l'enregistrement, les données sont vérifiées dans la base de données pour s'assurer qu'il n'y a pas de doublon et que les données sont conformes aux critères attendus. Si les données sont incorrectes, l'utilisateur est renvoyé à la saisie de données. Si les données sont valides, les données sont ajoutées à la base de données et l'utilisateur est renvoyé sur la liste des joueurs.

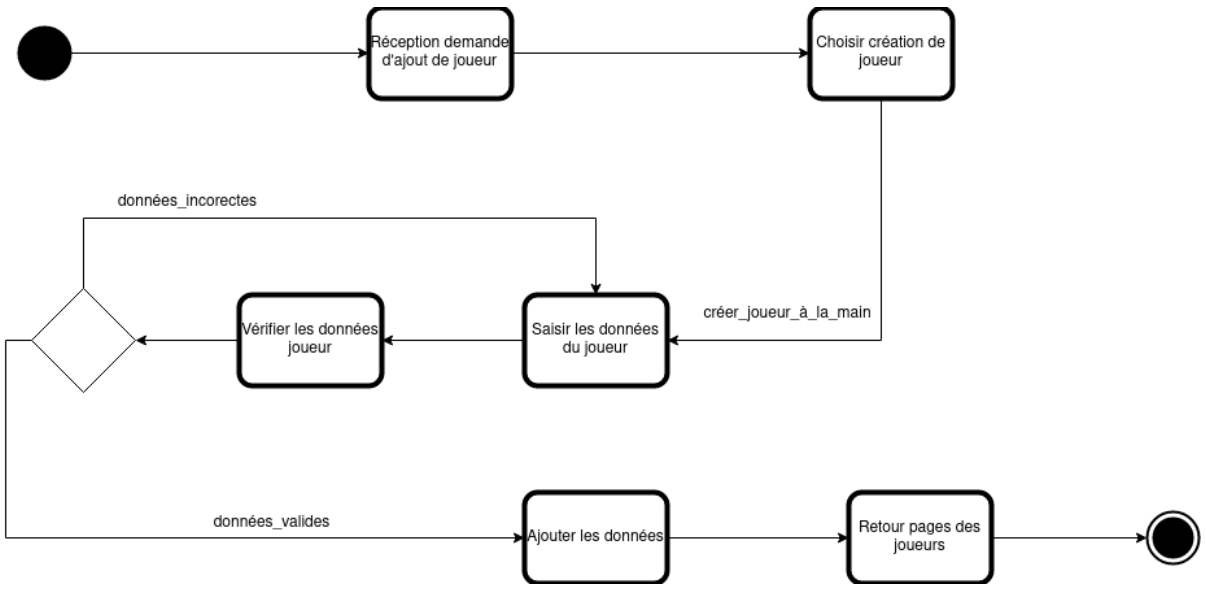


Diagramme de séquence

Un joueur peut être créé par un administrateur ou un éditeur de joueur. Ces derniers vont donc cliquer sur le bouton “créer” pour créer un joueur manuellement. Une demande de création est donc envoyée vers le servlet. Une page pour saisir les informations s’ouvre et l’utilisateur doit entrer toutes les informations du joueur à créer. Lorsque toutes les informations sont entrées, il clique sur “enregistrer”. L’information de sauvegarde est envoyée au servlet qui renvoie une requête vers la base de données pour sauvegarder les différentes données entrées. Une fois le joueur ajouté à la base de données, la page se ferme et le contrôleur met à jour la liste des joueurs.

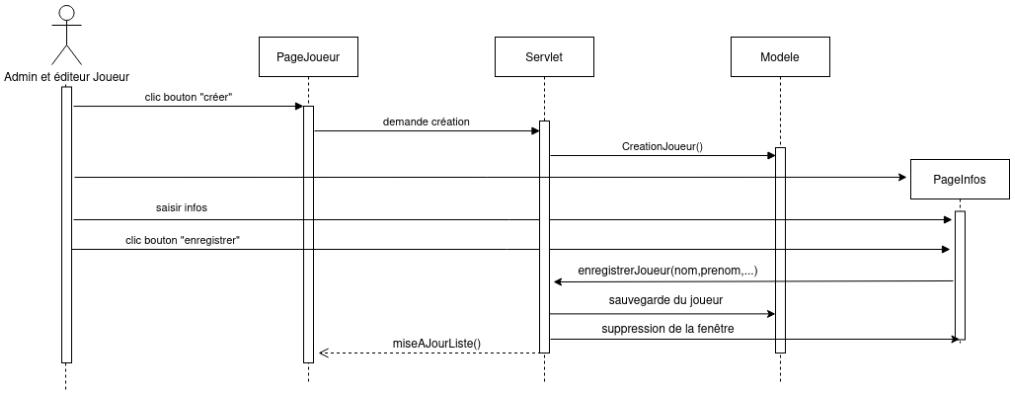


Diagramme d’état-transition

Lorsque nous sommes sur la page des joueurs, le joueur que l’on souhaite ajouter n’est pas encore créé. Après un clic sur la création manuelle, le joueur est en cours d’ajout. Après la saisie d’informations, si les informations ne sont pas correctes, le joueur ne sera pas créé, les données de la base de données non modifiées et l’utilisateur retourne sur la page de saisie d’informations. Le joueur est de nouveau en cours d’ajout. Si les données saisies sont correctes, les données sont ajoutées à la base de données et le joueur est donc créé. L’utilisateur est donc renvoyé vers la page de liste des joueurs. L’état du joueur passe donc en non créé car on considère qu’une fois revenu sur cette page, nous parlons du prochain joueur à créer et non pas du joueur précédemment créé.

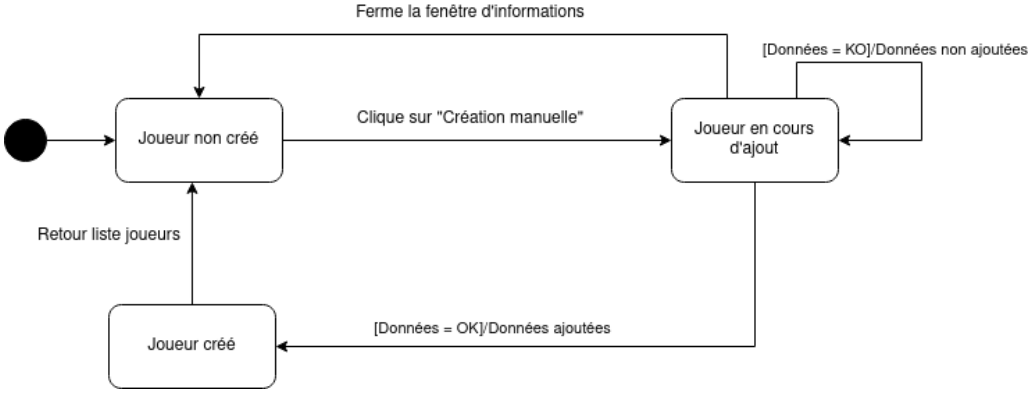
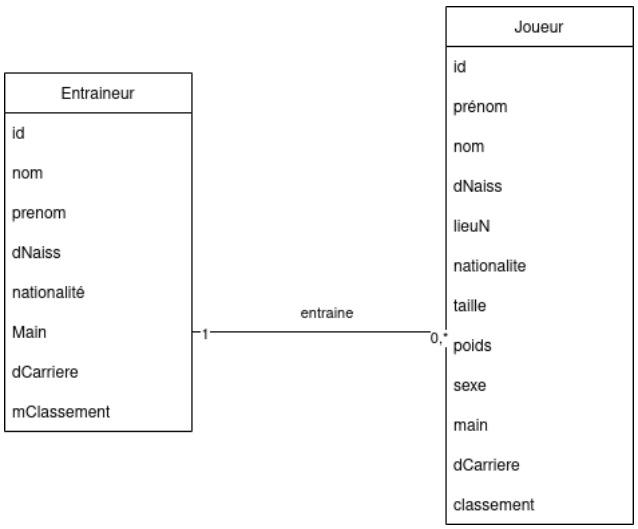


Diagramme de classe

Pour créer un joueur, nous avons besoin de la classe “Joueur” contenant toutes les valeurs demandées pour ajouter un joueur telles que son nom, son prénom ou sa main utilisée pour jouer. Nous avons également besoin de la classe "Entraîneur" nous permettant de spécifier l’entraîneur associé au joueur que l’on souhaite créer. Un joueur ne peut être entraîné que par un seul entraîneur, tandis qu’un entraîneur peut entraîner zéro ou plusieurs joueurs.



3. Créer des joueurs avec un fichier CSV

Dans ce cas d’utilisation, nous nous concentrerons sur l’ajout grâce à un fichier CSV. Pour chaque diagramme de ce use-case, on considère que l’administrateur ou l’éditeur de joueur est déjà sur cette page des joueurs.

Diagramme d’activité

Lorsqu’un administrateur ou un éditeur de joueur reçoit une demande d’ajout d’un joueur, celui-ci choisit l’option de création avec un fichier csv sur la page de la liste des joueurs. L’utilisateur choisit le fichier à importer puis clique sur “enregistrer” pour confirmer la récupération des données. Si les données ne sont pas valides et le chargement des données est incomplet, l’utilisateur doit spécifier un autre fichier CSV. Si le chargement des données est complet, les données sont ajoutées et le joueur créé. L’utilisateur se voit donc retourner sur la page de la liste des joueurs.

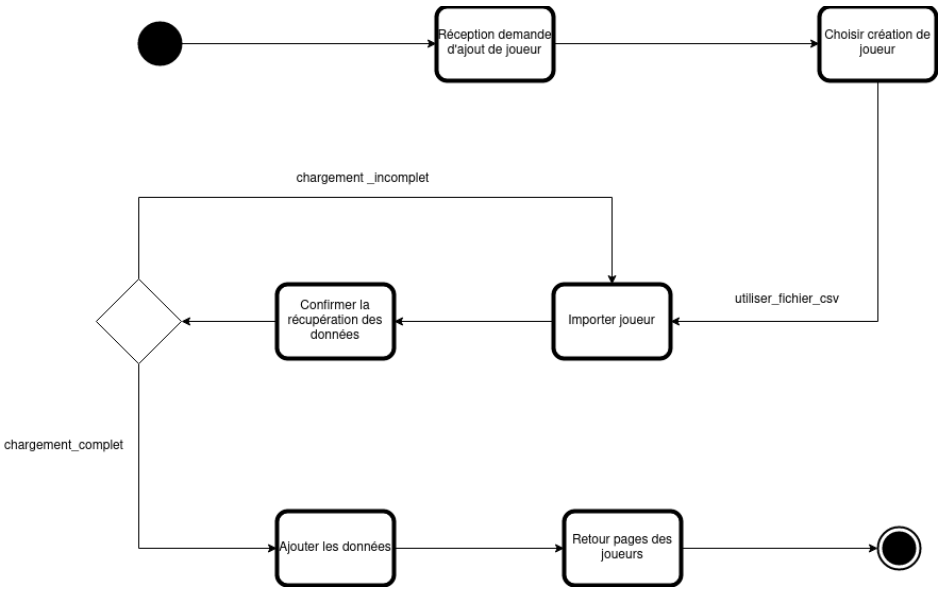


Diagramme de séquence

Des joueurs peuvent être ajoutés avec un fichier CSV par un administrateur ou un éditeur de joueur. Ces derniers doivent cliquer sur l'importation d'un fichier CSV. Une demande de création est donc envoyée vers le servlet. Une page pour choisir un fichier CSV s'ouvre et l'utilisateur doit choisir un fichier. Lorsqu'il choisit le fichier, l'utilisateur doit confirmer son choix. L'information de sauvegarde est envoyée au servlet qui renvoie une requête vers la base de données pour vérifier que les données à ajouter ne créent pas de doublon et qu'elles sont conformes aux types attendus. La base de données confirme que les données sont conformes au servlet qui envoie ensuite une requête de sauvegarde des différents joueurs dont les données sont dans le fichier. Une fois les joueurs ajoutés à la base de données, la page se ferme et le contrôleur met à jour la liste des joueurs.

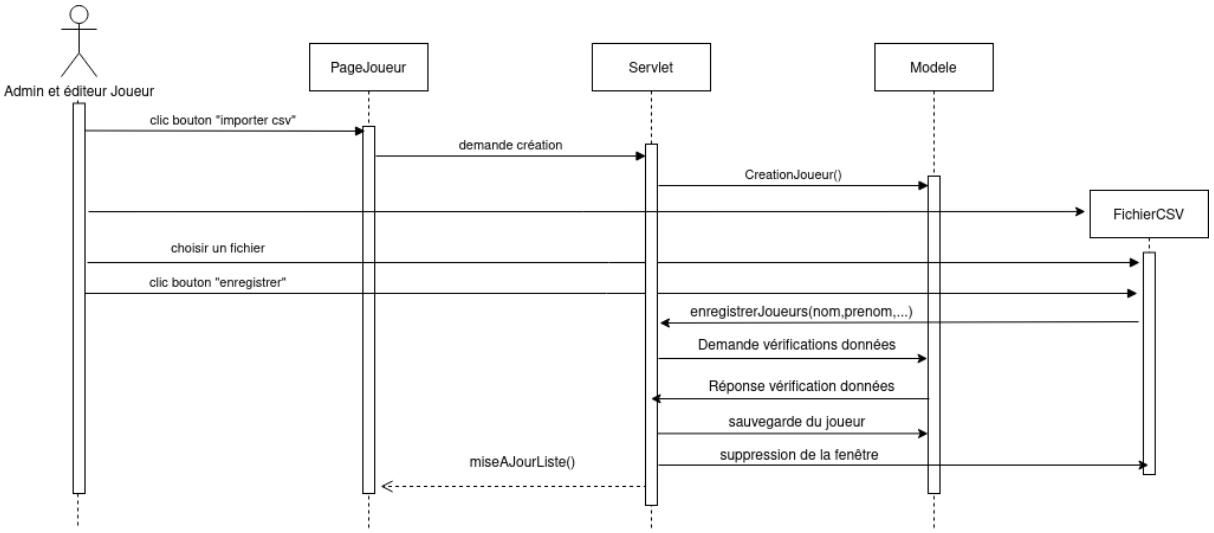


Diagramme d'état-transition

Lorsque nous sommes sur la page des joueurs, les joueurs que l'on souhaite ajouter ne sont pas encore créés. Après un clic sur la création avec un fichier CSV, le joueur est en cours d'ajout. Après le choix du fichier, si les informations ne sont pas correctes, les joueurs ne seront pas créés, les données de la base de données non modifiées et l'utilisateur retourne sur la liste des joueurs. Les joueurs sont de nouveau en cours d'ajout. Si les données sont correctes, les données sont ajoutées à la base de données et les joueurs sont donc créés. L'utilisateur est donc renvoyé vers la page de liste des joueurs. L'état du joueur passe donc en non créé car on considère qu'une fois revenu sur cette page, nous parlons du prochain joueur à créer et non pas du joueur précédemment créé.

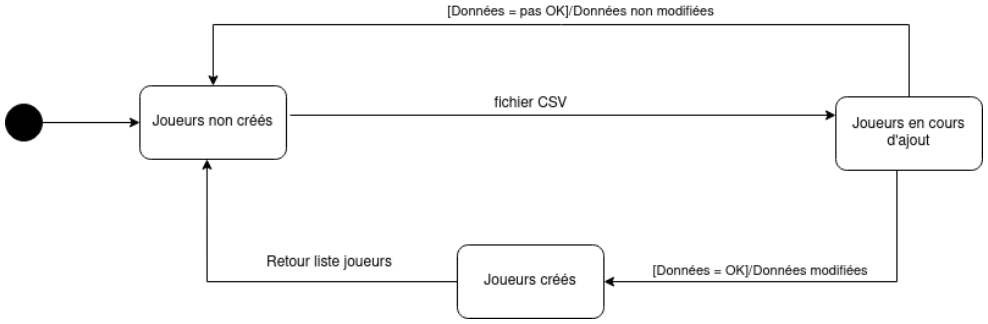
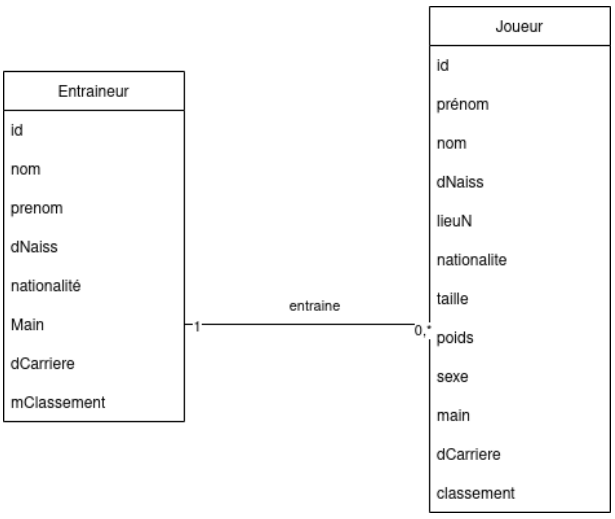


Diagramme de classe

Pour créer un joueur, nous avons besoin de la classe "Joueur" contenant toutes les valeurs demandées pour ajouter un joueur telles que son nom, son prénom ou sa main utilisée pour jouer. Nous avons également besoin de la classe "Entraîneur" nous permettant de spécifier l'entraîneur associé au joueur que l'on souhaite créer. Un joueur ne peut être entraîné que par un seul entraîneur, tandis qu'un entraîneur peut entraîner zéro ou plusieurs joueurs.



4. Modifier un match

Pour modifier un match, il faut se rendre sur la page des matchs. Pour chaque diagramme de ce use-case, on considère que l'administrateur ou l'éditeur de match est déjà sur la page des matchs. Pour accepter l'accès à la modification, il faut vérifier que le match n'a pas encore été joué. On considère donc ici que les matchs modifiés vérifient cette condition.

Diagramme d'activité

Lorsqu'un administrateur ou un éditeur de match reçoit une demande de modification d'un match, celui-ci choisit l'option de création sur la page de la liste des matchs. Une nouvelle fenêtre s'affiche et l'administrateur doit ensuite saisir les informations du match à ajouter puis il enregistre le match en cliquant sur "enregistrer". Lors de l'enregistrement, les données sont vérifiées dans la base de données pour s'assurer qu'il n'y a pas de doublon et que les données sont conformes aux critères attendus. Si les données sont incorrectes, l'utilisateur est renvoyé à la saisie de données. Si les données sont valides, les données sont modifiées dans la base de données et l'utilisateur est renvoyé sur la liste des matchs.

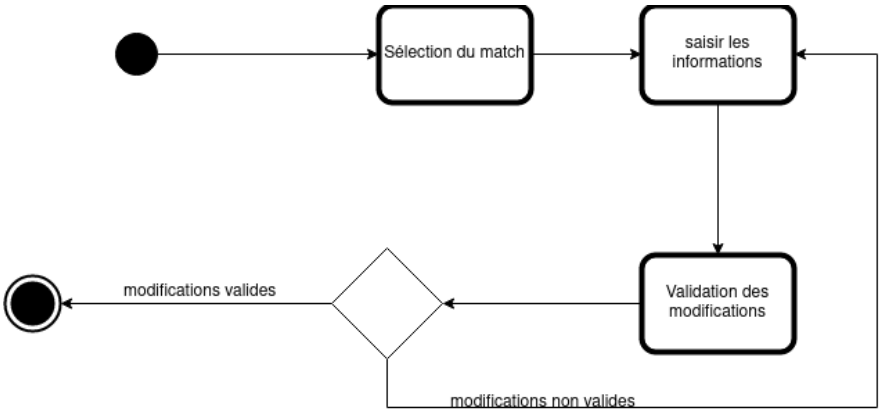


Diagramme de séquence

Un match peut être modifié par un administrateur ou un éditeur de matchs. Ces derniers vont donc cliquer sur le bouton "modifier" pour modifier un match. Une demande de modification est donc envoyée vers le servlet. Une page pour modifier les informations s'ouvre et l'utilisateur doit entrer toutes les informations du match à modifier. Lorsque toutes les informations sont entrées, il clique sur "enregistrer". Le servlet vérifie les disponibilités des joueurs concernés par le match, ainsi que le court utilisé pour vérifier qu'ils ne rentrent pas en conflit avec un autre match. L'information de sauvegarde est envoyée au servlet qui renvoie une requête vers la base de données pour sauvegarder les différentes données entrées. Une fois le

match modifié dans la base de données, la page se ferme et le contrôleur met à jour la liste des matchs.

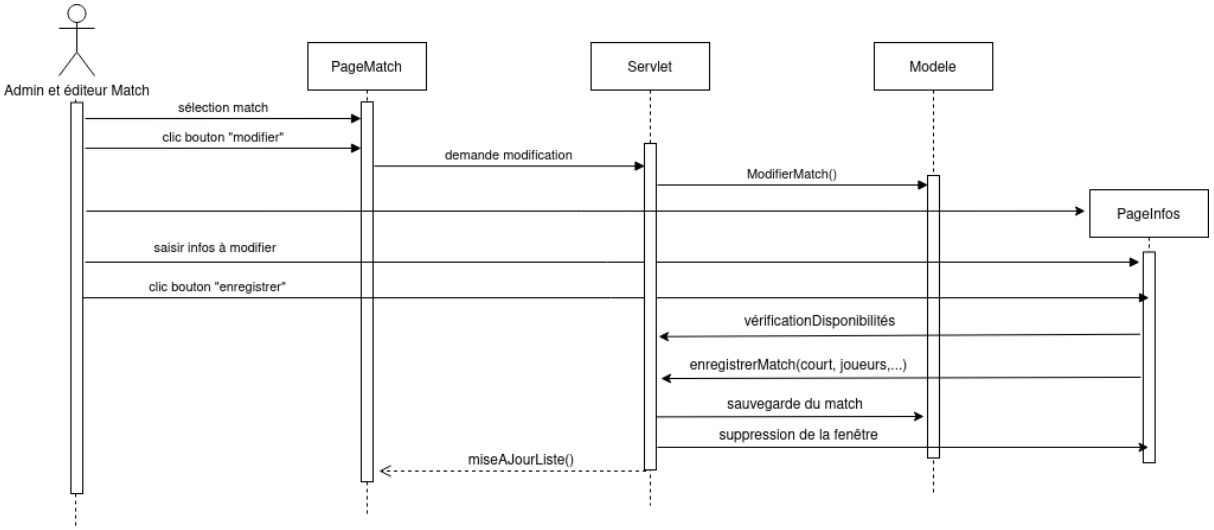


Diagramme d'état-transition

Lorsque nous sommes sur la page des matchs, le match que l'on souhaite modifier n'est pas encore modifié. Après un clic sur la modification, le match est en cours de modification. Après la saisie d'informations, si les informations ne sont pas correctes, le match reste en cours de modification, les données de la base de données ne sont pas modifiées et l'utilisateur doit modifier les informations non correctes.Si l'utilisateur annule la saisie de données, et ferme la fenêtre d'informations, le match retourne à l'état de non modifié. Si les données saisies sont correctes, les données sont modifiées dans la base de données et le match est donc modifié. L'utilisateur est donc renvoyé vers la page de liste des matchs. L'état du match passe donc en non modifié car on considère qu'une fois revenu sur cette page, nous parlons du prochain match à modifier et non pas du match précédemment modifié.

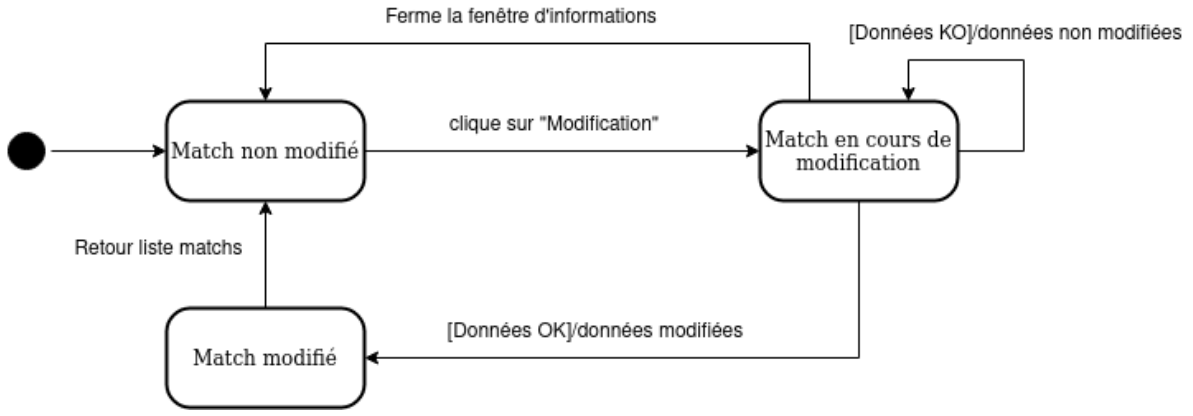
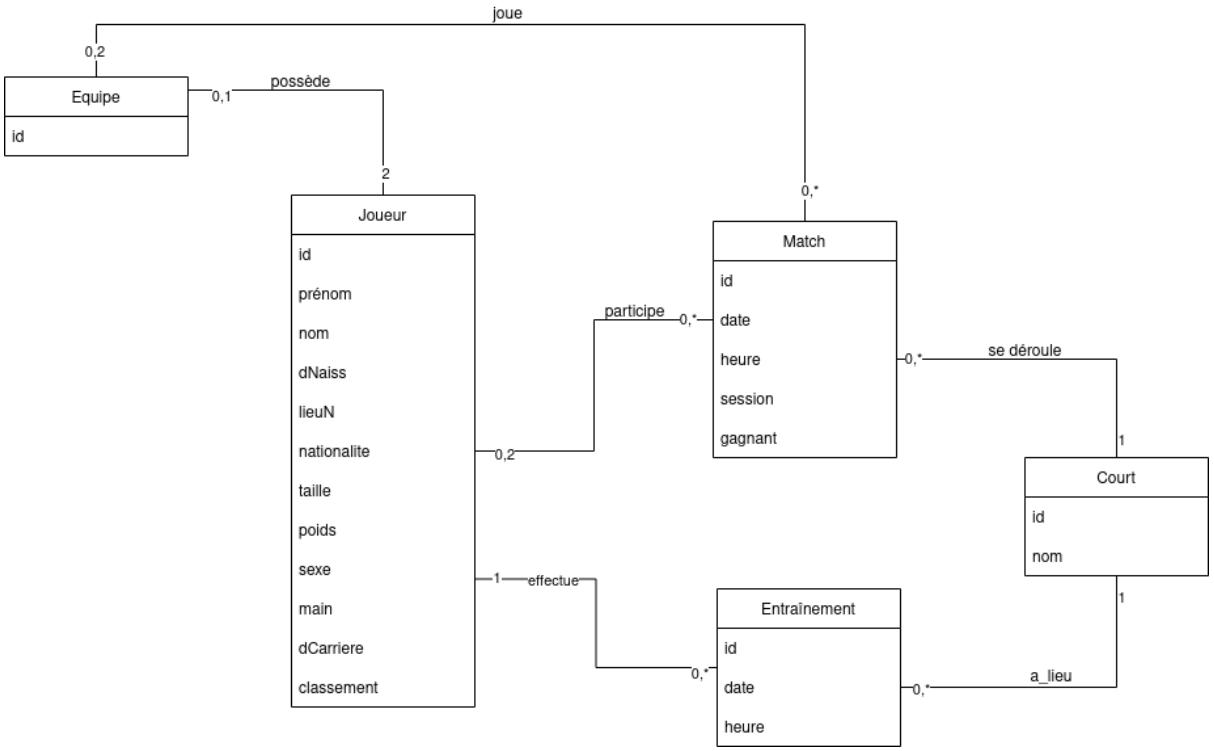


Diagramme de classe

Pour modifier un match, nous avons besoin de la classe “Match”, nous permettant de récupérer les différents attributs d’un match et de pouvoir ainsi les modifier. Nous avons également besoin des classes "Équipe" et “Joueur” qui nous permettent de lier un joueur ou une équipe à un match et de ce fait savoir s' il y a un conflit avec d’autres matchs du joueur ou de l’équipe concerné par le match modifié. Un joueur peut être dans aucune ou une équipe et une équipe contient deux joueurs. Une équipe ou un joueur peut jouer aucun ou plusieurs matchs et un match est composé d’aucun ou de deux joueurs ou équipes. Également, nous avons besoin des classes “Court” et “Entraînement”, nous permettant de savoir si le court n’est pas déjà utilisé pour un autre match ou pour un entraînement sur le même créneau et donc vérifier sa disponibilité. Un match ou un entraînement se déroule sur un court et il peut se dérouler aucun à plusieurs matchs ou entraînement sur un court.



5. Ajouter score

Pour ajouter un score, il faut se rendre sur la page match. Pour chaque diagramme de ce use-case, on considère que l’administrateur ou l’éditeur de match est déjà sur la page des matchs. Pour ajouter un score, il faut que la date du match soit passée. Nous allons donc considérer que cette condition est remplie. On considère donc qu’on ne peut pas ajouter de score si le match n’a pas encore été joué.

Diagramme d’activité

Lorsque l'administrateur ou l'éditeur de match souhaite ajouter un score à un match passé, celui-ci doit choisir l'option d'ajout de score de ce match. Une nouvelle fenêtre s'ouvre dans laquelle l'utilisateur saisit le score. Lors de l'enregistrement, les données saisies sont vérifiées pour s'assurer qu'elles correspondent bien au type attendu par la base de données. Si les données sont incorrectes, l'utilisateur est renvoyé à la saisie de score. Si les données sont valides, le score est ajouté au match et l'utilisateur est renvoyé sur la liste des matchs.

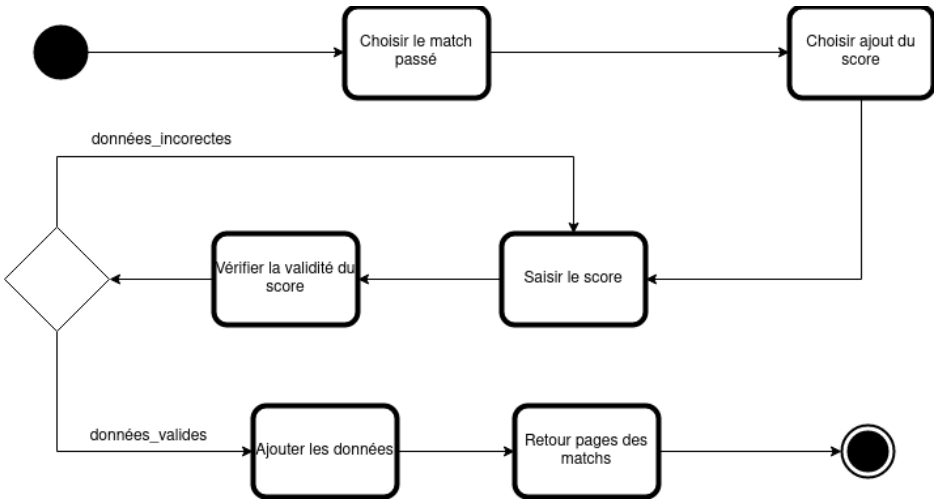


Diagramme de séquence

Un score peut être ajouté par un administrateur ou un éditeur de matchs. Ces derniers vont donc cliquer sur le bouton "ajouter un score" pour ajouter un score à un match. Une demande d'ajout est donc envoyée vers le servlet. Une page pour saisir les informations s'ouvre et l'utilisateur doit saisir les scores. Lorsque toutes les informations sont entrées, il clique sur "enregistrer". L'information de sauvegarde est envoyée au servlet qui renvoie une requête vers la base de données pour sauvegarder le score. Une fois le score sauvegardé dans la base de données, la page se ferme et le contrôleur met à jour la liste des matchs.

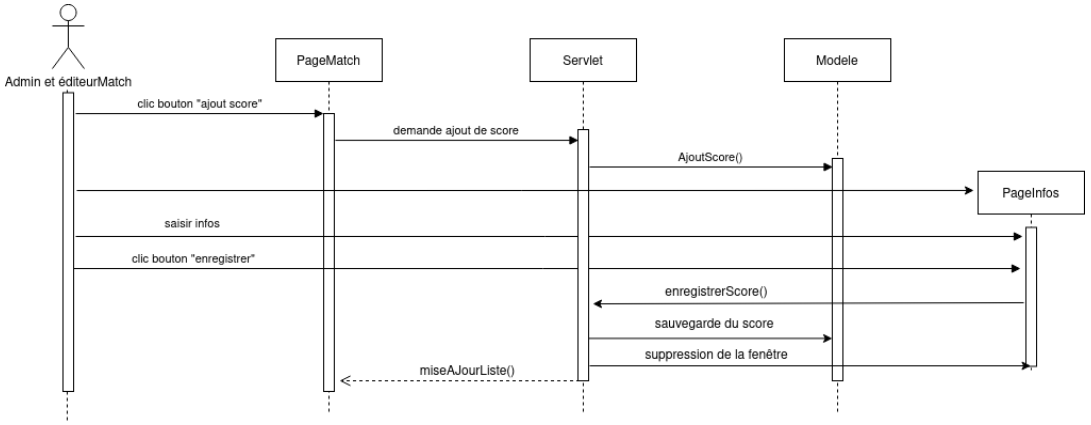


Diagramme d'état-transition

Lorsque nous sommes sur la page des matchs, le score que l'on souhaite ajouter n'est pas encore créé. Après un clic sur l'ajout, le score est en cours d'ajout. Après la saisie d'informations, si les informations ne sont pas correctes, le score ne sera pas créé, les données de la base de données non modifiées et l'utilisateur retourne sur la page de saisie d'informations. Le score est de nouveau en cours d'ajout. Si les données saisies sont correctes, les données sont ajoutées à la base de données et le score est donc créé. L'utilisateur est donc renvoyé vers la page de liste des matchs. L'état du score passe donc en non créé car on considère qu'une fois revenu sur cette page, nous parlons du prochain score à ajouter et non pas du score précédemment ajouté.

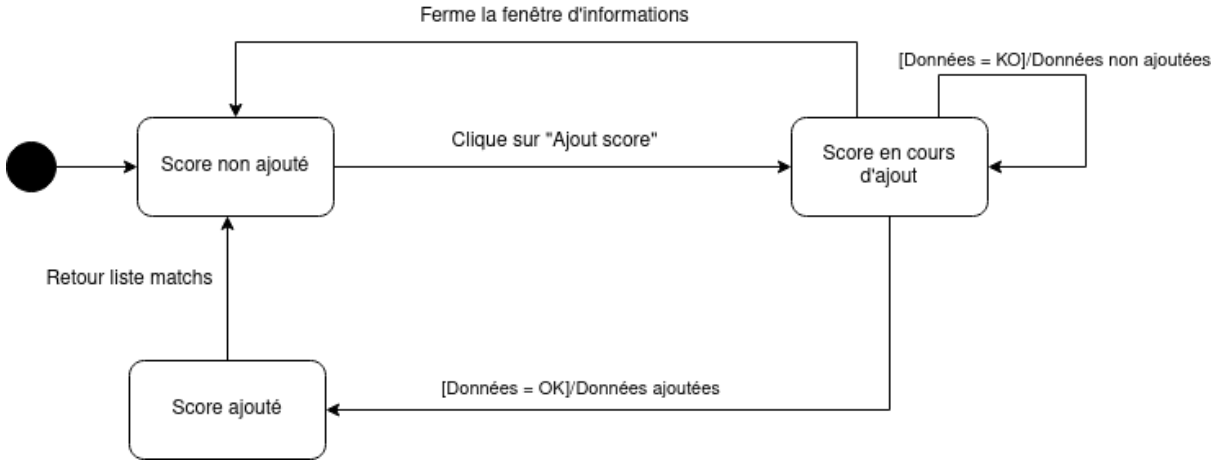
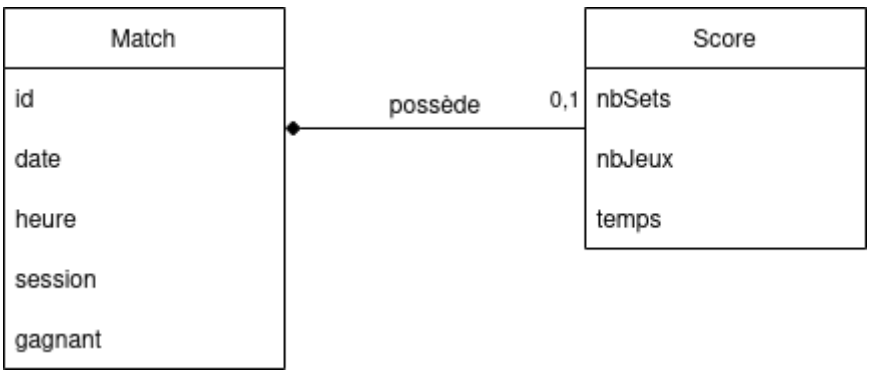


Diagramme de classe

Pour ajouter un score, nous avons besoin de la classe "Score", permettant d'enregistrer les différentes informations liées au score. Nous avons également besoin de la classe "Match" car le score est un composant de match et est donc obligatoirement lié à celui-ci. Un match possède aucun ou un score.



6. Créer un entraînement

Pour créer un entraînement, il faut se rendre sur la page entraînements. Pour chaque diagramme de ce use-case, on considère que l'administrateur ou l'éditeur de match est déjà sur la page des entraînements. On considère également qu'un joueur ne peut pas avoir plusieurs entraînements sur une même journée. On considère donc que le joueur n'a pas d'entraînement prévu le jour choisi pour l'entraînement et que le court choisi est disponible.

Diagramme d'activité

Lorsqu'un administrateur ou un éditeur de match reçoit une demande d'ajout d'un entraînement, celui-ci choisit l'option de création sur la page de la liste des joueurs. Lorsque l'utilisateur lance la demande, il faut vérifier la disponibilité du joueur sur la journée choisie car on considère qu'un joueur ne peut faire qu'un entraînement par jour. Si le joueur a déjà un entraînement de prévu, la création de l'entraînement est annulée. Si le joueur n'a pas d'entraînement de prévu, il faut ensuite vérifier la disponibilité du court. Si le court n'est pas disponible sur ce créneau, la page propose un autre créneau. Si ce créneau est refusé, la création de l'entraînement est annulée. Si le court est disponible ou que le nouveau créneau proposé est accepté, l'entraînement est créé.

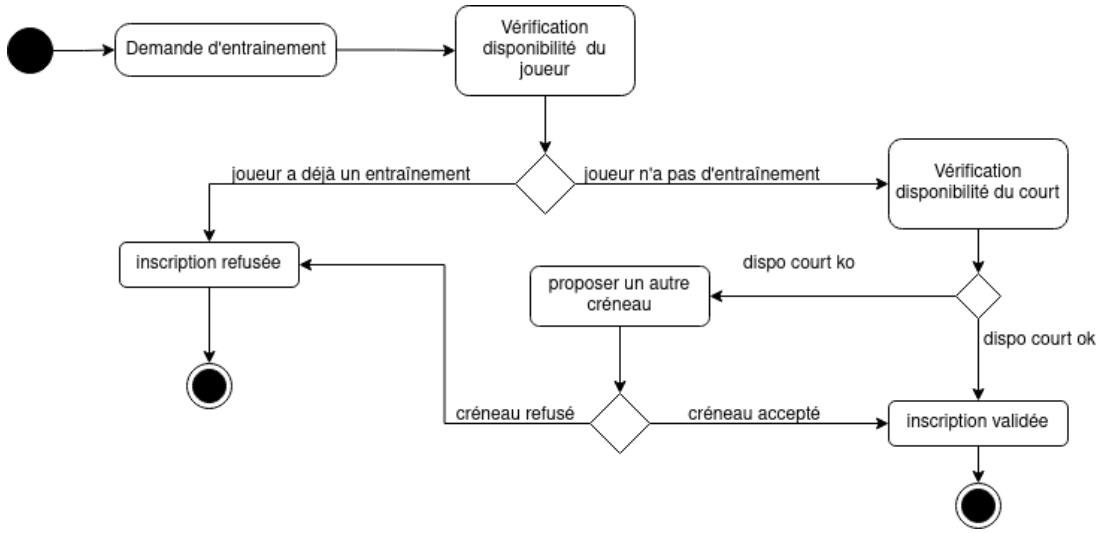


Diagramme de séquence

Un entraînement peut être créé par un administrateur ou un éditeur de match. Ces derniers vont donc cliquer sur le bouton "créer" pour créer un entraînement. Une demande de création est donc envoyée vers le servlet. Une page pour saisir les informations s'ouvre et l'utilisateur doit entrer toutes les informations du joueur à créer. Lorsque toutes les informations sont entrées, il clique sur "enregistrer". Le

servlet va donc ensuite vérifier la disponibilité des joueurs sur la journée pour ne pas que le joueur ait plus d'un entraînement par jour. Puis il vérifie la disponibilité du court pour ne pas entrer en conflit avec un autre match ou un autre entraînement sur le créneau sélectionné. L'information de sauvegarde est ensuite envoyée au servlet qui renvoie une requête vers la base de données pour sauvegarder les différentes données entrées. Une fois l'entraînement ajouté à la base de données, la page se ferme et le contrôleur met à jour la liste des entraînements.

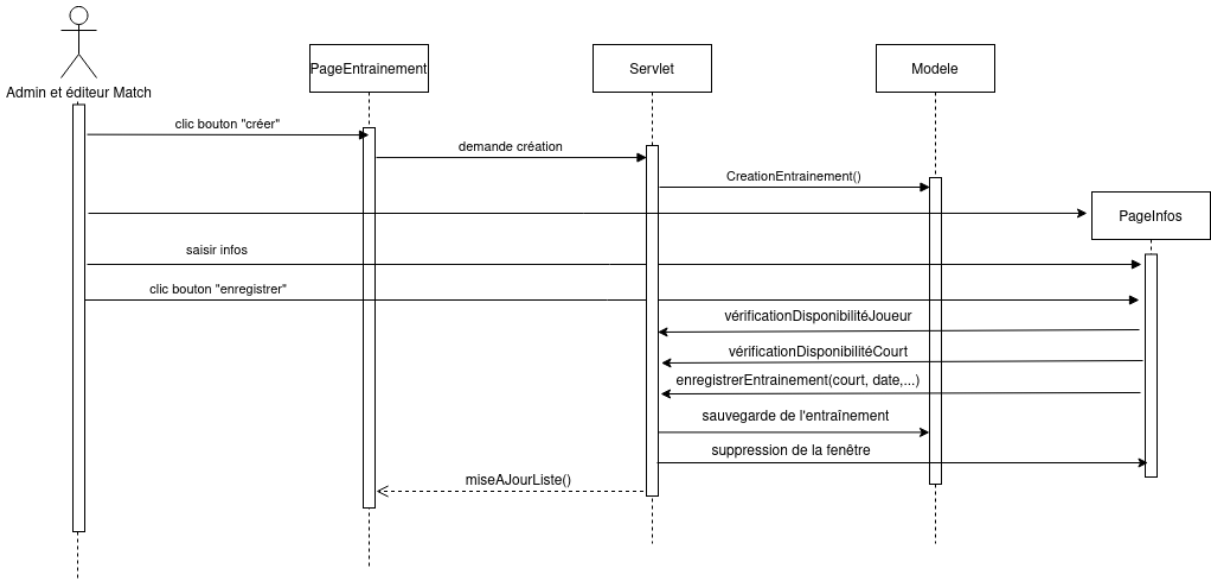


Diagramme d'état-transition

Lorsque nous sommes sur la page des entraînements, l'entraînement que l'on souhaite ajouter n'est pas encore créé. Après un clic sur la création, l'entraînement est en cours d'ajout. Après la saisie d'informations, si les informations ne sont pas correctes, l'entraînement ne sera pas créé, les données de la base de données non modifiées et l'utilisateur retourne sur la page de saisie d'informations. L'entraînement est de nouveau en cours d'ajout. Si l'utilisateur choisit d'annuler la création en fermant la fenêtre, l'entraînement revient à l'état de non créé. Si les données saisies sont correctes, les données sont ajoutées à la base de données et l'entraînement est donc créé. L'utilisateur est donc renvoyé vers la page de liste des entraînements. L'état de l'entraînement passe donc en non créé car on considère qu'une fois revenu sur cette page, nous parlons du prochain entraînement à créer et non pas de l'entraînement précédemment créé.

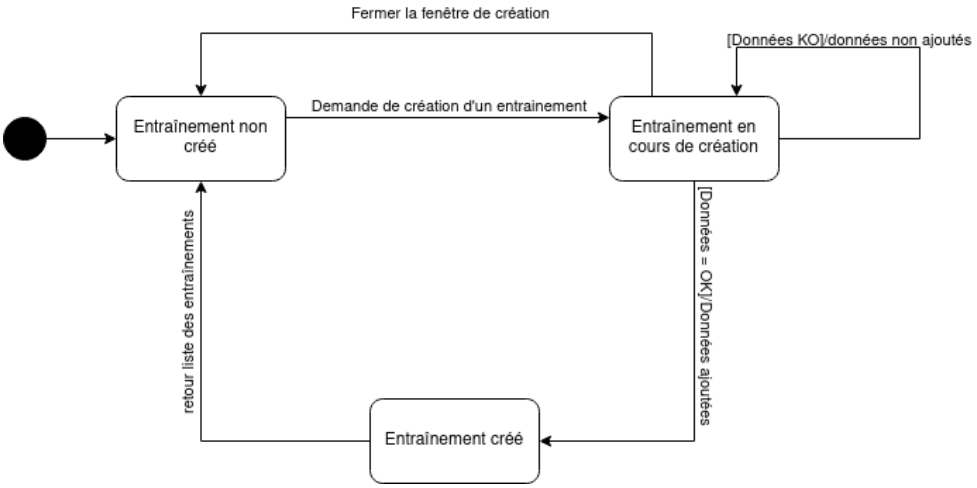
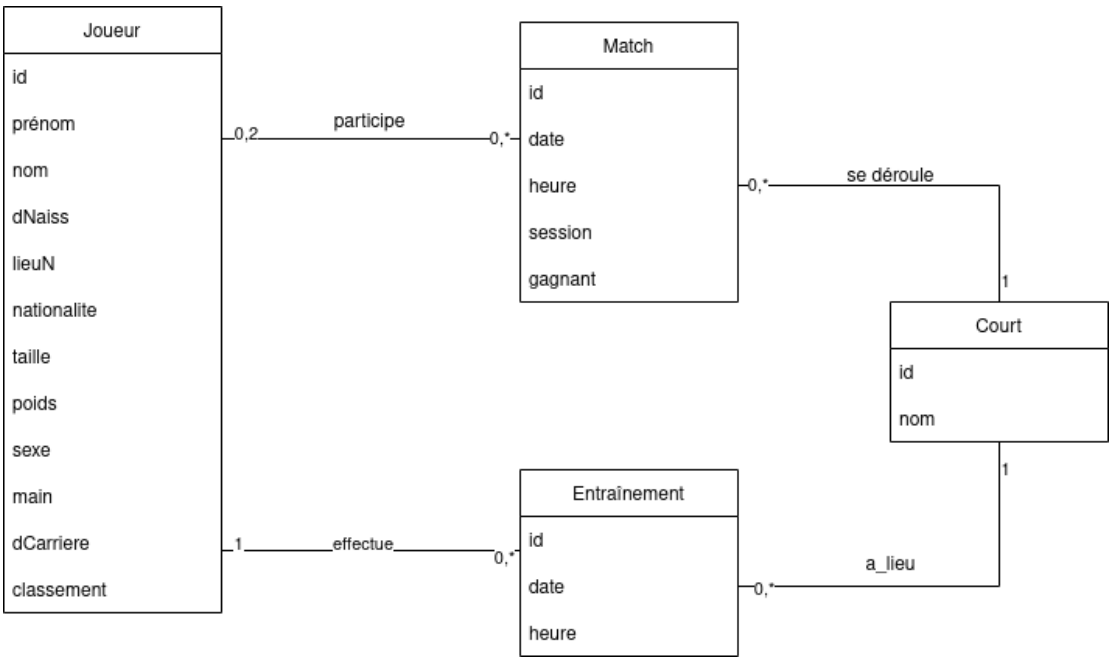


Diagramme de classe

Pour créer un entraînement, nous avons besoin de la classe “Entraînement”, nous permettant de renseigner les différents attributs d’un entraînement. Nous avons également besoin de la classe “Joueur” qui nous permet de lier un joueur à l’entraînement et de ce fait savoir s' il y a un conflit avec d’autres entraînements du joueur concerné par l’entraînement créé. Un joueur peut jouer aucun ou plusieurs entraînements et un entraînement est composé d’aucun ou de deux joueurs. Également, nous avons besoin des classes “Court” et “Match”, nous permettant de savoir si le court n’est pas déjà utilisé pour un autre entraînement ou pour un match sur le même créneau et donc vérifier sa disponibilité. Un match ou un entraînement se déroule sur un court et il peut se dérouler aucun à plusieurs matchs ou entraînement sur un court.



III. Diagramme d'objet général

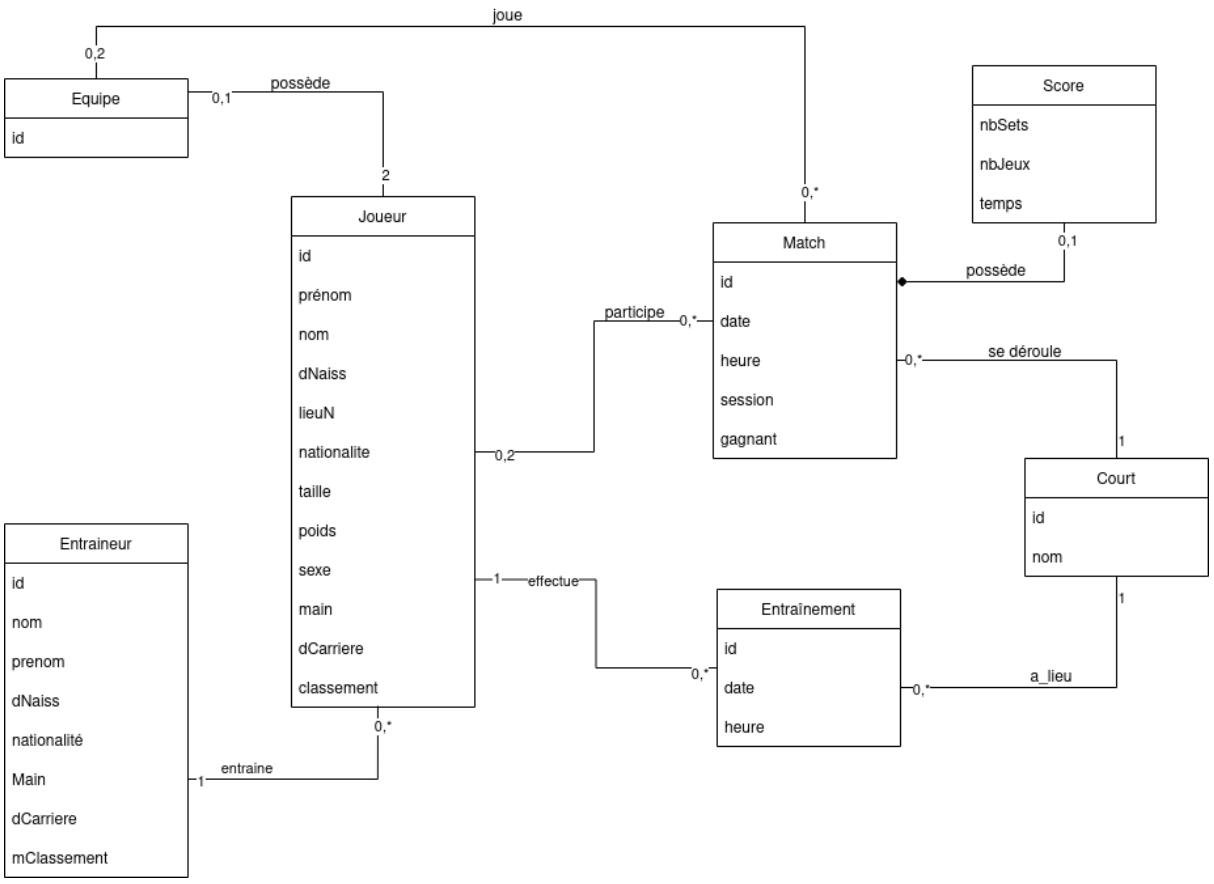
Nous avons coupé le diagramme d'objet général en deux parties. La première partie présente le système général du projet (joueurs, matchs, entraînements,...). La seconde partie représente la gestion des rôles (administrateurs, éditeurs,...).

1. Système général

Explication des différentes cardinalités entre les classes :

Un joueur a obligatoirement un entraîneur. Un entraîneur peut entraîner aucun ou plusieurs joueurs car nous considérons que nous pouvons d'abord créer un entraîneur sans le lier à un joueur.
Un joueur peut être dans aucune ou une équipe. On considère donc qu'un joueur peut être créé sans appartenir à une équipe. Une équipe possède deux joueurs et ne peut donc pas exister sans joueur.
Un joueur peut participer à aucun ou plusieurs matchs. Un joueur peut donc être créé sans nécessairement être associé à un match. Un match peut contenir aucun ou deux joueurs. Un match peut donc être créé sans nécessairement lui ajouter des joueurs.
Un joueur peut effectuer aucun ou plusieurs entraînements. Un entraînement doit obligatoirement être effectué par un joueur et ne peut donc pas être créé sans être relié à un joueur.
Une équipe peut jouer aucun ou plusieurs matchs. Un match peut être joué par aucune ou deux équipes. Un match peut donc être créé sans nécessairement lui ajouter d'équipes.
Un match possède aucun ou un score. Un score ne peut pas exister sans match. Le score ne peut pas être rentré si la date du match n'est pas passé. Le match peut donc exister sans un score.
Un match se déroule obligatoirement sur un court et ne peut pas être créé sans être associé à aucun court. Un court peut accueillir aucun ou plusieurs matchs.
De même, un entraînement se déroule obligatoirement sur un court et ne peut pas être créé sans être associé à aucun court. Un court peut accueillir aucun ou plusieurs entraînements.

Nous avons décidé de faire une classe "Score" pour ne pas surcharger la classe "Match" et également pour pouvoir renseigner plus tard ces informations en fonction de la date et de l'heure du match.



2. Gestion des rôles

Nous avons choisi de ne créer qu'une seule classe "Utilisateur" qui englobe les trois différents types d'éditeurs. Leur différenciation est gérée à travers l'attribut "rôle". En fonction de la lettre choisie, l'utilisateur aura le rôle d'éditeur de joueur, éditeur de match ou d'administrateur. En spécifiant "A", l'utilisateur sera un administrateur. En spécifiant la lettre "E", l'utilisateur sera un éditeur de joueur. En spécifiant la lettre "M", l'utilisateur sera un éditeur de match.
Les attributs "pseudos" et "mot_de_passe" spécifient les différentes informations à renseigner pour s'authentifier sur le site web.