EPHEC 2019-2020

HENSMANS Olivier

baccalauréat en informatique de gestion

FifaManager

Projet pour le cours de SGBD de Monsieur Vincent Fievez

Table des matières

[1 Contexte 4](#_Toc38566683)

[1.1 Cadre 4](#_Toc38566684)

[1.2 Projet 4](#_Toc38566685)

[2 Analyse métier 5](#_Toc38566686)

[2.1 Description de la solution envisagée 5](#_Toc38566687)

[2.2 Intervenants 5](#_Toc38566688)

[3 Fonctions attendues 6](#_Toc38566689)

[3.1 Exigences fonctionnelles 6](#_Toc38566690)

[3.1.1 BackEnd 6](#_Toc38566691)

[3.1.2 MatchManagement 6](#_Toc38566692)

[3.2 Exigences non fonctionnelles 6](#_Toc38566693)

[4 Contraintes Business 7](#_Toc38566694)

[5 Analyse fonctionnelle 9](#_Toc38566695)

[5.1 BackEnd -Diagramme de Use Cases 9](#_Toc38566696)

[5.1.1 Mockup : accueil 10](#_Toc38566697)

[5.1.2 Use Case : Générer un championnat 11](#_Toc38566698)

[5.1.3 Use Case : Transférer un joueur 16](#_Toc38566699)

[5.1.4 Use Case : Visualiser le classement par joueur 19](#_Toc38566700)

[5.1.5 Use Case : Visualiser le classement par équipe 21](#_Toc38566701)

[5.1.6 Use Case : Visualiser & mettre à jour un match 23](#_Toc38566702)

[5.2 Matchmanagement – Diagramme de Use Cases 24](#_Toc38566703)

[5.2.1 Mockup : accueil 24](#_Toc38566704)

[5.2.2 Use Case : Remplir une feuille de match 25](#_Toc38566705)

[5.2.3 Use Case : Inscrire / modifier les résultats 29](#_Toc38566706)

[5.3 Matchmanagement – diagramme d’état 33](#_Toc38566707)

[6 Contraintes fonctionnelles 34](#_Toc38566708)

[6.1 Règles d’accès et autorisations 34](#_Toc38566709)

[6.2 Règles de structures 34](#_Toc38566710)

[6.2.1 Championnats 34](#_Toc38566711)

[6.2.2 Quarters 34](#_Toc38566712)

[6.2.3 Intersaisons 34](#_Toc38566713)

[6.2.4 EquipesParticipation 35](#_Toc38566714)

[6.2.5 équipes 35](#_Toc38566715)

[6.2.6 Joueurs 35](#_Toc38566716)

[6.2.7 Transferts 36](#_Toc38566717)

[6.2.8 Goals 36](#_Toc38566718)

[6.2.9 Cartons jaunes 36](#_Toc38566719)

[6.2.10 Cartons Rouges 36](#_Toc38566720)

[6.2.11 Matchs 37](#_Toc38566721)

[6.2.12 Feuille de match 37](#_Toc38566722)

[6.2.13 JoueursParticipation 37](#_Toc38566723)

[6.3 Règles de validation 37](#_Toc38566724)

[6.3.1 Championnats 37](#_Toc38566725)

[6.3.2 Quarters 38](#_Toc38566726)

[6.3.3 Intersaisons 38](#_Toc38566727)

[6.3.4 EquipesParticipation 38](#_Toc38566728)

[6.3.5 équipes 38](#_Toc38566729)

[6.3.6 Joueurs 38](#_Toc38566730)

[6.3.7 Transferts 39](#_Toc38566731)

[6.3.8 Goals 39](#_Toc38566732)

[6.3.9 Cartons jaunes 40](#_Toc38566733)

[6.3.10 Cartons Rouges 40](#_Toc38566734)

[6.3.11 Matchs 41](#_Toc38566735)

[6.3.12 Feuille de match 41](#_Toc38566736)

[6.3.13 JoueursParticipation 41](#_Toc38566737)

[6.4 Règles de calcul 42](#_Toc38566738)

[6.4.1 Matchs 42](#_Toc38566739)

[6.4.2 JoueursParticipations 42](#_Toc38566740)

[7 Description des entités 43](#_Toc38566741)

[7.1 Championnats 43](#_Toc38566742)

[7.2 Équipes 43](#_Toc38566743)

[7.3 Joueurs 43](#_Toc38566744)

[7.4 Transferts 43](#_Toc38566745)

[7.5 Matchs 44](#_Toc38566746)

[7.6 Feuilles de match 44](#_Toc38566747)

[8 Schéma relationnel de la solution 45](#_Toc38566748)

[8.1 Entité Association 45](#_Toc38566749)

[8.2 Schéma Relationnel 46](#_Toc38566750)

[8.3 Implémentation des contraintes 47](#_Toc38566751)

[9 Analyse technique 51](#_Toc38566752)

[9.1 Technologies proposées 51](#_Toc38566753)

[9.2 Architecture applicative 51](#_Toc38566754)

[9.3 Listes erreurs 52](#_Toc38566755)

[9.3.1 TechnicalErrors – SQLExceptions 52](#_Toc38566756)

[9.3.2 BusinessErrors 54](#_Toc38566757)

[10 Lessons learned 55](#_Toc38566758)

[10.1.1 Ne rien faire à l’avance 55](#_Toc38566759)

[10.1.2 Bien identifier la technologie nécessaire 55](#_Toc38566760)

[10.1.3 Découper son code et tester sur des petits morceaux de code 55](#_Toc38566761)

[10.1.4 DATAGRIDVIEW – DATATABLE - Dataview 56](#_Toc38566762)

[10.1.5 Le mieux est l’ennemi du bien 56](#_Toc38566763)

[10.1.6 Ne pas faire le malin 56](#_Toc38566764)

[11 Conclusion 57](#_Toc38566765)

# Contexte

## Cadre

Ce projet est développé dans le cadre du cours de Projet de développement SGBD de Monsieur Fievez.

A travers ce projet plusieurs points sont demandés à l’étudiant :

* Analyse d’une demande d’un client (pas toujours claire et précise)
* Réalisation d’une entité association
* Réalisation d’une base de données
* Réalisation de triggers et procédures stockées pour appliquer les contraintes
* Réalisation d’un code en 3 couches – DAL / Business / View
* Utilisations de modèles et d’erreurs séparés de la couche business
* Accès aux données en lecture et écriture uniquement par procédures stockées

D’autres points mériteraient également notre attention mais l’idée générale est dessinée : l’étudiant doit prouver qu’il est capable de développer une solution à un problème énoncé de manière non précise et changeante.

Des choix techniques, méthodologiques et analytiques doivent être réalisés par l’étudiant et doivent être justifiés au lecteur.

## Projet

Le projet en lui-même consiste au développement d’un système d’information permettant de faire fonctionner deux systèmes applicatifs avec des fonctionnalités différentes :

* BackEnd
* MatchManagement

Ces deux systèmes doivent utiliser une même base de données mais des utilisateurs et vue différentes.

# Analyse métier

## Description de la solution envisagée

La solution doit permettre la gestion de championnats de foot de la création du championnat en tant que tel à la gestion du calendrier et des résultats des matchs, des transferts de joueurs et de la lecture du classement.

Plus précisément, et comme évoqué plus haut, la solution est découpée en 2 parties distinctes.

Une partie BackEnd qui s’occupera principalement de la gestion du championnat (création, inscription équipe, génération du calendrier des matchs, transfert des joueurs, classements).

Une partie MatchManagement qui s’occupera de la gestion des matchs en tant que telle, inscription des joueurs sur la feuille de match, inscriptions des goals, cartons jaunes et rouges.

## Intervenants

Deux intervenants existeront pour notre solution :

* Un utilisateur BackEnd pour la partie BackEnd
* Un utilisateur MatchManagement pour la partie MatchManagement

# Fonctions attendues

## Exigences fonctionnelles

### BackEnd

La partie BackEnd devra permettre :

|  |  |
| --- | --- |
| FM-BE001 | La gestion des championnats |
| FM-BE002 | La gestion du calendrier des matchs |
| FM-BE003 | La gestion du transfert de joueurs |
| FM-BE004 | La gestion des matchs |
| FM-BE005 | La visualisation des classements par championnat |

### MatchManagement

La partie MatchManagement devra permettre :

|  |  |
| --- | --- |
| FM-MM001 | La gestion des feuilles de match |
| FM-MM002 | La gestion des matchs individuellement |

## Exigences non fonctionnelles

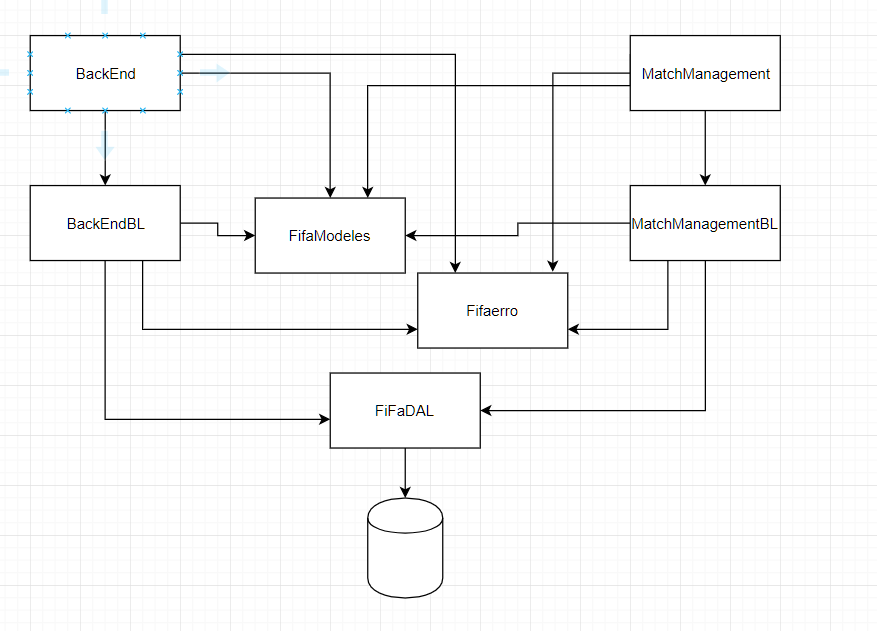
Le système d’information devra :

|  |  |
| --- | --- |
| FM-ENF001 | Être ergonomique pour l’utilisateur |
| FM-MM002 | Être Sécurisé |
| FM-MM003 | Permettre une vue complète et simplifiée des informations |

# Contraintes Business

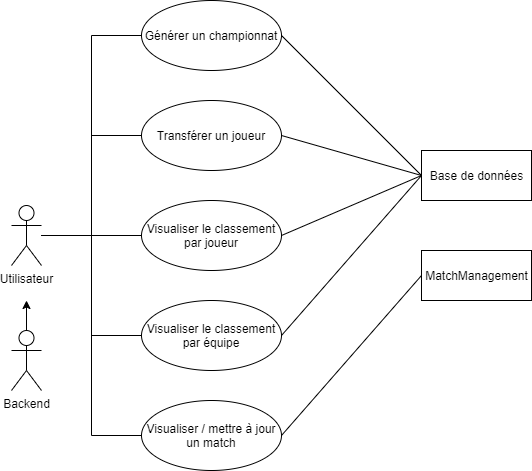
La solution proposée devra :

|  |  |
| --- | --- |
| FM-CB001 | Avoir une découpe en 3 couches (DAL, Business Layer, GUI) |
| FM-CB002 | Être divisé en 2 parties (BackEnd, MatchManahement) |
| FM-CB003 | Avoir une solution avec 7 projets (2 windows forms & 5 class libraries) et suivre le schéma ci-dessous |
| FM-CB004 | Accéder à la DB pour la partie BackEnd via Entity Framework DB First |
| FM-CB005 | Accéder à la DB pour la partie MatchManagement via ADO.NET |
| FM-CB006 | Avoir une seule base de données dédiée pour les 2 parties |
| FM-CB007 | Avoir un accès aux tables de la DB uniquement par l’utilisateur sa |
| FM-CB008 | Avoir 2 schémas différents (BackEnd et MatchManagement) |
| FM-CB009 | Avoir le schéma BackEnd accessible uniquement par sa et BackEnd |
| FM-CB010 | Avoir le schéma MatchManagement accessible uniquement par sa et MatchManagement |
| FM-CB011 | Accéder aux données et les modifier via des procédures stockées spécifiques par schéma. |



# Analyse fonctionnelle

## BackEnd -Diagramme de Use Cases



### Mockup : accueil

Une image contenant capture d’écran

Description générée automatiquement

Menu de base présentant les différentes options offertes à l’utilisateur.

### Use Case : Générer un championnat

#### Description textuelle ecran 1 – créer un championnat

Résumé : permet à l’utilisateur BackEnd de créer un championnat, inscrire les équipes éligibles dedans et fixer les dates des matchs

Acteurs : BackEnd *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application BackEnd est lancée
* L’utilisateur a cliqué sur le bouton « Générer un championnat »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. L’utilisateur encode l’année du championnat qu’il souhaite
2. Le logiciel vérifie que la date est valide (postérieur ou égale à 1900, inférieure ou égale à 9998) avec un format en quatre chiffres et s’il n’existe pas déjà un championnat pour cette année
3. Le logiciel calcule la date de début maximum du championnat
4. Le logiciel met à jour le datetime picker en limitant le choix de l’utilisateur
5. Le logiciel met à jour le résumé des dates en fonction de la date encodée dans le datetime picker
6. Le logiciel interroge la base de données pour savoir quelles équipes sont éligibles pour l’année encodée
7. La base de données répond avec la liste des équipes éligibles
   1. Une équipe doit posséder de 5 à 10 joueurs au début du championnat
8. Le logiciel affiche les équipes éligibles avec des cases sélectionnables
9. L’utilisateur sélectionne la date de début du championnat
10. Le logiciel met à jour le résumé
11. L’utilisateur sélectionne les équipes
12. L’utilisateur clique sur le bouton enregistrer et passer aux matchs
13. Le logiciel vérifie que le nombre d’équipe soit de minimum 2
14. Le logiciel interroge la base de données pour vérifier que le championnat n’existe pas déjà
15. Le logiciel crée le championnat dans la base de données
16. Le logiciel crée les quarters dans la base de données
17. Le logiciel crée l’intersaison dans la base de données
18. Le logiciel inscrit la participation des équipes au championnat dans la base de données
19. Le logiciel ouvre le second écran pour l’inscription des matchs.

Scénarios alternatifs :

*A1 :* L’utilisateur encode une année avec une étendue non-conforme (<1900 ou >9998) ou des caractères

1. Le logiciel affiche un message d’erreur
2. L’utilisateur recommence au point 1 du scénario nominal

*A2 :* L’utilisateur encode une année pour laquelle il y a déjà un championnat

1. Le logiciel affiche un message d’erreur
2. L’utilisateur recommence au point 1 du scénario nominal

*A3 :* L’utilisateur sélectionne moins de 2 équipes

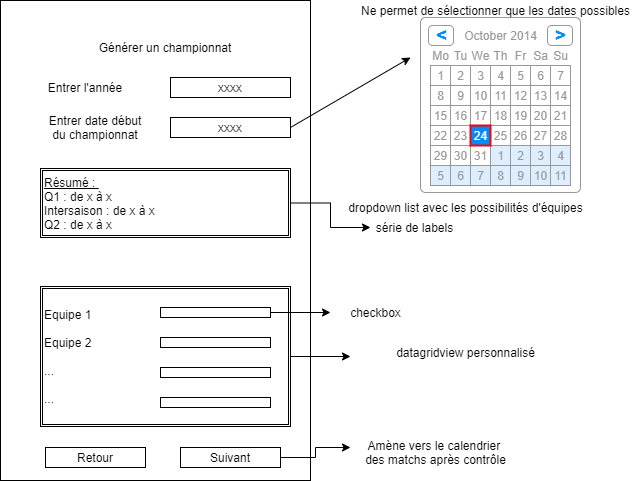
1. Le logiciel affiche un message d’erreur
2. L’utilisateur recommence au point 11 du scénario nominal

Enchainements d’erreurs :

*E1 :* L’utilisateur annule la création d’un championnat en cliquant sur le bouton « retour »

1. Le logiciel ferme la fenêtre et n’enregistre rien

#### Mockup ecran 1 – créer un championnat



En cliquant sur suivant le championnat et les quarters et l’intersaison sont créés s’ils répondent à tous les critères (n’existe pas déjà, …)

Vérifie également avant d’enregistrer une équipe si elle n’a pas déjà eu des transferts encodés l’année qui va être encodée

#### Description textuelle ecran 2 – générer le calendrier des matchs

Résumé : permet à l’utilisateur BackEnd de contrôler les dates affectées aux matchs d’un championnat et de donner des dates aux matchs qui n’en ont pas encore, ou de modifier les existantes dans la limite des contraintes.

Acteurs : BackEnd *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application BackEnd est lancée
* L’utilisateur vient de finir de créer un championnat via l’écran « générer un championnat »
* L’utilisateur a cliqué sur le bouton « Enregistrer »
* Toutes les conditions précédentes ont été remplies
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel génère automatiquement la liste des matchs et leur donne des dates selon les contraintes client et les choix de l’utilisateur lors de la création du championnat (année, liste des équipes, dates des quarters) :
   1. Chaque équipe joue une fois à domicile et une fois à l’extérieur contre toutes les autres équipes inscrites au championnat
   2. Une équipe ne peut pas joueur 2 fois sur un même week-end
   3. Une équipe ne peut pas jouer 2 fois le même jour, ni sans jour d’écart entre 2 matchs
   4. Les matchs d’une équipe doivent être répartis entre les 2 quarters
2. L’utilisateur vérifie que les dates lui conviennent bien et les modifie au besoin
3. L’utilisateur clique sur le bouton « sauvegarder »
4. Le logiciel crée les matchs dans la base de données.

Scénarios alternatifs :

*A1 :* Il y a trop d’équipes par rapport au nombre de week-end

1. Le logiciel indique comme date le 1er janvier 1801 pour les matchs du premier quarter qui n’ont pas pu recevoir de dates et le 1er janvier 1802 pour ceux du 2e quarter.
2. L’utilisateur modifie ces dates et recommence le scénario au point 3

*A2 :* L’utilisateur souhaite de l’aide pour trouver les lignes qu’il doit remplir

1. L’utilisateur clique sur le bouton aide
2. Le logiciel affiche en jaune toutes les lignes pour lesquelles une date valide n’a pas été donnée
3. Le scénario reprend au point 2 du scénario A1

Enchainements d’erreurs :

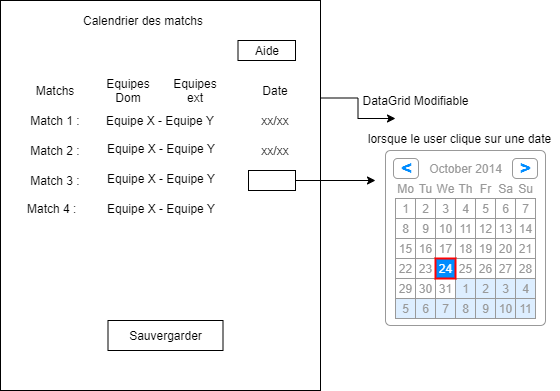
*E1 :* L’utilisateur n’a pas donné de date pour une ou plusieurs lignes qui le nécessitait

1. Le logiciel affiche un message d’erreur
2. Le logiciel affiche en jaune la ligne qui pose problème
3. Le scénario nominal reprend au point 2

*E2 :* L’utilisateur n’enregistre pas les matchs et ferme la fenêtre

1. Le logiciel affiche un message d’information
2. Le logiciel supprime le championnat précédemment crée dans la base de données
3. Le logiciel supprime les quarters précédemment crées dans la base de données
4. Le logiciel supprime l’intersaison précédemment crée dans la base de données

#### Mockup ecran 2 – générer le calendrier des matchs



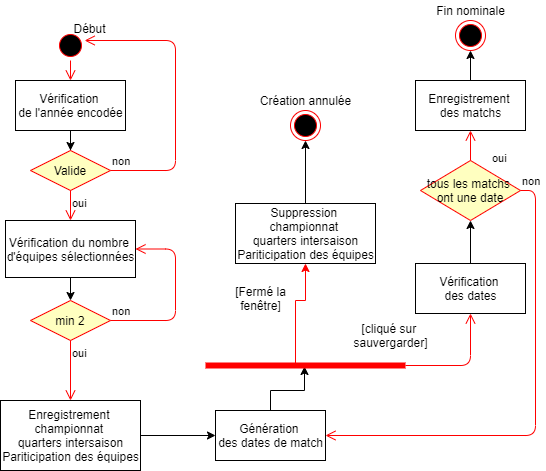
Ne sont affichés que les matchs pour le championnat généré sur la form précédente.

Les matchs sont divisés entre les 2 quarters de manière « équitable » (une équipe ne fait pas tous ses matchs à domicile en un quarter mais ils sont distribués entre les 2)

Les dates de matchs sont générées automatiquement pour autant qu’une équipe ne joue qu’une fois sur le weekend. Les matchs supplémentaires ont pour date par défaut 1/1/1801 pour le premier quarter et 1/1/1802 pour le 2ème quarter.

Dans tous les cas, une équipe ne peut pas jouer 2 matchs le même jour ou 2 jours de suite.

#### Diagramme d’activité : Générer un championnat



### Use Case : Transférer un joueur

#### Description textuelle– Transférer un joueur

Résumé : permet à l’utilisateur BackEnd de transférer des joueurs d’un club à l’autre

Acteurs : BackEnd *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application BackEnd est lancée
* L’utilisateur a cliqué sur le bouton « Transférer un joueur »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel demande à la base de données toutes les informations de transferts de joueurs.
2. La base de données renvoie les informations
3. Le logiciel affiche les données nécessaires pour l’utilisateur
4. L’utilisateur encode ses transferts (équipes de destination et date)
5. L’utilisateur appui sur le bouton « enregistrer et rafraichir »
6. Le logiciel vérifie que les dates remplissent toutes les conditions
   1. Une date de transfert doit être encodée pour qu’un transfert ait lieu
   2. Le joueur ne peut pas avoir participé à un match à une date ultérieure
   3. La date de transfert ne doit pas être comprise dans un quarter
   4. Vérifie que les équipes d’arrivée et de départ remplissent toujours les conditions de nombre de joueurs
   5. Vérifie si le transfert a lieu lors d’une intersaison que l’équipe de destination fasse bien partie des 3 derniers du classement
7. Le logiciel clôture l’ancien transfert s’il existait dans la base de données
8. Le logiciel enregistre le nouveau transfert
9. Le logiciel demande à la base de données toutes les informations de transferts de joueurs.
10. La base de données renvoie les informations
11. Le logiciel affiche les données nécessaires pour l’utilisateur

Scénarios alternatifs :

*A1 :* L’utilisateur encode uniquement une date de transfert

1. Le logiciel vérifie que les dates remplissent toutes les conditions
   1. Une date de transfert doit être encodée pour qu’un transfert ait lieu
   2. Le joueur ne peut pas avoir participé à un match à une date ultérieure
   3. La date de transfert ne doit pas être comprise dans un quarter
   4. Vérifie que l’équipe de départ remplisse toujours les conditions de nombre de joueurs
2. Le logiciel clôture l’ancien transfert s’il existait dans la base de données
3. Le logiciel demande à la base de données toutes les informations de transferts de joueurs.
4. La base de données renvoie les informations
5. Le logiciel affiche les données nécessaires pour l’utilisateur

Enchainements d’erreurs :

*E1 :* L’utilisateur a indiqué une équipe d’arrivée mais pas de date de transfert

1. L’utilisateur appui sur le bouton « enregistrer et rafraichir »
2. Le logiciel affiche un message d’erreur spécifique
3. Le scénario nominal reprend au point 4

*E2 :* L’utilisateur a indiqué une date qui est comprise dans un quarter ou une date antérieure à une participation du joueur.

1. L’utilisateur encode ses transferts (équipes de destination et date)
2. Le logiciel affiche un message d’erreur spécifique
3. Le scénario nominal reprend au point 4

*E3 :* L’utilisateur a indiqué une date qui ne vérifie pas les conditions de nombres de joueurs dans les équipes ou pour laquelle il y a des matchs non- joués pour une des 2 équipes ou qui est dans une intersaison et qui ne vérifie pas les conditions de classement

1. L’utilisateur appui sur le bouton « enregistrer et rafraichir »
2. Le logiciel affiche un message d’erreur spécifique
3. Le scénario nominal reprend au point 4

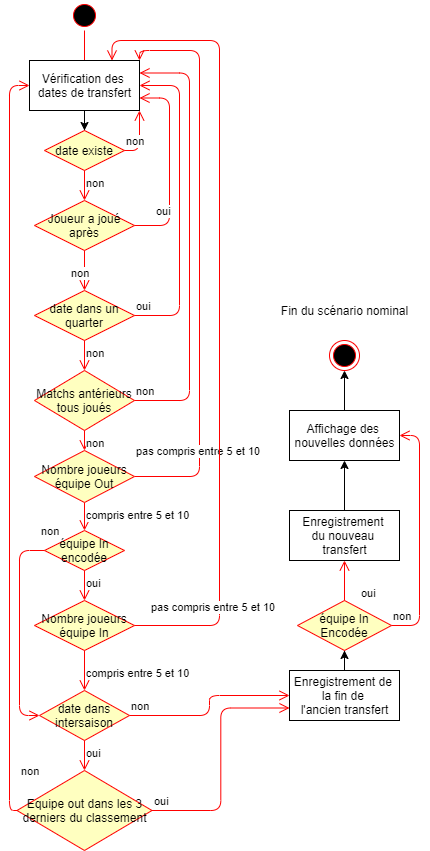
#### Mockup ecran – Transférer un joueur

Une image contenant capture d’écran

Description générée automatiquement

La date de transfert est au minimum un jour après la date in, s’il y en a une sinon elle est celle du jour. Pour rappel, une équipe ne peut être inscrite à un championnat si elle a déjà un transfert encodé la même année que celle du championnat.

#### Diagramme d’activité : Transférer un joueur



### Use Case : Visualiser le classement par joueur

#### Description textuelle– Visualiser le classement par joueur

Résumé : permet à l’utilisateur BackEnd de visualiser le classement par joueur

Acteurs : BackEnd *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application BackEnd est lancée
* L’utilisateur a cliqué sur le bouton « Visualiser le classement par joueur »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement pour le premier championnat par goal
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement par goal

Scénarios alternatifs :

*A1 :* L’utilisateur choisit dans la liste déroulante un autre championnat

1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement du championnat sélectionné par goal
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement par goal

*A2 :* L’utilisateur choisit dans la liste déroulante un type de classement

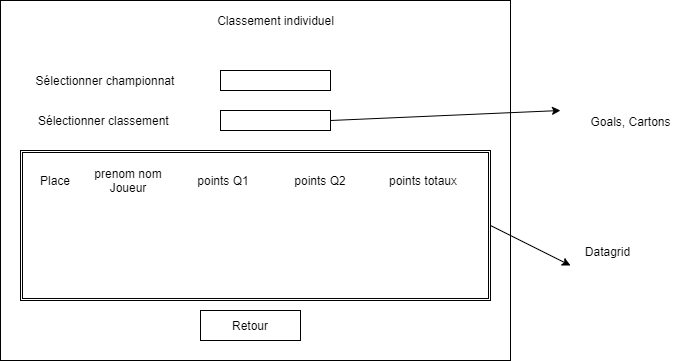
1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement du championnat sélectionné avec le type sélectionné
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement avec le type sélectionné

Enchainements d’erreurs :

*E1 :* Aucun championnat n’existe dans la base de données

1. Le logiciel affiche un message d’erreur
2. Le logiciel ferme la form

#### Mockup ecran – Visualiser le classement par joueur

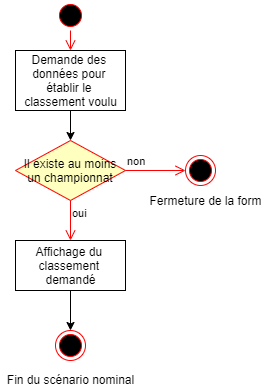


Affiche le classement des joueurs par goals ou cartons pour un championnat sélectionné dans la liste par l’utilisateur. Il est uniquement affiché la liste des joueurs inscrits dans une équipe qui est dans le championnat.

Il est à noter que les cartons rouges valent 3 cartons jaunes pour le classement.

Il ne s’agit que d’un tableau en lecture.

#### Diagramme d’activité : Visualiser le classement par joueur



### Use Case : Visualiser le classement par équipe

#### Description textuelle– Visualiser le classement par équipe

Résumé : permet à l’utilisateur BackEnd de visualiser le classement par équipe

Acteurs : BackEnd *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application BackEnd est lancée
* L’utilisateur a cliqué sur le bouton « Visualiser le classement par équipe »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement pour le premier championnat par points
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement par points

Scénarios alternatifs :

*A1 :* L’utilisateur choisit dans la liste déroulante un autre championnat

1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement du championnat sélectionné par points
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement par points

*A2 :* L’utilisateur choisit dans la liste déroulante un type de classement

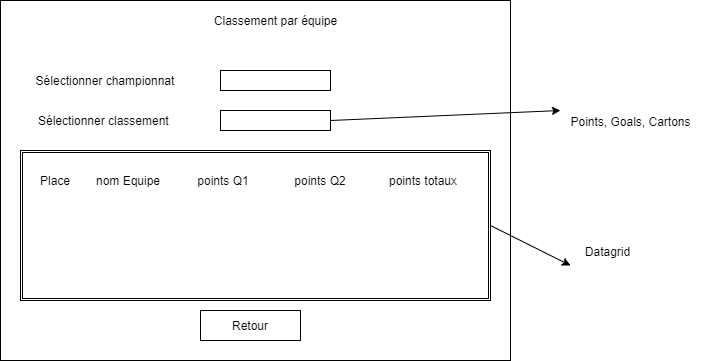
1. Le logiciel interroge la base de données pour récupérer toutes les informations nécessaires pour établir le classement du championnat sélectionné avec le type sélectionné
2. La base de données renvoie toutes les données
3. Le logiciel affiche le classement avec le type sélectionné

Enchainements d’erreurs :

*E1 :* Aucun championnat n’existe dans la base de données

1. Le logiciel affiche un message d’erreur
2. Le logiciel ferme la form

#### Mockup ecran – Visualiser le classement par équipe

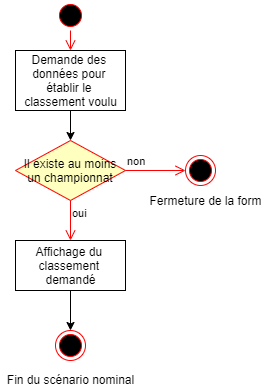


Affiche le classement des équipes par points, goals ou cartons pour un championnat sélectionné dans la liste par l’utilisateur. Il est uniquement affiché les équipes qui sont inscrites dans le championnat sélectionné.

Il est à noter que les cartons rouges valent 3 cartons jaunes pour le classement.

Il ne s’agit que d’un tableau en lecture.

#### Diagramme d’activité : Visualiser le classement par joueur



### Use Case : Visualiser & mettre à jour un match

#### Description textuelle– Visualiser & mettre à jour un match

Résumé : permet à l’utilisateur BackEnd de visualiser & modifier un match

Acteurs : BackEnd *(primaire)* et MatchManagement *(secondaire)*

Préconditions :

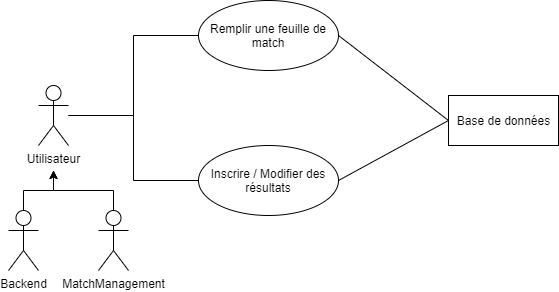
* L’application BackEnd est lancée
* L’utilisateur a cliqué sur le bouton « Visualiser le classement par équipe »
* Le programme MatchManagement est fonctionnel

Scénario nominal :

1. Le logiciel lance l’application MatchManagement avec tous les droits de l’utilisateur MatchManagement
2. L’écran de départ de MatchManagement s’affiche dans le MDI de BackEnd.

Pour la suite se référer à la partie MatchManagement

## Matchmanagement – Diagramme de Use Cases



### Mockup : accueil



En sélectionnant le championnat, le datagridview affiche tous les matchs du championnat.

Ne peuvent être modifiés au niveau de la feuille de match que les matchs qui n’ont pas encore été joués (pour lesquels une carte ou un goal n’a pas encore été inscrit) et pour lesquels aucun match précédent d’une des 2 équipes est « non joué » ou postérieure « joué ».

La feuille de résultat peut toujours être modifiée mais uniquement au niveau des goals si le match est considéré comme joué.

### Use Case : Remplir une feuille de match

#### Description textuelle – Remplir une feuille de match

Résumé : permet à l’utilisateur MatchManagement ou BackEnd de remplir une feuille de match

Acteurs : MatchManagement *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application MatchManagement est lancée
* L’utilisateur a sélectionné un match
* L’utilisateur a cliqué sur le bouton « Remplir une feuille de match »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel ouvre la fenêtre « feuilles de match »
2. Le logiciel demande les informations nécessaires à la base de données
3. La base de données répond
4. Le logiciel affiche tous les joueurs de chaque équipe, le nombre de cartons jaunes actifs, le nombre de suspensions restantes dû à des cartons rouges et le nombre de match restant dans le quarter pour l’équipe
5. L’utilisateur sélectionne entre 5 et 7 joueurs valides dans les 2 équipes
6. L’utilisateur clique sur le bouton « save »
7. Le logiciel enregistre dans la base de données les 2 feuilles de matchs
8. Le logiciel ferme la fenêtre
9. Au prochain rafraichissement de la fenêtre d’accueil le logiciel affichera que la fiche de match est remplie

Scénario alternatif :

*A1 :* L’utilisateur sélectionne moins de 5 joueurs pour une ou les 2 équipes volontairement

1. L’utilisateur clique sur le bouton « save »
2. Le logiciel affiche un message d’information demandant confirmation
3. L’utilisateur confirme
4. Le logiciel enregistre dans la base de données les 2 feuilles de matchs
5. Le logiciel ferme la fenêtre
6. Au prochain rafraichissement de la fenêtre d’accueil le logiciel affichera que la fiche de match n’est pas remplie mais que le match est joué

*A2*: L’utilisateur sélectionne une feuille de match incomplète (indiqué par le fait que la feuille de match n’est pas remplie mais que le match est joué)

1. L’utilisateur sélectionne entre 5 et 7 joueurs valides dans les 2 équipes
2. L’utilisateur clique sur le bouton « save »
3. Le logiciel enregistre dans la base de données les 2 feuilles de matchs
4. Le logiciel ferme la fenêtre
5. Au prochain rafraichissement de la fenêtre d’accueil le logiciel affichera que la fiche de match est remplie et que le match n’est pas joué

*A3*: L’utilisateur souhaite quitter l’application

1. L’utilisateur clique sur le bouton « Back »
2. Le logiciel ferme la fenêtre et revient à l’accueil

Enchainements d’erreurs :

*E1 :* L’utilisateur a choisi un match pour lequel la feuille de match est déjà remplie

1. Le logiciel affiche un message d’erreur

*E2 :* L’utilisateur sélectionne moins de 5 joueurs pour une ou les 2 équipes non-volontairement

1. L’utilisateur clique sur le bouton « save »
2. Le logiciel affiche un message d’information demandant confirmation
3. L’utilisateur infirme
4. Le scénario revient au point 5 du scénario nominal

*E3 :* L’utilisateur a choisi un match pour lequel il y a déjà un match joué postérieurement par une des 2 équipes au moins

1. Le logiciel affiche un message d’erreur

*E4 :* L’utilisateur a choisi un match pour lequel il a déjà un match antérieur d’une des 2 équipes au moins n’a pas été joué

1. Le logiciel affiche un message d’erreur

#### Mockup ecran – Remplir une feuille de match

Une image contenant capture d’écran

Description générée automatiquement

Seuls les joueurs faisant partie de l’équipe sont affichés dans la liste des joueurs. Un checkbox permet de les sélectionner facilement pour le match.

Les joueurs suspendus à la suite d’un carton rouge sont affichés en rouge et ne peuvent pas être inscrits

Les joueurs suspendus à la suite d’un carton jaune sont affichés en jaunes et ne peuvent pas être inscrits

Un message, avant sauvegarde, demande confirmation si les quotas de joueurs ne sont pas atteints.

#### Diagramme d’activité : Remplir une feuille de match

Une image contenant carte, texte

Description générée automatiquement

### Use Case : Inscrire / modifier les résultats

#### Description textuelle – Inscrire les résultats

Résumé : permet à l’utilisateur MatchManagement ou BackEnd de remplir les résultats d’un match

Acteurs : MatchManagement *(primaire)* et la base de données *(secondaire)*

Préconditions :

* L’application MatchManagement est lancée
* L’utilisateur a sélectionné un match
* L’utilisateur a cliqué sur le bouton « Inscrire / modifier les résultats »
* La base de données existe et est valide (accès, structure, …)

Scénario nominal :

1. Le logiciel ouvre la fenêtre « Résultats »
2. Le logiciel demande les informations nécessaires à la base de données
3. La base de données répond
4. Le logiciel affiche les goals, cartons jaunes et rouges qui ont déjà été enregistrés dans le match pour les 2 équipes
5. L’utilisateur remplit la minute et le joueur concerné par un évènement de match
6. L’utilisateur clique sur le bouton « sauvegarder »
7. Le logiciel vérifie que pour chaque champ minute un joueur soit choisi et le contraire
8. Le logiciel vérifie que chaque champ minute soit compris entre 0 et 120
9. Le logiciel vérifie qu’un joueur qui a reçu un carton rouge n’a pas marqué après ou reçu une carte jaune
10. Le logiciel affiche un message d’information prévenant l’utilisateur du remplissage unique des cartons
11. L’utilisateur confirme
12. Le logiciel enregistre dans la base de données les résultats du match
13. Le logiciel ferme la fenêtre
14. Au prochain rafraichissement de la fenêtre d’accueil le logiciel affichera que le match est joué.

Scénario alternatif :

*A1 :* L’utilisateur a choisi un match pour lequel la feuille de match n’est pas remplie (ou incomplète)

1. L’utilisateur ne peut modifier aucuns champs.

A2 : L’utilisateur a choisi un match pour lequel un transfert a eu lieu après le match (lors du championnat et pendant l’intersaison) et impliquant un joueur d’une des équipes du championnat

1. L’utilisateur ne plus modifier les goals, un message d’information est affiché

*A3 :* L’utilisateur a choisi un match pour lequel une feuille de match a été crée (lors du championnat) à une date postérieure au match

1. L’utilisateur ne peut pas modifier les cartons, un message d’information est affiché

Enchainements d’erreurs :

*E1 :* L’utilisateur ne remplit pas les champs joueurs et minutes pour chacune des lignes

1. Le logiciel affiche un message d’erreur
2. Le scénario nominal reprend au point 5

*E2 :* L’utilisateur donne un carton jaune à un joueur après qu’il a reçu un carton rouge

1. Le logiciel affiche un message d’erreur
2. Le scénario nominal reprend au point 5

*E3 :* L’utilisateur attribue un goal à un joueur après qu’il a reçu un carton rouge

1. Le logiciel affiche un message d’erreur
2. Le scénario nominal reprend au point 5

*E4 :* L’utilisateur encode une minute d’évènement non comprise entre 0 et 120

1. Le logiciel affiche un message d’erreur
2. Le scénario nominal reprend au point 5

#### Mockup ecran – Inscrire / modifier les résultats

Une image contenant capture d’écran

Description générée automatiquement

Le champ Joueur reprend le prénom et le nom du joueur qui a déjà été inscrit pour un goal ou carte.

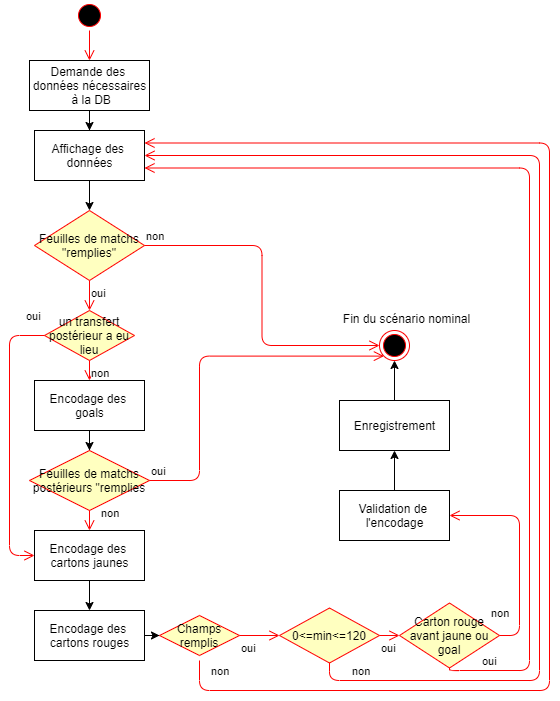
Changer le joueur est une combobox ne comprenant que les joueurs inscrits sur la feuille de match et permet de modifier goal ou carton ou d’en rajouter un.

La case temps doit être comprise entre 0 et 120

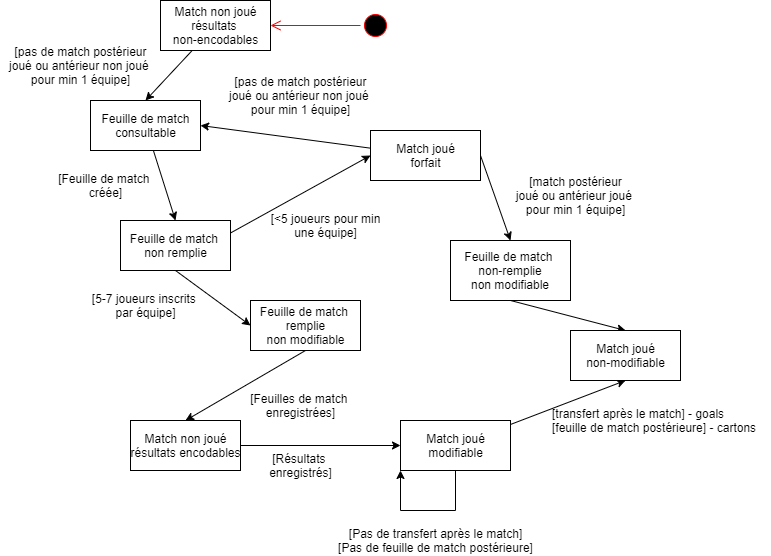
Les cartes ne peuvent être remplies que la première fois où on sauvegarde un résultat. En cas de forfait, il n’est pas possible de remplir les champs goals et cartons.

Une vérification est faite avant l’enregistrement pour voir si un joueur n’a pas marqué ou reçu un carton jaune après avoir reçu un carton rouge.

#### Diagramme d’activité : Inscrire / modifier les résultats



## Matchmanagement – diagramme d’état



# Contraintes fonctionnelles

## Règles d’accès et autorisations

|  |  |
| --- | --- |
| FM-RA001 | L’application BackEnd est uniquement accessible à l’utilisateur BackEnd |
| FM-RA002 | L’application BackEnd donne accès à l’application MatchManagement et aux accès de l’utilisateur MatchManagement |
| FM-RA003 | L’application MatchManagement est accessible à l’utilisateur BackEnd et à l’utilisateur BackEnd via l’application BackEnd |

## Règles de structures

### Championnats

|  |  |
| --- | --- |
| FM-RS001 | Un championnat se déroule sur une année civile |
| FM-RS002 | Un championnat est divisé en 2 quarters séparés par une intersaison |
| FM-RS003 | Le nombre d’équipes inscrites varie d’une année à l’autre |

### Quarters

|  |  |
| --- | --- |
| FM-RS004 | Un quarter dure 5 semaines avec une date de fin et de début |

### Intersaisons

|  |  |
| --- | --- |
| FM-RS005 | Une intersaison dure 6 mois avec une date de fin et de début |

### EquipesParticipation

|  |  |
| --- | --- |
| FM-RS006 | La participation concerne une équipe |
| FM-RS007 | La participation concerne un championnat |

### équipes

|  |  |
| --- | --- |
| FM-RS008 | Une équipe possède un nom |
| FM-RS009 | Une équipe peut posséder un logo |
| FM-RS010 | Une équipe peut s’inscrire à des championnats |
| FM-RS011 | Une équipe peut jouer des matches dans les championnats dans lesquelles elle est inscrite |
| FM-RS012 | Une équipe peut posséder des joueurs |

### Joueurs

|  |  |
| --- | --- |
| FM-RS013 | Un joueur possède un nom |
| FM-RS014 | Un joueur possède un prénom |
| FM-RS015 | Un joueur peut être inscrit à une équipe |
| FM-RS016 | Un joueur peut être inscrit à un match |
| FM-RS017 | Un joueur peut marquer des goals dans les matchs où il est inscrit |
| FM-RS018 | Un joueur peut recevoir des cartons jaunes dans les matchs où il est inscrit |
| FM-RS019 | Un joueur peut recevoir des cartons rouges dans les matchs où il est inscrit |

### Transferts

|  |  |
| --- | --- |
| FM-RS020 | Un transfert concerne un joueur |
| FM-RS021 | Un transfert concerne une équipe |
| FM-RS022 | Un transfert a une date de début |
| FM-RS023 | Un transfert peut avoir une date de fin |

### Goals

|  |  |
| --- | --- |
| FM-RS024 | Un goal est marqué par un joueur |
| FM-RS025 | Un goal est marqué pendant un match |

### Cartons jaunes

|  |  |
| --- | --- |
| FM-RS026 | Un carton jaune est donné à un joueur |
| FM-RS027 | Un carton jaune est donné pendant un match |

### Cartons Rouges

|  |  |
| --- | --- |
| FM-RS028 | Un carton rouge est donné à un joueur |
| FM-RS029 | Un carton rouge est donné pendant un match |

### Matchs

|  |  |
| --- | --- |
| FM-RS030 | Un match est joué par 2 équipes |
| FM-RS031 | Un match est joué à une date définie lors d’un quarter |
| FM-RS032 | Lors d’un match, chaque équipe gagne un certain nombre de points en fonction du résultat |

### Feuille de match

|  |  |
| --- | --- |
| FM-RS033 | Une feuille de match est liée à un match |
| FM-RS034 | Une feuille de match est liée à une équipe |

### JoueursParticipation

|  |  |
| --- | --- |
| FM-RS035 | La participation concerne un joueur |
| FM-RS036 | La participation concerne une feuille de match |

## Règles de validation

De manière générale (excepté

### Championnats

|  |  |
| --- | --- |
| FM-RV001 | Il ne peut y avoir qu’un championnat par année civile |

### Quarters

|  |  |
| --- | --- |
| FM- RV002 | Il ne peut y avoir que 2 quarters pour le même championnat |
| FM- RV003 | Un quarter doit se dérouler sur une seule année civile |

### Intersaisons

|  |  |
| --- | --- |
| FM- RV004 | Il ne peut y avoir d’une intersaison pour un championnat |
| FM- RV005 | Une intersaison doit se dérouler sur une seule année civile |

### EquipesParticipation

|  |  |
| --- | --- |
| FM- RV006 | Pour pouvoir s’inscrire dans un championnat une équipe doit être composée au minimum de 5 joueurs. |
| FM- RV007 | La participation d’une équipe est unique à un championnat |
| FM- RV008 | Une équipe ne peut pas avoir transfert pendant l’année d’un championnat avant son inscription à celui-ci |

### équipes

|  |  |
| --- | --- |
| FM- RV009 | Une équipe possède un nom unique |

### Joueurs

|  |  |
| --- | --- |
| FM- RV010 | Un joueur possède un nom et un prénom |

### Transferts

|  |  |
| --- | --- |
| FM- RV011 | Une équipe ne peut posséder plus de 10 joueurs sans date de fin |
| FM- RV012 | Un joueur ne peut pas jouer dans 2 équipes en même temps |
| FM- RV013 | Un transfert ne peut pas avoir lieu si des matchs antérieurs (du championnat) n’ont pas été joués |
| FM- RV014 | Un transfert à l’intersaison ne peut avoir lieu que vers une équipe classée dans les 3 dernières équipes du classement |
| FM- RV015 | Un transfert lors de l’intersaison doit s’assurer qu’il reste minimum 5 joueurs dans l’équipe de départ |
| FM- RV016 | Un transfert ne peut pas avoir lieu lors d’un quarter |
| FM- RV017 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |

### Goals

|  |  |
| --- | --- |
| FM- RV018 | Un goal ne peut être marqué que par un joueur qui participe au match |
| FM- RV019 | Pour qu’un goal soit enregistré, il faut que les deux feuilles du match existent et comptent minimum 5 joueurs |
| FM- RV020 | Un goal est marqué entre la minute 0 et la minute 120 |
| FM- RV021 | Un goal ne peut être marqué par un joueur que s’il n’a pas reçu de carton rouge avant dans le même match |
| FM- RV022 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |

### Cartons jaunes

|  |  |
| --- | --- |
| FM- RV023 | Un carton jaune est donné entre la minute 0 et la minute 120 |
| FM- RV024 | Un carton jaune ne peut être donné qu’à un joueur qui participe au match |
| FM- RV025 | Pour qu’un carton jaune soit enregistré, il faut que les deux feuilles du match existent et comptent minimum 5 joueurs |
| FM- RV026 | Un carton jaune ne peut être reçu par un joueur que s’il n’a pas reçu de carton rouge avant dans le même match |
| FM- RV027 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |

### Cartons Rouges

|  |  |
| --- | --- |
| FM- RV028 | Un carton rouge est donné entre la minute 0 et la minute 120 |
| FM- RV029 | Un carton rouge ne peut être donné qu’à un joueur qui participe au match |
| FM- RV030 | Pour qu’un carton rouge soit enregistré, il faut que les deux feuilles du match existent et comptent minimum 5 joueurs |
| FM- RV031 | Un seul carton rouge peut être donné par joueur par match |
| FM- RV032 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |

### Matchs

|  |  |
| --- | --- |
| FM-RS033 | Une équipe ne peut pas jouer contre soi-même |
| FM-RS034 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |
| FM-RS035 | Les dates des matchs doivent être comprises dans les quarter du championnat |
| FM-RS036 | Il ne peut pas avoir 2 fois le même match (A-B et A-B) |
| FM-RS037 | Si un carton jaune est encodé pour le match, le match est considéré comme joué |
| FM-RS038 | Si un carton rouge est encodé pour le match, le match est considéré comme joué |
| FM-RS039 | Si un goal est encodé pour le match, le match est considéré comme joué |

### Feuille de match

|  |  |
| --- | --- |
| FM-RS040 | Il ne peut avoir qu’une feuille de match par équipe et par match |
| FM-RS041 | Toute mise à jour réalisée doit s’assurer d’être plus récente que la version enregistrée dans la base de données |

### JoueursParticipation

|  |  |
| --- | --- |
| FM-RS042 | Il ne peut y avoir plus de 7 joueurs inscrits |
| FM-RS043 | Un joueur ne peut être inscrit que s’il joue au moment du match pour l’équipe |
| FM-RS044 | Un joueur qui a reçu un carton rouge dans les 3 matchs antérieurs ne peut pas être inscrit |
| FM-RS045 | Un joueur qui a autant ou plus de cartons jaunes actifs (voir règle dans les règles de calcul) que de matchs restants dans le quarter ne peut pas être inscrit |

## Règles de calcul

### Matchs

|  |  |
| --- | --- |
| FM-RC001 | En cas de victoire d’une des 2 équipes, la victorieuse gagne 3 points, celle qui a perdu 0 points |
| FM-RC002 | En cas de nul, chaque équipe gagne 1 point |
| FM-RC003 | Si au moins une des 2 feuilles de match comporte moins de 5 joueurs, le match est considéré comme forfait |
| FM-RC004 | En cas de forfait, l’équipe qui a inscrit le moins de joueur perds le match |
| FM-RC005 | En cas de forfait, si le nombre de joueurs inscrit par les 2 équipes est égal alors le match est considéré comme nul. |
| FM-RC006 | Les matchs générés sont distribués automatiquement entre les 2 quarters de manière « équitable » entre les matchs à domicile et les matchs à l’extérieur |
| FM-RC007 | Un match du 1er quarter ne peut être mis au 2ème et vice versa |
| FM-RC008 | Les matchs générés sont répartis lors des week-end |
| FM-RC009 | Les matchs générés ne peuvent pas faire jouer une même équipe 2 fois le même weekend |
| FM-RC010 | En cas de nombre de weekend insuffisant, c’est à l’utilisateur de choisir les dates |

### JoueursParticipations

|  |  |
| --- | --- |
| FM-RC007 | Si un joueur reçoit un carton rouge, il ne peut plus participer aux 3 prochains matchs (dans le même quarter) |
| FM-RC007 | Un joueur qui a autant ou plus de cartons jaunes actifs que de matchs restants dans le quarter ne peut pas être inscrit |
| FM-RC008 | Par quarter, les cartons jaunes actifs sont calculés par le nombre de cartons jaunes reçus moins les matchs non-joués par le joueur (sans que cela soit dû à une carte rouge) |
| FM-RC009 | A la fin de chaque quarter, le nombre de cartons jaunes et rouges reprend à 0 |

# Description des entités

## Championnats

Un championnat se déroule sur une année civile.

Il est divisé en 2 quarters de 5 semaines séparés par une intersaison de 6 mois.

Le nombre d’équipes inscrites varie d’une année à l’autre.

Pour pouvoir s’inscrire dans un championnat une équipe doit être composée au minimum de 5 joueurs.

## Équipes

Une équipe est composée de 5 à 10 joueurs et possède un nom et peut avoir un logo

Une équipe peut s’inscrire à des championnats.

## Joueurs

Un joueur possède un nom, un prénom

Un joueur est inscrit ou non dans une équipe

Il peut posséder plusieurs cartes jaunes et au maximum une carte rouge.

S’il possède une carte rouge active, il ne peut pas disputer les 3 prochains matchs

S’il possède le même nombre ou plus de cartes jaunes actives que de matchs restants dans l’intersaison, il ne peut plus jouer pendant le quarter.

S’il manque un match et :

* Qu’il a un carton rouge avec des suspensions
* Qu’il n’a pas de carton rouge avec des suspensions et qu’il a un ou plusieurs carton jaunes actifs, le carton jaune le plus ancien est désactivé
* Qu’il n’ait ni carton rouge avec suspensions ni un carton jaune actif alors rien ne se passe

À la fin d’un quarter tous les cartons jaunes sont désactivés et les suspensions pour les cartons rouges sont remises à 0.

## Transferts

Un joueur ne peut pas changer d’équipe lors d’un quarter.

Un joueur ne peut changer d’équipe à l’intersaison que si son équipe possède plus de 5 joueurs et que l’équipe dans laquelle il va est classée parmi les 3 dernières du championnat.

Un joueur peut changer d’équipe entre 2 championnats.

## Matchs

Les matchs ne se jouent que le samedi ou le dimanche sauf s’il y a trop de match pour le nombre de week-end. Une équipe ne joue qu’une fois par week-end.

Un match est joué par 2 équipes. Chaque équipe remplit une feuille de match.

Si la feuille de match d’une équipe compte moins de 5 joueurs et :

* Moins de joueurs que celle de l’adversaire alors elle perd automatiquement le match.
* Le même nombre de joueur que celle de l’adversaire alors c’est un match nul.

Un match possède un résultat et donne un certain nombre de points par équipe :

* Gagné : 3 points
* Perdu : 0 point
* Nul : 1 point

Lors d’un match, un joueur peut marquer un goal ou prendre un carton jaune ou rouge.

## Feuilles de match

Une feuille de match peut compter 7 joueurs maximum et est unique par match et par équipe.

Un joueur ne peut être inscrit que s’il est dans l’équipe.

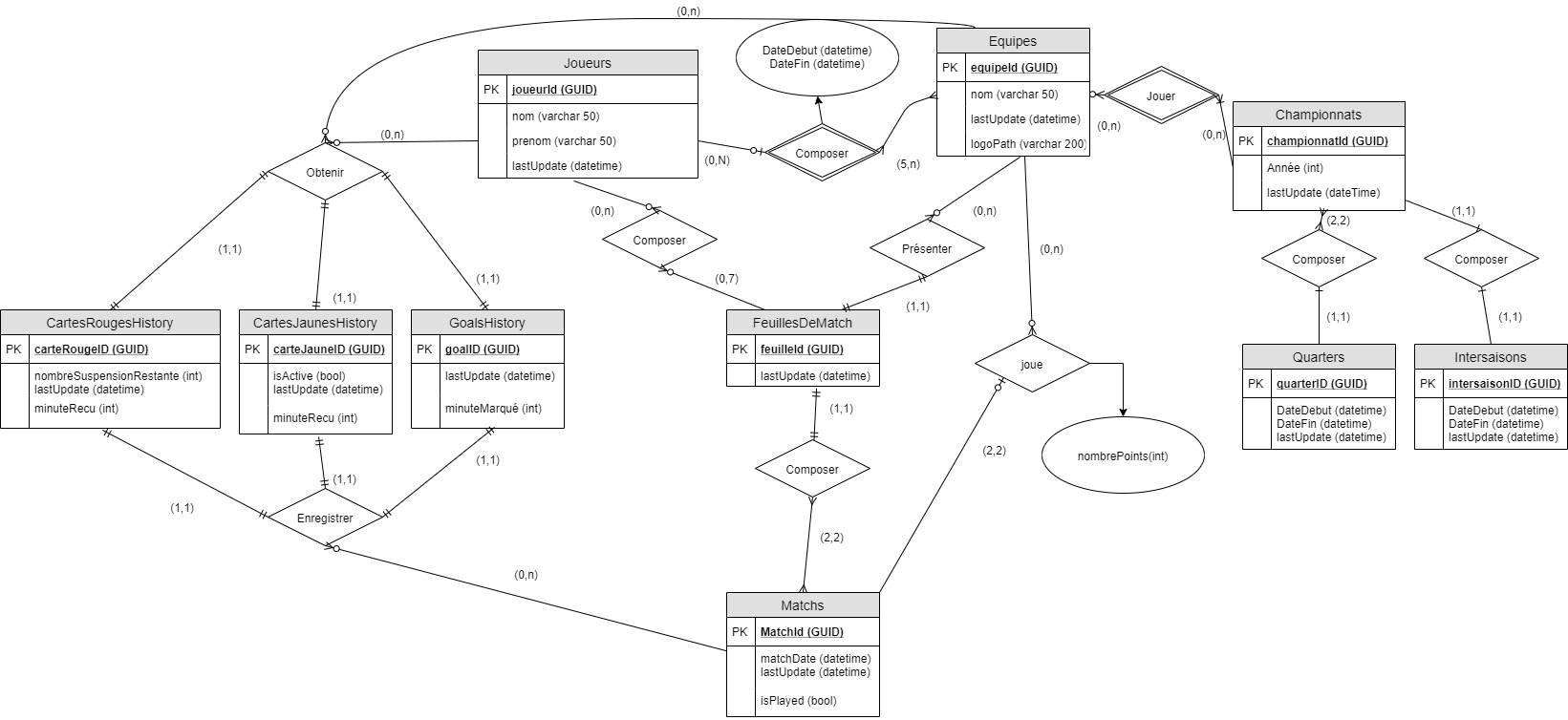
Un joueur qui a une carte rouge « active » ne peut pas être inscrit sur la feuille de match.

Si un joueur à le même nombre de carte jaune que de match restant à jouer pour son équipe, il ne peut pas être inscrit sur la feuille de match.

# Schéma relationnel de la solution

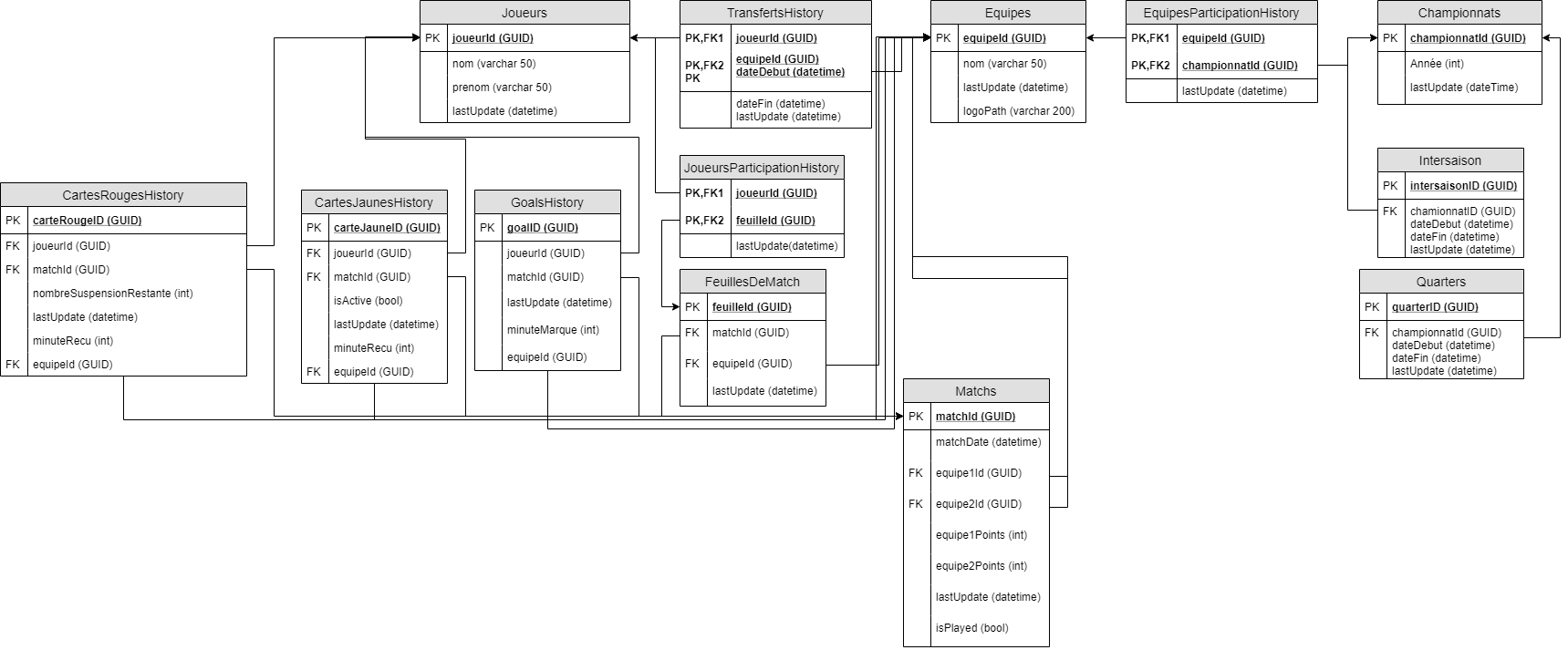
## Entité Association

Vous trouverez ci-dessous l’entité association proposée pour ce projet.



## Schéma Relationnel

Vous trouverez ci-dessous le schéma relationnel proposé pour ce projet.



## Implémentation des contraintes

|  |  |  |
| --- | --- | --- |
| **Numéro** | **Règle** | **Où ?** |
| R1 | Un championnat possède une année | BDD : Contrainte not null FORM : Générer un championnat |
| R2 | Il ne peut y avoir qu'un championnat par année | BDD : Contrainte unique BDD : BackEnd.Championnat\_Add (PS) BL : ChampionnatsService FORM : Générer un championnat |
| R3 | L'année d'un championnat est supérieure ou égale à 1900 | BDD : FormatAnneesChampionnats (Trigger) FORM : Générer un championnat |
| R4 | Il ne peut y avoir qu'une intersaison par championnat | BDD : Contrainte unique BDD : BackEnd.Intersaisons.Add (PS) BDD : Intersaisons\_UniciteChampionnat (Trigger) FORM : Générer un championnat |
| R5 | Pour chaque intersaison, l'année de date de début, de fin et du championnat doivent être identiques | BDD : BackEnd.Intersaisons.Add (PS) FORM : Générer un championnat |
| R6 | Il ne peut y avoir que 2 quarters maximum par championnat | BDD : BackEnd.Quarters.Add  (PS) BDD : MaxQuarters (Trigger) FORM : Générer un championnat |
| R7 | Pour chaque quarter, l'année de date de début, de fin et du championnat doivent être identiques | BDD : BackEnd.Quarters.Add (PS) FORM : Générer un championnat |
| R8 | le nom d'une équipe doit être unique | BDD : Contrainte unique BDD : Equipes\_UniciteNom (Trigger) |
| R9 | logoPath si null est généré automatiquement en noLogo | BDD : Valeur par défaut noLogo |
| R10 | Une équipe ne peut participer qu'une fois à un championnat | BDD : Contrainte unique equipeId / championnatId BDD : BackEnd.EquipeParticipation\_Add (PS) FORM : Générer un championnat |
| R11 | Une équipe doit avoir assez de joueurs pour s'inscrire à un championnat | BDD : BackEnd.EquipeParticipation\_Add (PS) BDD : EquipesParticipation\_EquipeActive (Trigger) BL : GenerationTabEquipeSelection FORM : Générer un championnat |
| R12 | Une équipe ne peut pas avoir des transferts dans l'année du championnat avant son inscription au championnat | BL : GenerationTabEquipeSelection FORM : Générer un championnat |
| R13 | Un joueur possède un nom et un prénom | BDD : Contrainte not null |
| R14 | Les points de chaque équipe lors d’un match sont initialisés à 0 | BDD : Valeur par défaut 0 |
| R15 | Un booleen isPlayed décrit si un match a été joué (non par défaut) | BDD : Valeur par défaut 0 |
| R16 | Un match possède une date qui doit être comprise dans un quarter | BDD : BackEnd.Matchs\_Add (PS) BDD : BackEnd et MatchManagement.Matchs\_Update (PS) BDD : Matchs\_WithinChampionnat (Trigger) BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R17 | Un match ne doit pas avoir 2x la même équipe | BDD : BackEnd.Matchs\_Add (PS) BDD : BackEnd et MatchManagement.Matchs\_Update (PS) BDD : Matchs\_SameTeams  (Trigger) BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R18 | Un match avec la combinaison équipe1 - équipe2 doit être unique dans un championnat | BDD : BackEnd.Matchs\_Add (PS) BDD : BackEnd et MatchManagement.Matchs\_Update (PS) BDD : Matchs\_Doublons   (Trigger) BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R19 | Les matchs à domicile et à l'extérieur d'une équipe lors d'un championnat sont répartis équitablement entre les 2 quarters | BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R20 | Les matchs se jouent par défaut le weekend, une équipe ne peut jouer qu'une fois par weekend. En cas de trop de match par rapport au dates disponibles, l'utilisateur choisit les dates | BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R21 | Une équipe ne pas jouer 2 matchs le même jour ou à un jour d'intervalle | BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R22 | Un match du premier quarter, ne peut être assigné par l'utilisateur que dans le 1er quarter | BL : GenerationTabCalendrierMatchs FORM : Calendrier des matchs |
| R23 | Toute modification d'un match doit s'assurer que le match n'a pas déjà reçu une modification plus récente | BDD : Matchs\_UpdatePlusVieux (Trigger) |
| R24 | Un match est joué dès qu'un carton jaune est encodé et les points sont initialisés | BDD : CartonsJaunesHistory\_PlayedMatch (Trigger) BL : CartesJaunesService |
| R25 | Un match est joué dès qu'un carton rouge est encodé et les points sont initialisés | BDD : CartonsRougesHistory\_PlayedMatch (Trigger) BL : CartesRougesService |
| R26 | Un match est joué dès qu'un goal est inscrit et les points sont comptés | BDD : GoalsHistory\_PlayedMatch (Trigger) BL : GoalsService |
| R27 | Si une des 2 feuilles de match compte moins de 5 joueurs alors l’autre équipe gagne automatiquement | BDD : JoueursParticipationHistory\_CheckFeuillesDeMatch (Trigger) FORM : Feuille de match |
| R28 | Si une des 2 feuilles de match compte moins de 5 joueurs alors le match est considéré comme joué | BL : CheckConditionsResultats |
| R29 | Si les 2 équipes ont moins de 5 joueurs sur leur feuille de match, celle avec le plus de joueur gagne le match | BDD : JoueursParticipationHistory\_CheckFeuillesDeMatch (Trigger) FORM : Feuille de match |
| R30 | Les cartons rouges ne peuvent être reçus qu'entre la minute 0 et 120 d'un match | BDD: CartonsRougesHistory\_LimiteMinuteRecue (Trigger) FORM : Inscriptions des résultats |
| R31 | A l'attribution d'un carton rouge, un joueur est suspendu pour 3 matchs | BL : GenerationTableauxFeuille FORM : Feuille de Match FORM : Inscriptions des résultats |
| R32 | Une carte rouge ne peut être attribuée qu'à un joueur qui joue le match | BDD: CartonsRougesHistory\_EstDansFeuilleDeMatch (Trigger) FORM : Inscriptions des résultats |
| R33 | Pour qu'une carte rouge soit donnée, il faut que les 2 feuilles de matchs soient complétées et valides | BDD : CartonsRougesHistory\_EnoughJoueurs (Trigger) BL : CheckConditionsResultats FORM : Accueil MatchManagement, Inscription des résultats |
| R34 | Toute modification d'un carton rouge doit s'assurer que la carte n'ait pas déjà reçu une modification plus récente | BDD : CartonsRougesHistory\_UpdatePlusVieux (Trigger) |
| R35 | Un joueur ne peut recevoir qu'un carton rouge par match | BDD : MatchManagement.CartonsRougesHistory\_Add (PS) BDD : MatchManagement.CartonsRouges\_Update (PS) BDD : CartonsRougesHistory\_MaxCartonParMatch (Trigger) FORM : Inscriptions des résultats |
| R36 | Les cartons jaunes ne peuvent être reçus qu'entre la minute 0 et 120 d'un match | BDD : CartonsJaunesHistory\_LimiteMinuteRecue (Trigger) FORM : Inscriptions des résultats |
| R37 | Toute modification d'un carton jaune doit s'assurer que la carte n'ait pas déjà reçu une modification plus récente | BDD : CartonsJaunesHistory\_UpdatePlusVieux (Trigger) |
| R38 | Une carte jaune ne peut être attribuée qu'à un joueur qui joue le match | BDD : CartonsJaunesHistory\_EstDansFeuilleDeMatch(Trigger) FORM : Inscriptions des résultats |
| R39 | Pour qu'une carte jaune soit donnée, il faut que les 2 feuilles de matchs soient complétées et valides | BDD : CartonsJaunesHistory\_EnoughJoueurs  (Trigger) BL : CheckConditionsResultats FORM : Accueil MatchManagement, Inscription des résultats |
| R40 | A l'attribution d'un carton jaune, il est actif | BDD : Valeur par défaut isActive 1 |
| R41 | Les goals ne peuvent être inscrits qu'entre la minute 0 et 120 d'un match | BDD : GoalsHistory\_LimiteMinuteRecue (Trigger) FORM : Inscriptions des résultats |
| R42 | Toute modification d'un goal doit s'assurer que le goal n'ait pas déjà reçu une modification plus récente | BDD : GoalsHistory\_UpdatePlusVieux (Trigger) |
| R43 | Un goal ne peut être marqué que par un joueur qui joue le match | BDD: GoalsHistory\_EstDansFeuilleDeMatch  (Trigger) FORM : Inscriptions des résultats |
| R44 | Pour qu'un goal soit inscrit, il faut que les 2 feuilles de matchs soient complétées et valides | BDD : GoalsHistory\_EnoughJoueurs (Trigger) BL : CheckConditionsResultats FORM : Accueil MatchManagement, Inscription des résultats |
| R45 | Pour qu'un goal soit inscrit ou modifié, il faut qu'il n'y ait pas eu de transferts postérieurs (dans un même championnat) | BL : CheckConditionsResultats |
| R46 | Il ne peut y avoir qu’une feuille de match pour une équipe pour un match précis. | BDD : MatchManagement.FeuilleDeMatch\_Add (PS) BDD : MatchManagement.FeuilleDeMatch\_Update (PS) BDD : FeuillesDeMatch\_MaxFeuilleParEquipe (Trigger) BL : FeuillesMatchService |
| R47 | Il ne peut avoir que 2 feuilles de match pour un même match. | BDD : MatchManagement.FeuilleDeMatch\_Add (PS) BDD : MatchManagement.FeuilleDeMatch\_Update (PS) BDD : FeuillesDeMatch\_MaxFeuilleParMatch  (Trigger) |
| R48 | Toute modification d'une feuille de match doit s'assurer que la feuille n'ait pas déjà reçu une modification plus récente | BDD : MatchManagement.FeuilleDeMatch\_Update (PS) BDD : FeuillesDeMatch\_UpdatePlusVieux (Trigger) |
| R49 | Une feuille de match ne peut être remplie que si les antérieure l'on également été. | FORM : Accueil MatchManagement |
| R50 | Une équipe ne peut être composée que de 10 joueurs maximum | BDD : TransfertsHistory\_NombreJoueursMax (Trigger) BDD : TransfertsHistory\_NombreJoueurs (Trigger) BL : TransfertsService |
| R51 | Un joueur ne peut pas changer d’équipe lors d’un quarter. | BDD : TransfertsHistory\_DateTransfert (Trigger) BL : TransfertsService |
| R52 | Aucun transfert ne peut avoir lieu si une des 2 équipes a encore des matchs à jouer à une date ultérieure (dans les limites du championnat) | BL : TransfertsService |
| R53 | Un joueur ne peut changer d’équipe à l’intersaison que si son équipe possède plus de 5 joueurs et que l’équipe dans laquelle il va est classée parmi les 3 dernières du championnat. | BDD : TransfertsHistory\_DateTransfert (Trigger) BL : TransfertsService |
| R54 | Un joueur ne peut pas jouer dans 2 équipes en même temps | BDD : BackEnd.Tansferts\_Add (PS) BDD : TransfertsHistory\_DeuxEquipes (Trigger) BL : TransfertsService |
| R55 | Toute modification d'un transfert doit s'assurer que le transfert n'ait pas déjà reçu une modification plus récente | BDD : TransfertsHistory\_UpdatePlusVieux (Trigger) |
| R56 | Il ne peut avoir que 7 joueurs maximum par feuille de match | BDD : MatchManagement.JoueursParticipation\_Add (PS) BDD : JoueursParticipationHistory\_Max (Trigger) BDD : JoueursParFeuille (Trigger) |
| R57 | Un joueur ne peut être inscrit que s’il joue dans l’équipe de la feuille de match | BDD : MatchManagement.JoueursParticipation\_Add (PS) BDD : JoueursParticipationHistory\_EstDansEquipe (Trigger) FORM : Feuille de match |
| R58 | Un joueur qui a une carte rouge active ne peut pas être inscrit sur la feuille de match | BDD : MatchManagement.JoueursParticipation\_Add (PS) BDD : JoueursParticipationHistory\_CartonRouge (Trigger) FORM : Feuille de match |
| R59 | Un joueur qui a le même nombre ou plus de carte jaune que de match restant à jouer pour son équipe, il ne peut pas être inscrit sur la feuille de match. | BL : GenerationTableauxFeuille |

# Analyse technique

## Technologies proposées

La base de données a été réalisée en SQL Server.

La partie MatchManagement fait appel à ADO.NET

La partie BackEnd fait appel à Entity Framework – DataBase First

Les interfaces utilisateurs sont réalisés via des Windows Forms

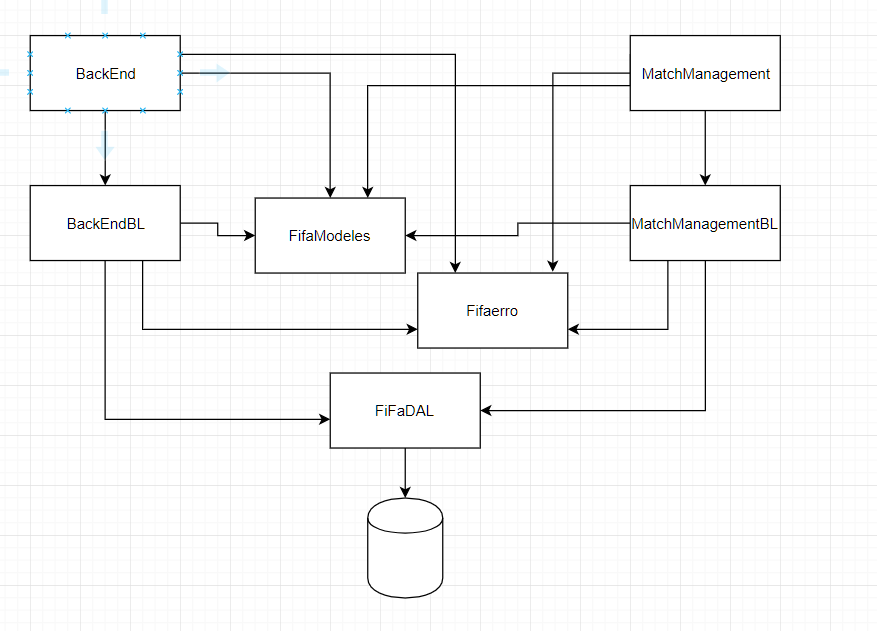
## Architecture applicative

Comme expliqué plus haut, un découpage en couches a été appliqué et ce pour les 2 parties BackEnd et MatchManagement :

* La base de données
* La DAL qui accède à la base de données
* La couche Business qui réalise les traitements sur les données ainsi obtenues
* La couche interface qui affiche et permet à l’utilisateur de réaliser des actions définies.

De plus, une librairie a été réalisée pour stocker les modèles des objets et une autre pour les erreurs.

Pour résumer les différentes interactions, une image vaut 100 mots :



## Listes erreurs

2 types d’erreurs sont prévues pour cette solution.

D’une part, les technicalErrors qui sont soulevées par la base de données au travers des triggers et des procédures stockées.

D’autre part, les BusinessErrors qui sont soulevées dans les couches Business de la solution.

### TechnicalErrors – SQLExceptions

|  |  |  |
| --- | --- | --- |
| **Numéro** | **Message** | **État** |
| 50001 | Les années doivent être supérieures ou égales à 1900 | 1 |
| 50002 | Ce championnat existe déjà | 2 |
| 50003 | Ce championnat n’existe pas | 2 |
| 50004 | La date de début du championnat est trop tardive | 3 |
| 50005 | Il existe déjà une intersaison pour ce championnat | 2 |
| 50006 | Il existe déjà un quarter pour ce championnat | 2 |
| 50007 | Il ne peut avoir que 2 quarters par championnat | 3 |
| 50008 | Cette équipe n’existe pas | 2 |
| 50009 | L'équipe est déjà inscrite à ce championnat | 3 |
| 50010 | La date de match ne tombe pas pendant un quarter existant | 4 |
| 50011 | Les 2 équipes inscrites sont les mêmes | 4 |
| 50012 | Un match similaire a déjà été prévu pour ce championnat | 4 |
| 50013 | Une mise à jour plus récente a été effectuée | 5 |
| 50014 | Un carton rouge a déjà été donné à ce joueur pendant ce match ! | 4 |
| 50015 | Il ne reste plus de suspension à cette carte rouge | 4 |
| 50016 | Ce carton rouge n’existe pas | 2 |
| 50017 | Un carton est obtenu entre 0 et 120 minutes de match | 6 |
| 50018 | Un goal est marqué entre 0 et 120 minutes de match | 6 |
| 50019 | Ce carton jaune n’existe pas | 2 |
| 50020 | Ce goal n’existe pas | 2 |
| 50021 | Une feuille de match existe déjà pour cette équipe et pour ce match ! | 4 |
| 50022 | Il existe déjà 2 feuilles de match pour ce match ! | 4 |
| 50023 | Cette feuille de match n’existe pas | 2 |
| 50024 | Il y a déjà 7 joueurs inscrits sur cette feuille de match | 4 |
| 50025 | Ce joueur n’est pas inscrit dans l’équipe | 4 |
| 50026 | Joueur inscrit avec un carton rouge, ne peut pas être inscrit | 4 |
| 50027 | Joueur inscrit avec autant ou plus de cartons jaunes que de matchs restants | 4 |
| 50028 | Joueur déjà inscrit sur la feuille de match | 2 |
| 50029 | L’équipe doit avoir entre 5 et 10 joueurs | 4 |
| 50030 | L’équipe compte déjà 10 joueurs | 4 |
| 50031 | Un joueur ne peut pas être transférer lors d''un quarter | 4 |
| 50032 | Un joueur ne peut être transféré que dans une des 3 dernières équipes du classement lors de l’intersaison | 4 |
| 50033 | Il ne peut avoir qu’une intersaison par championnat | 3 |
| 50034 | Un nom ne peut être utilisé que par une équipe | 3 |
| 50035 | Une équipe doit respecter le nombre de joueurs min et max pour s’inscrire dans un championnat | 4 |
| 50036 | Un joueur doit quitter son équipe avant d''en rejoindre une nouvelle | 2 |
| 50037 | Une des 2 feuilles d’équipe pour ce match n" pas remplie | 2 |
| 50038 | Les années des 2 dates doivent être identiques | 2 |
| 50039 | L’année de date de début doit être la même année que son championnat | 2 |

### BusinessErrors

Les BusinessErrors sont reprises dans les classes suivantes :

* ChampionnatsService
* JoueursService
* TransfertsService
* ClassementEquipe

# Lessons learned

Autant des erreurs ont été réalisées dans la gestion de ce projet et dans l’approche de ce problème, autant l’identification de ces erreurs et leur résolution m’a permis d’apprendre beaucoup de chose sur le travail d’analyse et de développement mais également sur mon approche d’un problème.

### Ne rien faire à l’avance

Je pense que le premier point que ce travail m’a appris et il s’agit sans doute d’un des plus important est de ne rien faire à l’avance. J’ai travaillé couche par couche, ce qui n’est pas forcément mauvais mais m’a demandé beaucoup de travail de réécriture et d’adaptation sur des procédures stockées et triggers. De plus, certaines de ces procédures se sont révélées par la suite inutiles.

Il aurait été, je pense, plus efficace de faire un socle de base à chaque couche et de développer par la suite selon les besoins. Les mockups (réalisés avant la base de données) de limiter au maximum ce type d’erreurs et sont vraiment important à la réalisation d’un projet de ce type.

### Bien identifier la technologie nécessaire

Au départ, la partie BackEnd a été développée en utilisant Entity Framework – CodeFirst. Je pensais que cette solution me permettrait plus de flexibilité au niveau du code. Malheureusement, je n’avais pas identifié le problème des procédures stockées, CodeFirst limitant le nombre de procédures stockées mappées par entité et ce travail nous interdisant tout accès direct aux tables cela a transformé une flexibilité en quelque chose de très lourd.

Mais cette mauvaise identification m’a permis d’apprendre plus en profondeur le code first et la DB first, leurs différences techniques et d’approches. De plus, en Code First, l’utilisation de LINQ m’a permis d’améliorer grandement mon utilisation des prédicats.

### Découper son code et tester sur des petits morceaux de code

Il s’agit du premier « gros » projet de développement à réaliser pour l’Ephec, plein d’enthousiasme et ayant pour une fois des journées entières à consacrer au codage, je me suis laissé aller à développer de gros morceaux de code et de grosses fonctionnalités pour ne faire mes tests qu’à la fin de la journée et avec peu de commentaires.

Il est inutile de dire que très rapidement, je me suis rendu compte de l’importance de diviser au maximum les fonctions en méthodes, de commenter mon code et de rendre lisible et facilement compréhensibles mes variables et méthodes.

### DATAGRIDVIEW – DATATABLE - Dataview

L’utilisation des forms pour ce projet m’a très vite imposé l’utilisation de nombreux DataGridView pour rendre l’interface le plus amical possible avec l’utilisateur. Matière survolée aux cours.

Ces utilisations m’ont demandé beaucoup de recherches sur la nature même des DataGridView et leur approche. A travers ses recherches j’en ai également beaucoup appris sur les DataTable et DataView.

### Le mieux est l’ennemi du bien

Encore maintenant alors que tout fonctionne bien, en regardant mon code le plus ancien, j’ai l’envie de le refactoriser, de le simplifier, de l’améliorer.

Je me rends compte qu’à travers ce projet mon code s’est amélioré en de nombreux aspects. Malgré tout, il me faut laisser certaines parties de code « anciennes ». Ces parties ont fait leurs preuves à travers de nombreux tests et fonctionnent pour ce qu’on leur demande.

Certaines modifications réalisées uniquement pour améliorer le code m’ont amené à bien plus de changements que je ne le pensais et ont été énergivores. Ce n’est pas forcément une mauvaise chose car cela m’a appris d’autres choses mais dans un projet il faut également savoir gérer son temps.

### Ne pas faire le malin

Lors de la création de la base de données, j’ai voulu complexifier à dessin l’utilisation des tables pour montrer ce que je pouvais faire.

Je suis évidemment vite rendu compte que c’était très énergivore et complètement inutile. Il faut rester le plus simple à toute occasion, voici ce que j’en retire également.

# Conclusion

Un tel projet n’est jamais évident. Il y a eu des moments de doutes, de questionnements, des retours en arrières dans une approche, un client n’est jamais précis dans sa demande, son contexte n’est jamais bien défini.

J’espère que la réflexion au travers des différents chapitre a pu mettre en lumière de manière suffisamment simple et compréhensible le résultat de ces doutes.

Les différentes étapes : schéma entité-association – database – mockups – code m’ont permis à chaque fois d’affiner cette pensée et de préciser ma solution.

Je pense offrir ici une solution qui répond à toutes les demandes critiques et propose même certaines bases permettant une modification future plus facile.

Ce projet m’a énormément appris et m’a ouvert l’appétit pour faire de plus grands projets encore, j’ai hâte de travailler en équipe sur un grand projet.

Je finirai en remerciant Monsieur Fievez pour m’avoir donné cette opportunité de développer ma pensée et mes connaissances.