

# Projet Développement Mobile 2023



Réalisé par Popadiuc Claudiu

Supervisé par Monsieur Riggio, Jonathan

## Table des matières

<b>Introduction .....</b>	<b>3</b>
<b>Description des technologie utilisées .....</b>	<b>4</b>
<b>Sujet d'application.....</b>	<b>5</b>
<b>Fonctionnalités .....</b>	<b>6</b>
<b>Analyse.....</b>	<b>35</b>
<b>Limitations et développement futur .....</b>	<b>37</b>
<b>Play Store .....</b>	<b>39</b>
<b>Conclusion .....</b>	<b>40</b>

# Introduction

Dans le cadre du cours de Mobile, il nous a été demandé de réaliser une application mobile qui sera développée uniquement sur Android Studio en utilisant le langage Java. Pour ma part, j'ai réalisé une application de gestion des matchs de football destiné aux arbitres, comprenant un historique des matchs ainsi que les profils des joueurs.

Dans ce rapport, je vais commencer par une description des technologies utilisées ainsi que mes motivations. Ensuite, je rentrerai dans le vif du sujet en expliquant le sujet de mon application suivi par les fonctionnalités de mon projet, ainsi que tout ce qui a été implémenté d'un point de vue à la fois fonctionnel et technique, en mettant en avant les défis techniques et les solutions apportées. Les points d'intérêt concernant la structure de mon implémentation seront ensuite présentés sous forme de diagramme.

Pour donner suite à cela, il y aura une section UI/UX qui détaillera les points d'attention apportés à l'UI/UX, ainsi que la façon dont les besoins et les préférences des utilisateurs ont été pris en compte.

Je continuerai mon rapport avec les limites de mon application et je répondrai à des questions telles que dans quels cas d'utilisation mon application pourrait ne pas fonctionner comme prévu ? Y a-t-il des aspects techniques non traités ? Si j'avais plus de temps pour le projet, qu'aurais-je amélioré ?

Je finirai ensuite mon rapport avec une conclusion en résumant ce qui a été vu dans le rapport, ce que j'ai réussi à faire ou non durant le projet, et les apprentissages que j'en tire. Pour finir, je donnerai mon avis et mon ressenti sur le développement de mon application.

## Description des technologies utilisées

Pour réaliser ce projet, j'ai utilisé exclusivement Android Studio comme environnement de travail. Android Studio est un environnement de développement intégré (IDE) utilisé pour créer des applications mobiles pour la plate-forme Android. Les langages de programmation possibles sont Java et Kotlin. Android Studio est largement utilisé par les développeurs du monde entier pour créer des applications mobiles pour la plate-forme Android.

Pour mon projet, j'ai utilisé uniquement le langage Java. Java est un langage de programmation populaire utilisé pour créer une grande variété d'applications, notamment des applications de bureau, mobiles, et web. Il est orienté objet, portable et largement utilisé dans l'industrie pour sa fiabilité et sa sécurité.

Dans Android Studio, le code Java est associé au code XML qui est disponible lors de la création d'une activité. XML est un langage de balisage utilisé pour décrire la structure et le contenu des données de manière structurée et lisible par les machines. Il est largement utilisé dans le développement d'applications web et d'autres domaines pour échanger et stocker des données.

## Sujet d'application

De manière simple et structurée, voici les fonctionnalités de mon application :

Le but de mon application est de permettre à l'utilisateur d'avoir sa propre saison de football avec les joueurs qui y participent. L'application permettra de créer des matchs et d'enregistrer l'historique de tous les matchs précédents. Chaque joueur pourra avoir son propre profil avec des statistiques telles que le nombre de défaites ou de victoires.

Une notification sera envoyée pour le chronomètre qui décompte le temps restant lorsqu'un match est joué, de sorte que l'arbitre n'aura pas besoin de se connecter à l'application pour consulter le temps restant. Une autre notification sera envoyée à l'arbitre une fois que le match sera terminé.

L'application sera également compatible avec les tablettes. Il y aura également une API qui permettra d'indiquer la position des joueurs comme pour les joueurs professionnels (ceci est expliqué plus en détail dans la section sur l'API).

Une carte sera disponible lors de la création de chaque match par l'arbitre, qui pourra ajouter le lieu où le match est joué. Dans l'historique, l'emplacement exact où le match de football a été joué sera disponible.

Voici le fonctionnement de l'application :

Un utilisateur pourra s'inscrire ou non dans l'application, Un utilisateur sera soit « Arbitre » soit « Joueur » S'il est arbitre, il pourra créer des profils pour les joueurs qui auront accès à l'application. L'arbitre pourra également créer des matchs qui seront enregistrés dans l'historique. La différence entre l'utilisateur « arbitre » et l'utilisateur « joueur » c'est que les joueurs ne pourront pas créer de matchs. Ils pourront seulement consulter l'historique des matchs ainsi que les profils des joueurs de la saison sans pouvoir modifier leurs statistiques. Seul l'arbitre pourra le faire.

# Fonctionnalités

## Format tablette, téléphone

Lorsque j'ai conçu mon application, j'ai pris en compte la diversité des appareils sur lesquels elle pourrait être utilisée. Cela inclut à la fois les téléphones et les tablettes, qui ont des tailles d'écran et des résolutions différentes. Mon objectif était de créer une expérience utilisateur cohérente et optimale, quel que soit le dispositif utilisé.

Pour garantir une compatibilité étendue, j'ai utilisé différentes techniques et fonctionnalités. Par exemple, lorsque l'utilisateur passe de l'orientation portrait à l'orientation paysage ou vice versa, le fond d'écran s'adapte automatiquement pour correspondre à la nouvelle orientation. Cela permet une expérience visuelle fluide et agréable, sans aucune distorsion ou perte de qualité.

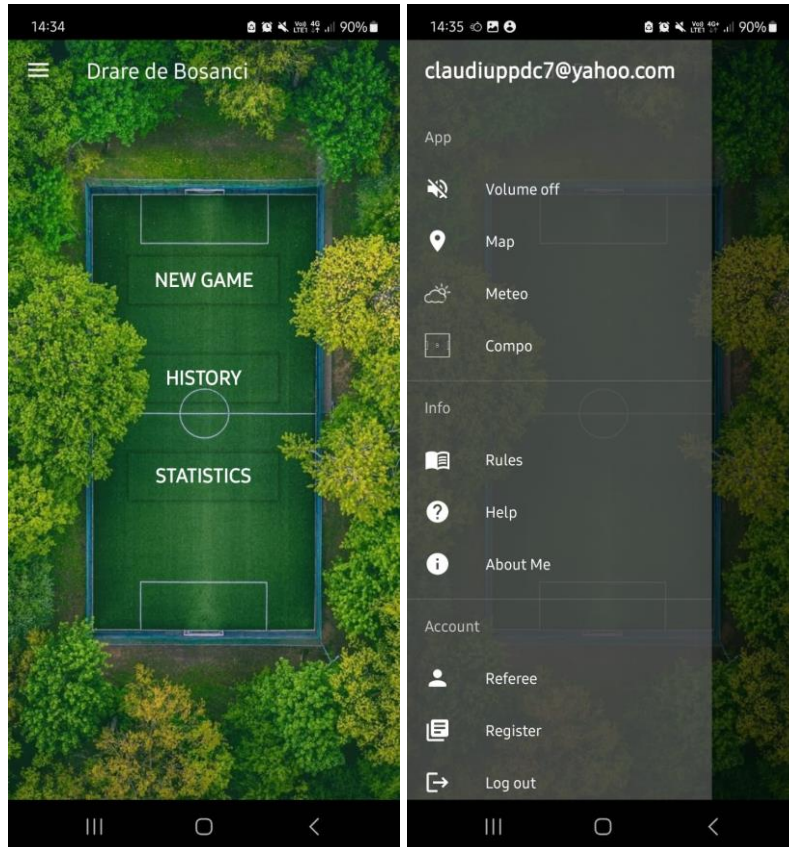
Une autre considération importante était de m'assurer que l'application s'adapte aux différentes tailles d'écran des appareils. Pour cela, j'ai utilisé des éléments de mise en page flexibles, tels que les `LinearLayout`, en utilisant judicieusement les attributs `"match_parent"` et `"wrap_content"`. Cela permet à l'interface utilisateur de se redimensionner dynamiquement en fonction de l'espace disponible sur l'écran. Ainsi, l'application peut tirer pleinement parti de l'espace d'affichage, que ce soit sur un petit téléphone ou sur un grand écran de tablette.

De plus, j'ai intégré des `ScrollView` dans plusieurs parties de l'application. Cela permet aux utilisateurs de faire défiler le contenu lorsque l'espace vertical est limité, comme lorsqu'un clavier virtuel est affiché. Ainsi, même dans ces situations, les utilisateurs peuvent accéder à tous les éléments de l'interface et interagir avec eux sans contraintes.

En créant une conception responsive et en utilisant ces techniques d'adaptabilité, j'ai pu offrir une expérience utilisateur cohérente et optimisée, quels que soient les différents appareils sur lesquels mon application est utilisée. Les utilisateurs peuvent profiter pleinement de toutes les fonctionnalités de l'application, peu importe la taille de leur écran, garantissant ainsi une expérience utilisateur de haute qualité.

## Fonctionnalités propres à mon application

### Page d'accueil



Voici ma page d'accueil, la première page qui s'affiche lors du démarrage de l'application. On y trouve trois boutons principaux : "New Game", "History" et "Statistics", ainsi qu'un menu situé en haut à gauche. Dans cette section, je vais vous présenter progressivement toutes les fonctionnalités de mon projet, en commençant par les boutons principaux et en terminant par le menu.

Le bouton " New Game " permet de démarrer une nouvelle partie de jeu. En appuyant sur ce bouton, l'utilisateur est dirigé vers l'interface de création d'une nouvelle partie, où il peut saisir les informations nécessaires telles que les noms des équipes, la durée du match, etc.

Le bouton " History " donne accès à l'historique des parties précédentes. Lorsque l'utilisateur le sélectionne, il est redirigé vers une page qui affiche une liste des parties enregistrées.

Le bouton " Statistics " permet d'accéder aux statistiques des joueurs. En appuyant dessus, l'utilisateur est dirigé vers une page qui affiche les statistiques individuelles de chaque joueur enregistré. Il peut sélectionner un joueur spécifique pour afficher ses performances détaillées et ses données statistiques.

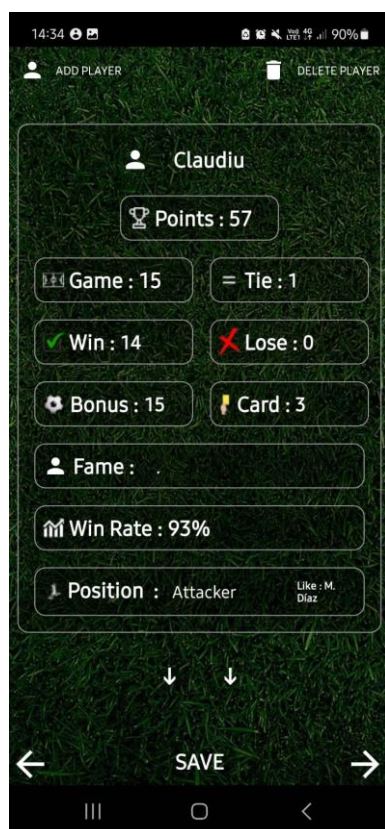


Le menu en haut à gauche regroupe différentes options supplémentaires. Par exemple, il peut contenir des fonctionnalités telles que la gestion des joueurs, la personnalisation des paramètres du jeu, l'accès à l'aide ou aux paramètres de l'application. Ce menu offre une navigation pratique pour explorer les différentes sections et fonctionnalités de l'application.

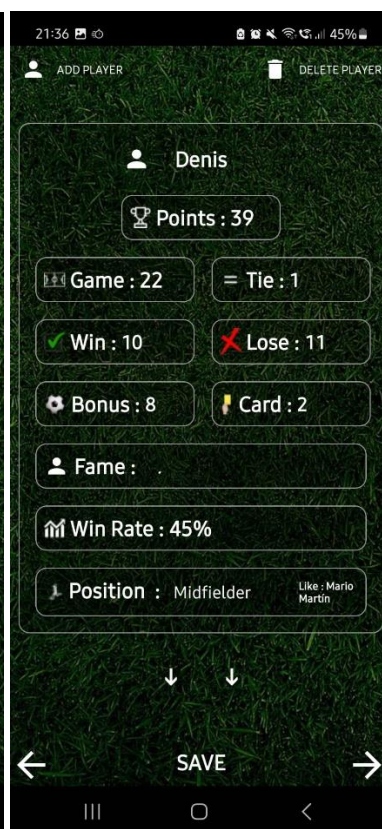
En résumé, ma page d'accueil présente les trois boutons principaux : "Nouvelle partie", "Historique" et "Statistiques", ainsi qu'un menu en haut à gauche pour accéder à d'autres fonctionnalités. Cela permet à l'utilisateur d'interagir facilement avec l'application et d'explorer toutes ses fonctionnalités.

## Statistiques des joueurs

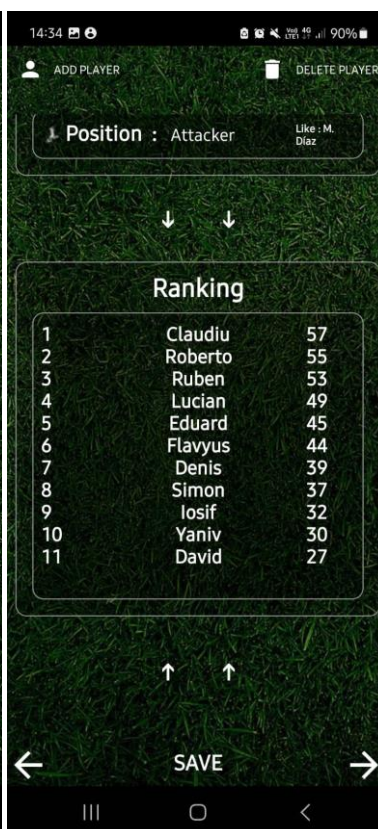
Exemple statistique :



Exemple statistique :



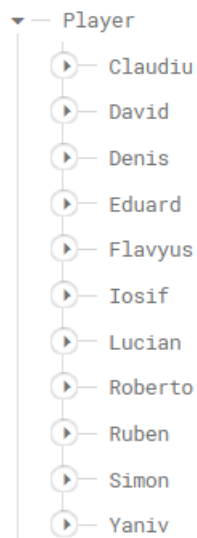
Classement :



Position : Attacker (Like: M. Diaz)		
1	Claudio	57
2	Roberto	55
3	Ruben	53
4	Lucian	49
5	Eduard	45
6	Flavyus	44
7	Denis	39
8	Simon	37
9	Iosif	32
10	Yaniv	30
11	David	27

Pour bien comprendre le système des statistiques, il est important de savoir qu'un joueur peut être ajouté en appuyant simplement sur le bouton situé en haut à gauche de l'application. Lorsque ce bouton est activé, un builder apparaît, demandant de saisir le nom de la personne. Une fois que l'utilisateur appuie sur "OK", le nom du joueur est ajouté à la base de données. Voici la liste de mes joueurs actuels :





Pour afficher les statistiques d'un joueur spécifique, il suffit de sélectionner son nom dans la liste des joueurs. Cette liste est présentée sous forme d'un spinner, où il est possible d'appuyer sur le nom du joueur souhaité. Une fois le joueur sélectionné, les statistiques correspondantes s'afficheront à l'écran, offrant ainsi un aperçu complet de ses performances.

Une autre façon d'accéder aux statistiques est de faire défiler les écrans. L'application propose une animation fluide qui permet de passer d'un joueur à l'autre en faisant glisser l'écran horizontalement. De cette manière, il est possible d'explorer les statistiques de chaque joueur de manière intuitive.

De plus, des flèches gauche et droite sont également présentes pour faciliter la navigation entre les joueurs. En cliquant sur ces flèches, l'utilisateur peut passer d'un joueur à l'autre et voir instantanément les statistiques mises à jour à l'écran.

Lorsque les statistiques d'un joueur sont affichées, elles fournissent des informations détaillées sur ses performances. Ces statistiques permettent d'évaluer la contribution individuelle du joueur à l'équipe et d'identifier ses points forts et ses domaines à améliorer.

Grâce à ces différentes méthodes d'affichage des statistiques, il est facile pour les utilisateurs de suivre et d'analyser les performances de chaque joueur de manière claire et conviviale. Cela facilite l'évaluation de l'équipe dans son ensemble et permet de prendre des décisions éclairées pour optimiser les stratégies et les formations. Voici un exemple dans la base de données :

```
└─ Claudiu
   ├── bonusText: "15"
   ├── fameText: "."
   ├── loseText: "0"
   ├── name: "Claudiu"
   ├── point: "57"
   ├── position: "Attacker"
   ├── tieText: "1"
   ├── winText: "14"
   └── yellowText: "3"
```

Une fois lancée dans l'application, l'interface affichera les statistiques des joueurs de la même manière que dans les captures d'écran initiales. Cette présentation claire et intuitive permet aux utilisateurs de consulter facilement les performances individuelles de chaque joueur.

En plus de la consultation, l'application offre également la possibilité de modifier ou de supprimer un joueur existant. Pour cela, il suffit de sélectionner le nom du joueur dans la liste et d'accéder aux options de modification ou de suppression. Par exemple, en cliquant sur le bouton situé en haut à droite, il est possible de supprimer un joueur de la base de données. Ces fonctionnalités offrent à l'utilisateur la possibilité de maintenir une base de données de joueurs à jour et de la personnaliser en fonction de ses besoins spécifiques.

Il est important de noter que toutes les statistiques ne peuvent pas être modifiées par l'utilisateur. Certains champs, tels que "bonus", "lose", "fame", "win", "position", "tie" et "yellowCard", sont modifiables. En revanche, les champs tels que "winRate", "points" et "Game" sont automatiquement mis à jour en fonction des autres champs. Par exemple, le champ "Game" représente le nombre total de victoires, de défaites et de matchs nuls, tandis que le champ "winRate" calcule la probabilité de victoire en fonction du nombre de victoires et de matchs joués. Quant au champ "points", il attribue 1 point pour chaque match nul, 3 points pour chaque victoire, 1 point pour chaque bonus et -1 point pour chaque troisième carton jaune reçu.

Ce système de statistiques facilite le suivi et l'analyse des performances individuelles de chaque joueur, ce qui s'avère extrêmement utile pour évaluer et améliorer les performances de l'équipe dans son ensemble.

Enfin, en faisant défiler vers le bas, les utilisateurs pourront accéder à un système de classements regroupant tous les joueurs inscrits dans la base de données. Ce classement permet de visualiser et de comparer les performances des joueurs, offrant ainsi un aperçu complet de leur contribution à l'équipe.

Pour mieux comprendre le fonctionnement des statistiques, je vais expliquer de manière technique. J'ai déjà expliqué comment les trois champs automatiques sont gérés, en effectuant simplement des calculs et en utilisant des écouteurs sur les TextView, les champs sont mis à jour en temps réel. Pour ce qui est de l'ajout d'un joueur, le bouton déclenchera un constructeur qui demandera le nom du joueur, et ce nom sera instantanément ajouté à la base de données en tant que nœud avec toutes les

statistiques vides. Pour supprimer un joueur, il suffit de prendre son nom et de le supprimer de la base de données en temps réel.

Afin de sauvegarder toutes ces modifications chaque fois que nous quittons la page ou l'application, il y a un bouton "Sauvegarder" qui, comme son nom l'indique, enregistre les données. Il récupérera tous les champs qui ont été modifiés dans les statistiques grâce au dataSnapshot et les mettra à jour dans la base de données avec les nouvelles données. Ce processus n'est pas plus compliqué que cela, puisque j'avais déjà réalisé l'historique, le principe est le même, et cela a été facile pour moi de l'implémenter.

En ce qui concerne le classement des joueurs sous les statistiques, ce n'était pas non plus très compliqué. J'ai ajouté les points de chaque joueur à une liste, puis j'ai trié cette liste avant de simplement l'afficher dans le TextView. Le cadre "Classement" n'est en réalité qu'un seul TextView, qui affiche donc le classement de tous les joueurs avec leurs points respectifs, étant donné que la liste est triée. J'ai simplement ajouté un compteur qui correspond à la taille de la liste, pour afficher les positions de 1 à 12, et cela se met à jour automatiquement si un joueur est ajouté, passant ainsi de 1 à 13.

Pour chaque point récupéré, j'ai également récupéré le nom de la personne correspondante, ce qui me permet d'afficher le classement, le nom et les points un par un dans mon TextView. C'est ainsi que nous obtenons le résultat que vous pouvez voir dans la capture d'écran.

## API

Dans les statistiques, vous pouvez remarquer un champ appelé "Position" qui est un spinner. Ce spinner comprend 4 éléments : "Gardien de but, Défenseur, Milieu de terrain, Attaquant". En fonction de la position enregistrée, un TextView affichera le nom d'un joueur de l'équipe de football correspondant à cette position. Par exemple, si je sélectionne "Attaquant", le TextView affichera : "K. Benzema".

Dans les captures d'écran des statistiques, vous pouvez voir deux exemples de ce fonctionnement.



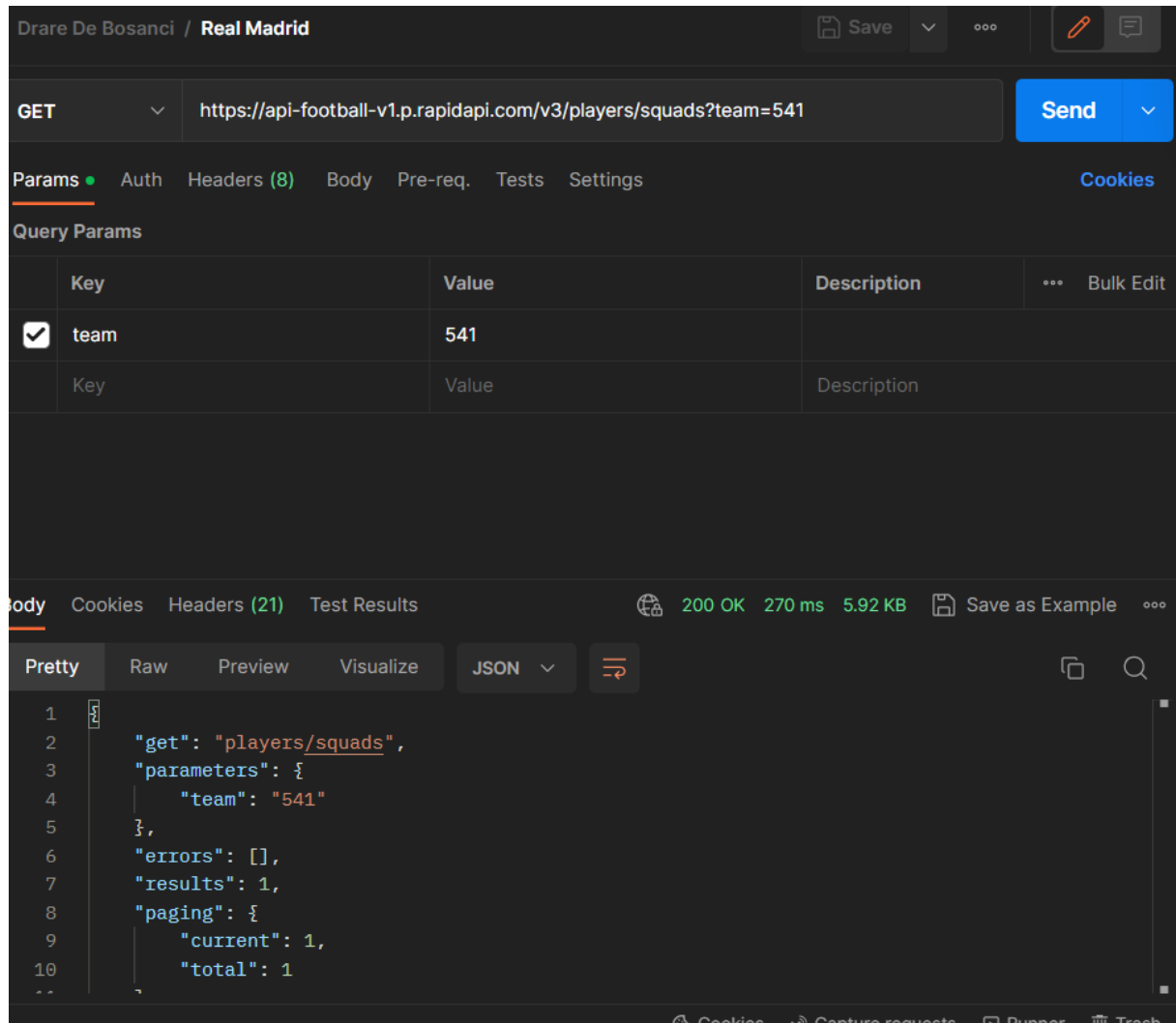
Pour réaliser cette fonctionnalité, j'ai rencontré quelques difficultés. Au début, j'ai suivi un tutoriel en ligne qui expliquait comment récupérer le contenu d'une adresse web au format JSON en utilisant un code de base. J'ai réutilisé ce code en y intégrant un fichier JSON contenant les statistiques nécessaires pour concevoir mon application. Cependant, il n'y avait pas de requêtes API disponibles pour accéder à ces données, ce qui m'a obligé à trouver une véritable API avec des requêtes fonctionnelles. J'ai découvert l'API suivante :

<https://rapidapi.com/api-sports/api/api-football/>.

Cependant, mon plus grand problème était que je n'arrivais tout simplement pas à effectuer des requêtes pour appeler cette API dans mon application. J'ai passé des heures à résoudre ce problème. Pour commencer, j'ai testé l'API dans Postman, Postman est un logiciel qui permet d'envoyer des requêtes HTTP et de les tester. Il offre une interface conviviale où vous pouvez spécifier les paramètres de la requête, tels que l'URL, les en-têtes et les paramètres, et visualiser les réponses renvoyées par le

serveur. Cela facilite le processus de développement et de débogage des API, car vous pouvez vérifier leur fonctionnement et leur intégration avant de les utiliser dans votre propre application.

Voici un aperçu de mon expérience avec Postman :



Finalement, j'ai réussi à intégrer les requêtes API dans mon application et à récupérer les données nécessaires pour afficher les statistiques des joueurs. Cela a été une étape cruciale pour rendre mon application fonctionnelle et fournir des informations actualisées et précises.

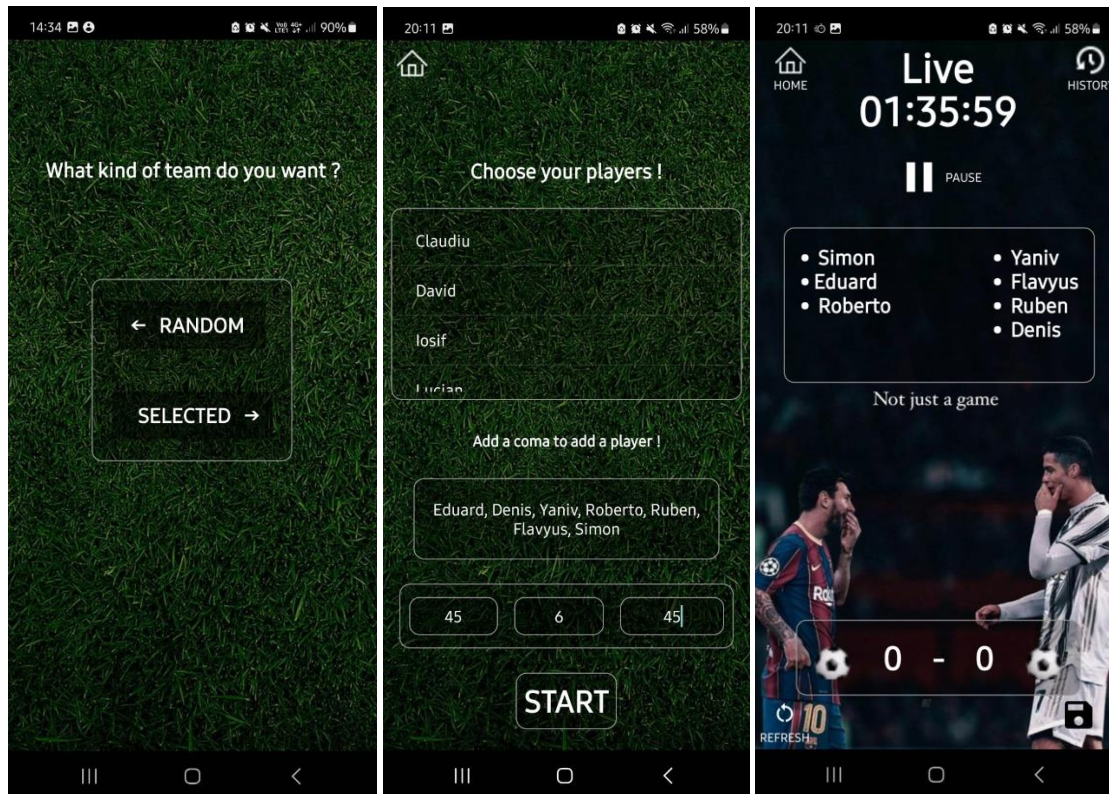
Grâce à cette API, j'ai pu obtenir des données en temps réel et offrir aux utilisateurs une expérience complète et enrichissante en termes de statistiques de football. Cela a été une victoire pour moi, malgré les défis rencontrés lors de l'intégration de l'API dans mon application.

## Création de match

TeamSelection :

Team :

Live :



Voici le fonctionnement de la création d'un match :

Après avoir appuyé sur le bouton " New Game" sur la page d'accueil, nous arrivons sur la première page qui demande simplement le type d'équipe souhaité. Si l'option choisie est "RANDOM" (random), nous serons redirigés vers la page "Team". Sur cette page, il suffit de sélectionner les joueurs disponibles dans mes statistiques. En cliquant sur l'un des joueurs, il sera ajouté à la section "Players" et sera supprimé de la liste des joueurs disponibles, indiquant ainsi à l'utilisateur qu'il a été sélectionné et évitant les doublons. Si l'utilisateur souhaite ajouter un joueur qui ne figure pas dans les statistiques pour ce match, il pourra l'ajouter manuellement en appuyant sur la section "Players" et en écrivant le nom du joueur, suivi d'une virgule, comme indiqué juste au-dessus de la section.

Une fois cette étape terminée, l'utilisateur devra ajouter une durée de jeu pour chaque mi-temps et pour la pause, qui ne pourra pas dépasser 2 chiffres. Ainsi, une mi-temps ne peut pas durer plus de 99 minutes, et la même restriction s'applique à la pause, afin d'éviter des durées aberrantes telles que 45125 minutes pour une pause. Si on n'ajoute pas au minimum 2 joueurs, et qu'on n'indique pas le temps de jeux pour chaque champ, alors le match ne peut pas commencer ce qui est logique, il n'y a pas de match sans temps de jeux et pas d'équipes sans au moins 1 joueur dans chaque équipes.

Une fois toutes ces étapes remplies, nous arrivons à la dernière partie, qui est la diffusion en direct du match. C'est ici que l'arbitre pourra ajouter le nombre de buts en cliquant sur les ballons de football pour chaque équipe. Si, par exemple, les équipes tirées au sort ne conviennent pas, il y a le bouton "Actualiser" en bas à gauche, qui permet de relancer le tirage au sort et de former deux nouvelles équipes avec les joueurs sélectionnés pour le match, remettant ainsi les compteurs de buts à zéro. Si tout est prêt, le match peut commencer en cliquant sur le bouton "Démarrer", qui sera alors mis en mode "Pause" car le match est en cours et le temps s'écoule. On peut mettre le chronomètre en pause si on souhaite interrompre le match.

Le match se déroule tranquillement. Une fois le match terminé, il peut être enregistré en cliquant sur le bouton en bas à droite, ce qui affichera une fenêtre contextuelle demandant à l'utilisateur s'il souhaite enregistrer le match maintenant. Si l'utilisateur le souhaite, il peut enregistrer un match avant la fin du temps de jeu.

En haut à droite du Live, il y a une icône qui permet d'accéder directement à l'historique des matches joués. Si l'utilisateur accède à cet historique, le match en cours sera supprimé s'il n'est pas enregistré. Un dialogue s'affiche alors pour demander si l'utilisateur souhaite quitter le Live sans enregistrer. En revanche, si le match est enregistré, le dialogue n'apparaît pas.

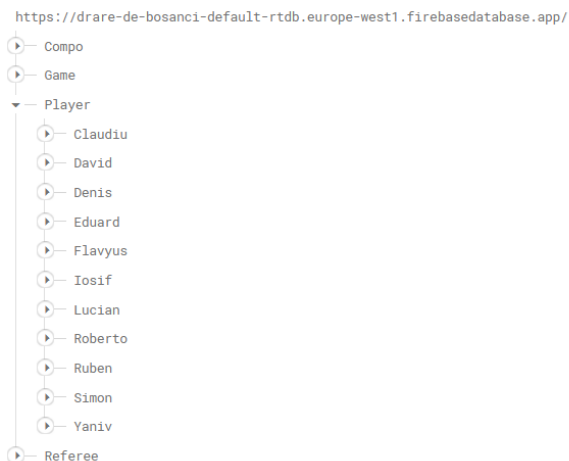
Il convient de noter que pour le système d'équipes "Selected", le principe est similaire à celui du mode "Random", à la différence qu'il y a deux champs de sélection, un pour chaque équipe. Les joueurs peuvent être ajoutés de la même manière que dans le mode "Random".

Pour le Live du match en mode "Selected", le bouton en bas à gauche ne permettra pas de redistribuer aléatoirement les équipes comme dans le mode "Random", car les équipes seront déjà choisies par l'arbitre lui-même. Il n'est donc pas nécessaire de le faire. En revanche, si l'on appuie sur ce bouton, les compteurs de buts seront remis à zéro.

Même si cela n'est pas visible directement, il y a des animations entre les activités. Par exemple, on peut appuyer sur "Random" ou faire glisser vers la gauche comme indiqué par la flèche, de même pour "Selected".

D'un point de vue technique, tout cela n'est pas très compliqué à comprendre. Pour la partie "Random", le champ d'équipe est un EditText, et pour la sélection des joueurs, c'est une

ListView avec tous les joueurs provenant de la base de données en temps réel.



Comme vous pouvez le voir, ce sont tous les joueurs de mes statistiques qui sont affichés. Ils seront ajoutés à ma liste, et chaque élément peut être cliqué pour être ajouté dans l'EditText. Ma liste a un écouteur qui permet de détecter toute modification apportée à la liste et de la mettre à jour. Ainsi, lorsque je clique sur un élément, son nom est supprimé et grâce à l'écouteur, cela fonctionne sans problème. Ensuite, pour chaque élément sélectionné, le texte de l'élément est simplement ajouté à l'EditText, suivi d'une virgule. Pourquoi une virgule ? Parce que plus tard, je vais prendre le contenu de l'EditText et le diviser en une liste, et la virgule servira de séparateur entre les nombres et les noms des joueurs. Cela sera fait lorsque l'on appuie sur le bouton "Start".

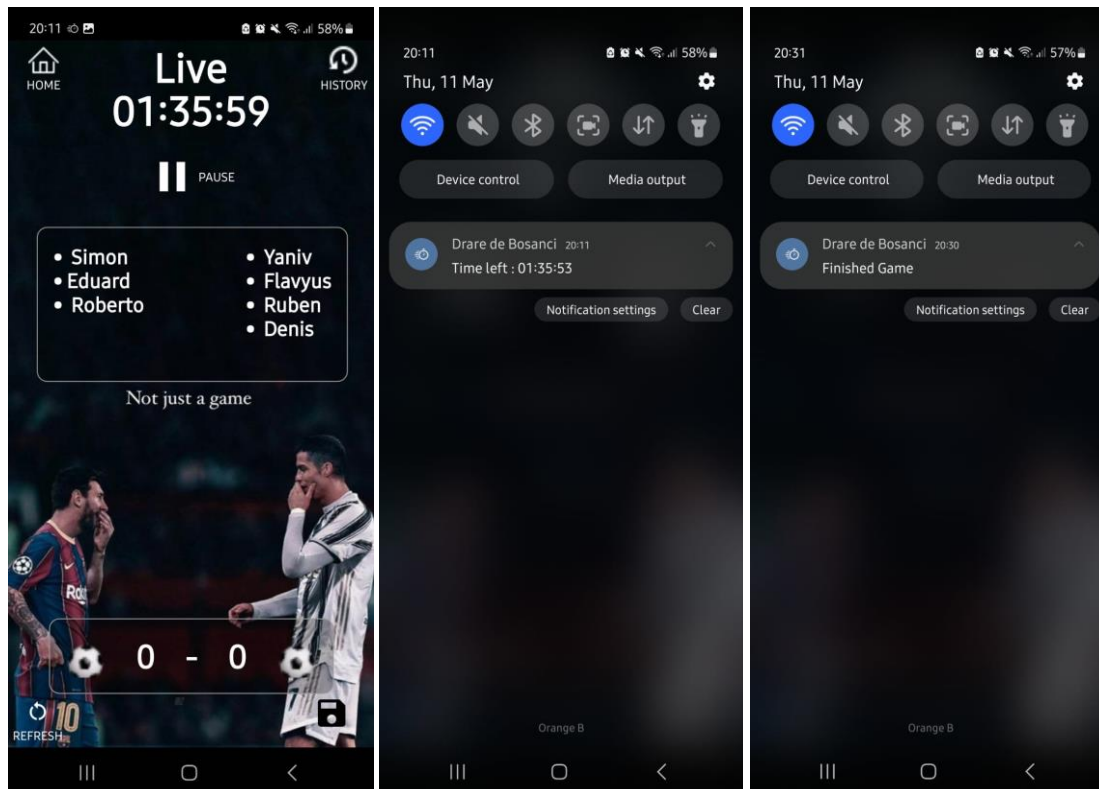
Tous les joueurs seront placés dans une liste, qui sera ensuite mélangée à l'aide de `Collections.shuffle()`. Ensuite, la liste sera simplement divisée en deux listes, représentant les joueurs répartis aléatoirement dans les équipes. Le même principe est appliqué pour les équipes prédéfinies, mais il n'y aura pas de tirage au sort, les joueurs seront directement affectés à l'équipe 1 ou à l'équipe 2 en fonction de leur liste respective.

Une fois que le match commence, chaque liste sera simplement affichée dans un `TextView` situé sous le temps, là où se trouvent les noms, dans la page Live, dans le grand cadre. Ainsi, chaque joueur sera affiché dans son équipe. Comme vous pouvez le voir dans la capture d'écran, même s'il y a un nombre impair de joueurs, cela créera tout de même des équipes avec 3 et 4 joueurs, au lieu de former des équipes avec 2 et 6 joueurs. Cela est pratique pour garantir l'équité des équipes lorsqu'il s'agit d'une répartition aléatoire.

Pour les champs de temps, j'ai ajouté une contrainte qui limite la saisie à seulement 2 chiffres, grâce à un écouteur de texte qui vérifie à chaque fois qu'un chiffre est ajouté dans le champ que la longueur de la chaîne soit inférieure à 2. J'ai également utilisé un attribut `inputType` de type "number" pour ces champs, ce qui signifie que les lettres ne seront pas autorisées dans ces champs.



## Notifications



Mon système de notification est très simple. Une notification est envoyée lorsque l'arbitre crée un match et que l'on arrive sur la page du Live. Un chronomètre est alors affiché dans la notification, indiquant le décompte du temps restant. Lorsque le match est terminé, l'arbitre reçoit une notification lui indiquant que le match est fini.

Dans les captures d'écran ci-dessus, il est 20h31, car j'ai simplement enlevé le chronomètre de 1h35 pour ne pas attendre réellement 1h35 afin de prendre la capture d'écran montrant la notification de fin de match. Je tiens à préciser cela pour que vous sachiez qu'il n'y a aucune manipulation.

De plus, j'ai ajouté la fonctionnalité de mise en pause du chronomètre, aussi bien dans la notification que dans l'affichage de l'application.

Cependant, je dois admettre que cette fonctionnalité m'a pris beaucoup de temps, malgré sa simplicité apparente. J'ai rencontré plusieurs problèmes lors de sa mise en place. Tout d'abord, il fallait afficher le chronomètre dans la notification, en récupérant la valeur qui était affichée dans le grand TextView de l'application. Une fois cela réussi, d'autres problèmes sont survenus.

Pour résoudre ces problèmes, j'ai dû me pencher sur les mécanismes de gestion des notifications et trouver des solutions adaptées. J'ai également dû gérer la mise en pause du chronomètre à la fois dans la notification et dans l'application. Cela nécessitait une synchronisation précise entre les deux éléments.

Malgré ces difficultés, j'ai finalement réussi à mettre en place un système de notification fonctionnel et cohérent. Les notifications permettent à l'arbitre de rester informé de l'avancement du match, de connaître le temps restant et de savoir quand le match est terminé.

Malgré ces difficultés, j'ai réussi à développer un système de notification robuste, fiable et fonctionnel. Les notifications permettent à l'arbitre de suivre facilement l'évolution du match, de connaître le temps restant et d'être informé dès que le match se termine.

La mise en place de cette fonctionnalité a été extrêmement complexe et chronophage. En effet, pour obtenir une notification qui se met à jour chaque seconde, il aurait fallu envoyer une notification à chaque seconde. Cependant, cela aurait entraîné un flux incessant de notifications, ce qui aurait été très perturbant pour l'utilisateur.

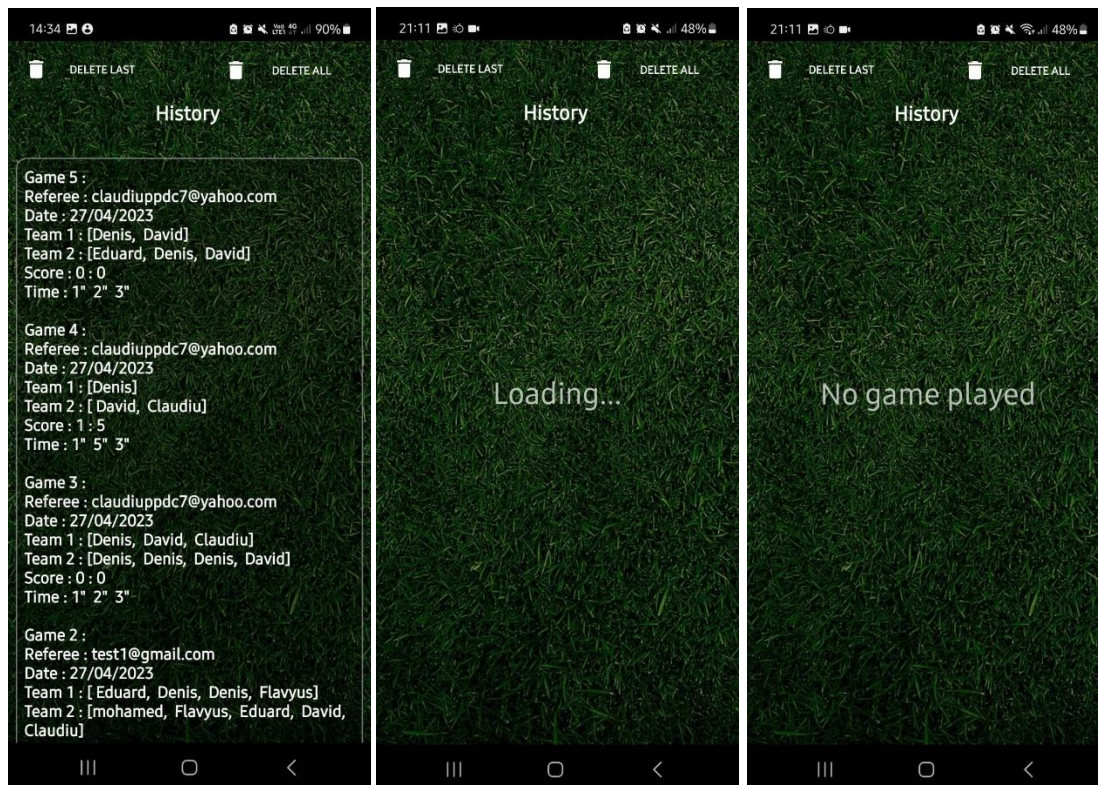
Finalement, j'ai trouvé une solution en utilisant une méthode appelée "onTick". Cette méthode était appelée à chaque décrémentation du chronomètre, ce qui permettait de mettre à jour l'affichage de la notification sans pour autant envoyer une nouvelle notification à chaque seconde. Ainsi, la méthode se contentait de mettre à jour le texte affiché dans la notification en fonction du temps restant. Cependant, il était important de ne pas récupérer la valeur du chronomètre à chaque seconde à partir du TextView, car cela aurait provoqué l'envoi répété de notifications. J'ai donc modifié cette partie du code pour que la notification elle-même puisse récupérer la valeur du chronomètre et gérer la décrémentation du temps, de manière similaire à l'affichage dans le TextView du Live.

J'ai également rencontré quelques difficultés lors de la mise en pause du chronomètre. Lorsque j'ai essayé d'implémenter cette fonctionnalité pour permettre la pause du match, j'ai dû effectuer plusieurs calculs pour résoudre certains problèmes. Par exemple, lorsque j'appuyais sur pause, le chronomètre se terminait et affichait la dernière valeur du chronomètre. Si je le relançais, il redémarrait complètement. De plus, j'ai également rencontré des problèmes avec la gestion de plusieurs chronomètres en même temps, notamment dans la section de la notification de la pause, car les mêmes problèmes d'affichage se présentaient.

Finalement, j'ai compris qu'il était nécessaire de stocker la dernière valeur du chronomètre et de reprendre simplement le chronomètre à partir de cette valeur lorsque je le relançais, afin d'éviter les incohérences dans l'affichage et le fonctionnement du chronomètre.

Malgré ces défis, j'ai réussi à mettre en place un système de notification fonctionnel et fiable. Les notifications se mettent à jour régulièrement sans inonder l'utilisateur de notifications répétitives. De plus, la fonctionnalité de pause du chronomètre a été implémentée de manière efficace, permettant à l'arbitre de gérer les pauses du match de manière fluide.

## Historique de match



Dans mon application, j'ai mis en place un système d'historique des matches créés. Cette fonctionnalité permet d'avoir un historique de tous les matches enregistrés lors de la création. La page d'historique affiche, pour chaque match enregistré, le numéro de la partie, l'adresse e-mail de la personne ayant créé le match, la date du match, les joueurs des deux équipes, le score final du match, ainsi que la durée de chaque mi-temps et pause. Les matches sont affichés dans une ScrollView, du plus récent au plus ancien.

Sur cette page, on trouve également deux boutons. Le premier, situé à gauche, permet de supprimer uniquement le dernier match joué, qui est le plus récent. Le second bouton, situé à droite, permet de supprimer tous les matches de l'historique. Pendant le chargement de la page, un TextView affiche le message "Chargement en cours..." et s'il n'y a aucun match, le texte affiché sera "Aucun match joué".

Pour parvenir à ce résultat, j'ai dû utiliser plusieurs méthodes. Dans un ancien schéma, chaque attribut de chaque match était affiché dans un TextView séparé, ce qui rendait la consultation des scores, du temps, etc. de chaque match fastidieux.

Cependant, j'ai amélioré cette mise en page en regroupant les attributs de chaque match de manière plus claire et intuitive. Ainsi, les informations de chaque match sont présentées de manière structurée et facilement lisibles, ce qui facilite la consultation de l'historique des matches.



Avant :



Après :



Avant toute chose, il est important de savoir que les données à afficher dans l'historique sont récupérées depuis la page en direct du match en cours. Lorsque je sauvegarde un match en cours, les informations correspondantes sont ajoutées à la base de données, plus précisément dans le nœud "Game".



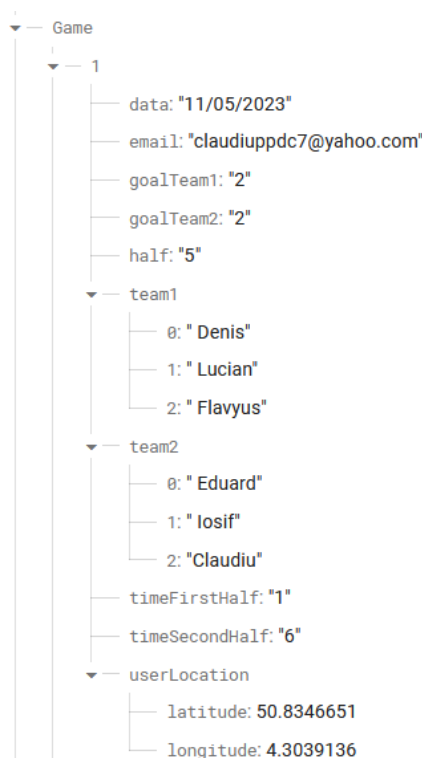
Lorsque je sauvegarde un match en cours, je récupère toutes les valeurs pertinentes de ce match et les ajoute dans un nouveau nœud dans la base de données. Cela me permet d'organiser les attributs de chaque match de manière structurée et accessible.

Chaque match enregistré possède un ensemble d'attributs clés qui sont sauvegardés pour une consultation ultérieure. Ces attributs comprennent des informations telles que le

numéro de la partie, l'adresse e-mail de la personne ayant créé le match, la date du match, les joueurs des deux équipes, le score final ainsi que les durées des différentes périodes de jeu et des pauses.

En ajoutant ces attributs dans un nouveau nœud de la base de données, je peux facilement les récupérer lors de l'affichage de l'historique des matches. Cela garantit que toutes les informations pertinentes sont disponibles et organisées de manière cohérente.

Lorsque l'utilisateur accède à la page de l'historique, les attributs de chaque match sont extraits de la base de données et affichés de manière claire et lisible. Les informations sont présentées dans un ordre chronologique, permettant ainsi de visualiser facilement les détails de chaque match enregistré.



Dans la version précédente (à gauche), chaque cadre visible était en réalité un TextView affichant une valeur récupérée de la base de données. Par exemple, lors de l'enregistrement d'un match, les scores des équipes 1 et 2 étaient stockés dans Firebase. Dans mon code, j'accédais à la base de données pour récupérer ces données, puis je les affichais dans le TextView "Scores". Ce processus était répété pour chaque cadre affiché dans l'historique.

Au début, j'ai rencontré une difficulté liée à l'affichage de l'heure et du score dans un ordre différent. Cela signifiait que la partie 1 ne correspondait pas aux données affichées dans l'historique. Heureusement, j'ai pu résoudre ce problème en utilisant l'objet TreeMap. Cette structure de données m'a permis de trier ma Map en parcourant les nœuds de ma base de données, assurant ainsi un affichage cohérent dans l'historique.

Par la suite, j'ai décidé de modifier l'affichage en utilisant l'image de droite. Cette modification m'a permis de récupérer de manière synchrone tous les attributs de ma base de données en une seule fois grâce à l'utilisation de SnapShot. Cette approche s'est révélée plus efficace et m'a permis de simplifier le processus de récupération et d'enregistrement des données dans le reste de mon application.

La mise en place des boutons a également été une étape importante. Pour le bouton "Tout supprimer", la logique était simple : il récupérait simplement le nœud "Game" de ma base de données et supprimait toutes les valeurs qui y étaient stockées. En revanche, la suppression du dernier match était plus complexe. Il fallait vérifier quel match avait été ajouté en dernier en examinant le numéro de la partie. En fonction de cela, si c'était le dernier match, il était simplement supprimé. Cela impliquait la mise en place de certaines conditions supplémentaires pour réaliser cette fonctionnalité.

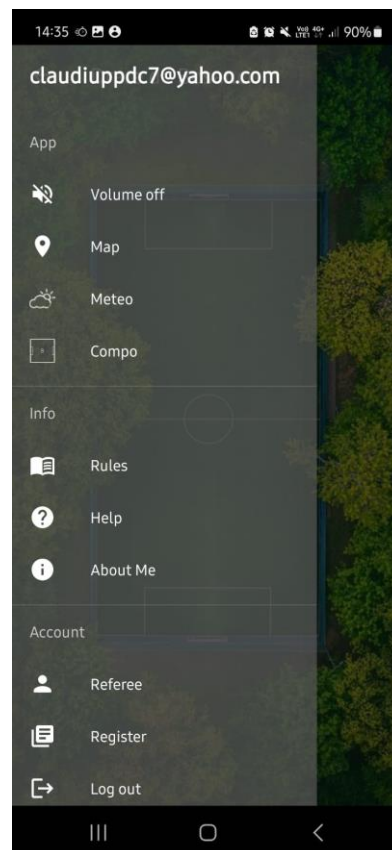
Dans l'ensemble, ces ajustements et améliorations ont permis d'obtenir un affichage plus clair et cohérent dans l'historique des matches, tout en facilitant la récupération et la gestion des données de la base de données. Cela a été une expérience gratifiante de voir que tout fonctionnait correctement et de manière efficace.

## Menu

Page d'accueil :



Menu :



Pour réaliser le menu, j'ai suivi un tutoriel disponible sur YouTube :

[https://www.youtube.com/watch?v=6mgTJdy\\_di4&t=640s](https://www.youtube.com/watch?v=6mgTJdy_di4&t=640s)

Ensuite, j'ai adapté le menu en fonction des besoins de mon application.

Comme on peut le constater, le menu est organisé en plusieurs catégories, avec différents éléments. Dans la catégorie "App", nous avons des options telles que le volume, la carte, la météo et la composition. La catégorie "Info" comprend trois pages d'informations : Règles, Aide et À Propos de Moi. Enfin, dans la dernière catégorie, nous avons la section "Compte", qui comprend les pages d'inscription, de connexion et une page spéciale pour l'administrateur.

Ce menu offre une navigation claire et intuitive pour l'utilisateur. Il lui permet d'accéder facilement aux différentes fonctionnalités de l'application, qu'il s'agisse de gérer le son, d'afficher une carte, de consulter des informations importantes ou de gérer son compte utilisateur.

L'utilisation de catégories et d'éléments bien organisés dans le menu rend la navigation fluide et facilite la recherche des fonctionnalités spécifiques. L'utilisateur peut sélectionner une catégorie, puis choisir l'option qui correspond à ses besoins.

## Musique

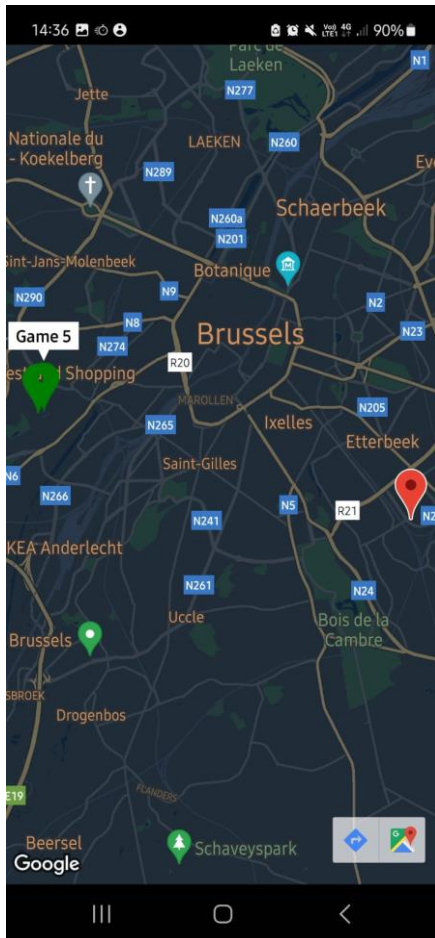
Je vais commencer par expliquer la musique de l'application. Lorsque cette fonctionnalité est activée, une musique se lance et se met en pause lorsque je la réactive. Cette fonctionnalité est réalisée grâce à la classe MediaPlayer, qui permet de récupérer le fichier audio à partir du dossier "raw" et de le lire.

La classe MediaPlayer offre des méthodes pour contrôler la lecture des fichiers audio. Dans ce cas, j'ai utilisé la méthode "start()" pour démarrer la lecture de la musique lorsque l'utilisateur appuie sur l'option correspondante. Lorsque la musique est en cours de lecture, si l'utilisateur appuie à nouveau sur l'option, j'ai utilisé la méthode "pause()" pour mettre la musique en pause.

Cela permet d'ajouter une expérience sonore agréable à l'application, en créant une ambiance musicale pour l'utilisateur. Il peut choisir d'activer ou de désactiver la musique selon ses préférences en appuyant sur l'option correspondante dans le menu.



## Map



Lorsque l'utilisateur appuie sur l'option "Map" dans le menu, une page dédiée s'ouvre, affichant une carte Google que j'ai réalisée en suivant deux tutoriels. Le premier tutoriel explique comment commencer avec l'API Android de Google Maps, disponible à l'adresse suivante :

<https://developers.google.com/maps/documentation/android-sdk/start?hl=fr>

Le deuxième tutoriel concerne la création d'un projet dans la console Google Cloud, nécessaire pour utiliser l'API Maps, et peut être trouvé ici :

<https://console.cloud.google.com/welcome?project=tchoo-tchoo-tracher&hl=fr>

La carte affichée est centrée sur la position actuelle de l'utilisateur et un marqueur rouge est placé à l'emplacement correspondant à l'école HELB. Des marqueurs verts sont également ajoutés pour indiquer la localisation de chaque match de football. Si l'utilisateur clique sur l'un de ces marqueurs, le numéro de la partie associée à cette localisation est affiché.

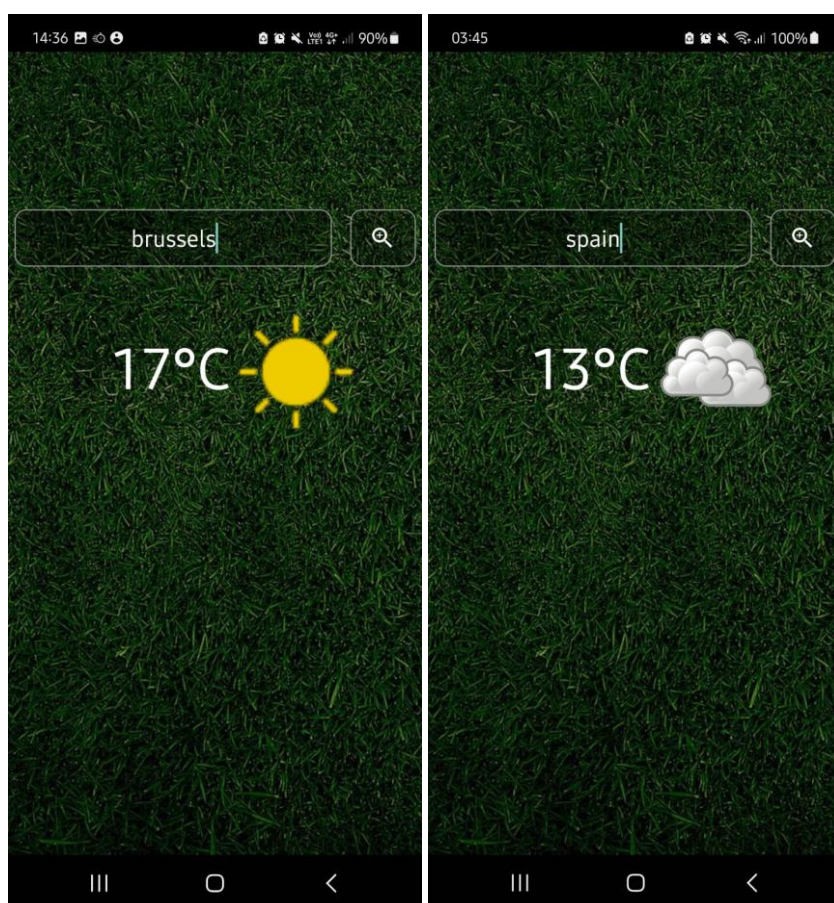
La réalisation de cette fonctionnalité a été assez simple. Lorsque j'enregistre un match, j'enregistre également dans ma base de données les données de latitude et de longitude correspondantes à la localisation du match. Ainsi, lors de l'affichage de la carte, j'utilise ces données pour placer les marqueurs aux emplacements appropriés.

```
▼ — userLocation
  — latitude: 50.8346651
  — longitude: 4.3039136
```

En récupérant les données de longitude et de latitude de ma base de données, j'ajoute ensuite un marqueur à cette position spécifique sur la carte, accompagné du numéro de la partie correspondante. Ainsi, chaque marqueur vert représente un match de football avec son numéro associé.

Cette fonctionnalité permet à l'utilisateur de visualiser facilement les lieux des matchs de football sur une carte interactive. Cela peut être utile pour planifier des déplacements ou obtenir des indications sur les lieux de jeu. Grâce à l'intégration de l'API Google Maps, l'application offre une expérience visuelle enrichissante et facilite la navigation géographique pour les utilisateurs.

## Météo



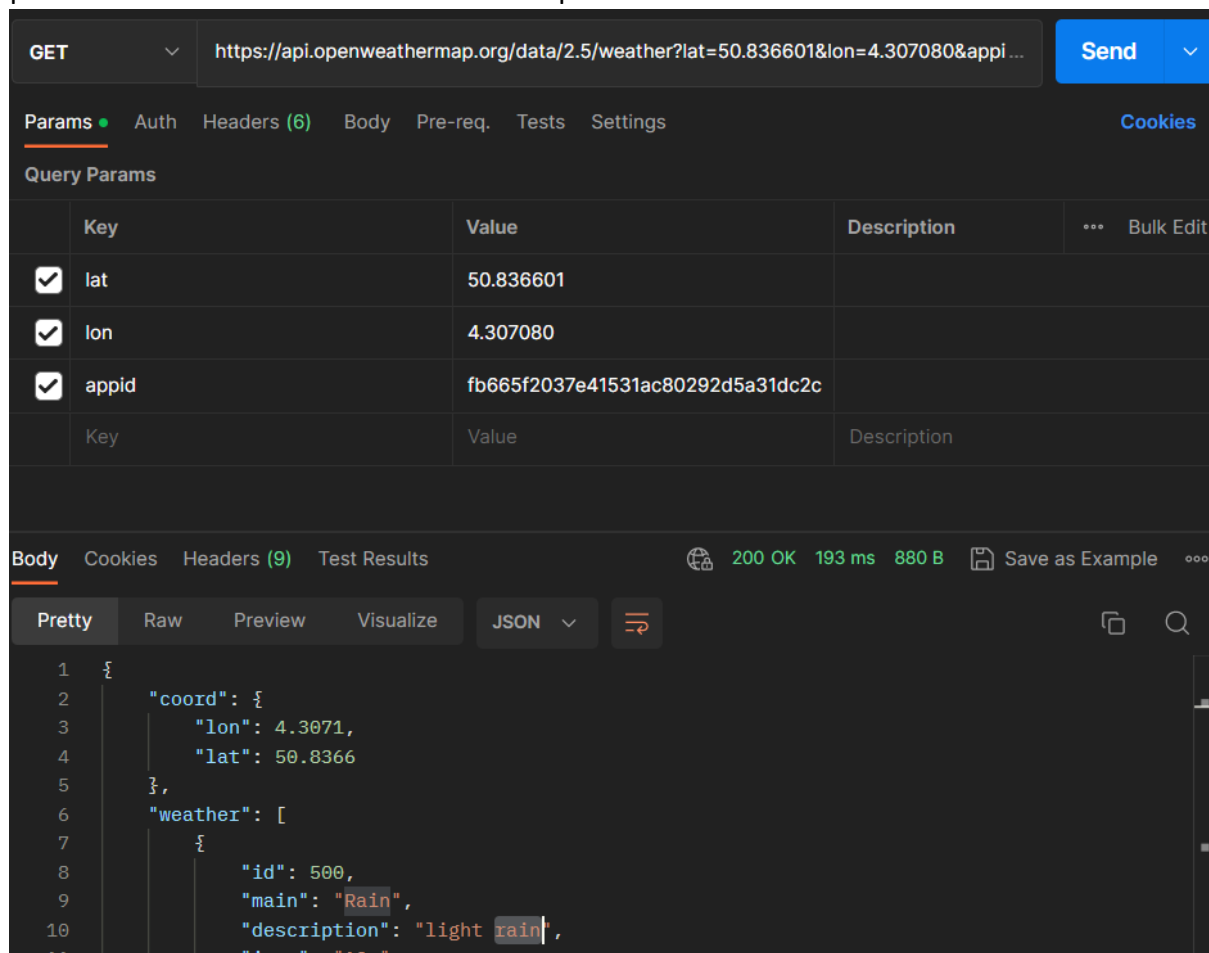
La météo n'était pas une fonctionnalité essentielle de mon application, mais j'ai décidé de l'intégrer à des fins d'expérience personnelle, car cela pouvait être pertinent. Lorsque nous souhaitons jouer un match à un endroit spécifique, il est souvent intéressant de savoir s'il fera beau ou non.

J'ai utilisé une API météorologique provenant de <https://openweathermap.org/api>.

J'ai suivi une démarche similaire à celle que j'ai utilisée pour l'API des positions des joueurs en utilisant l'application Postman, qui est une ressource très utile. Grâce à cela, j'ai pu facilement implémenter les requêtes dans mon code pour utiliser l'API météorologique.

J'ai ajouté un champ de texte et un bouton à mon application. Lorsque l'utilisateur clique sur le bouton, le texte saisi est récupéré. Si la ville ou le pays saisis sont valides, les conditions météorologiques actuelles de cette localisation sont affichées. Si la température est supérieure à 15°C, une image représentant le soleil est affichée, sinon une image représentant des nuages est affichée.

Cette fonctionnalité permet aux utilisateurs de consulter rapidement les informations météorologiques pour une ville ou un pays spécifique, ce qui peut être utile lors de la planification d'activités extérieures telles que des matchs de football.



GET <https://api.openweathermap.org/data/2.5/weather?lat=50.836601&lon=4.307080&appid=fb665f2037e41531ac80292d5a31dc2c> Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	lat	50.836601			
<input checked="" type="checkbox"/>	lon	4.307080			
<input checked="" type="checkbox"/>	appid	fb665f2037e41531ac80292d5a31dc2c			
	Key	Value	Description		

Body Cookies Headers (9) Test Results 200 OK 193 ms 880 B Save as Example

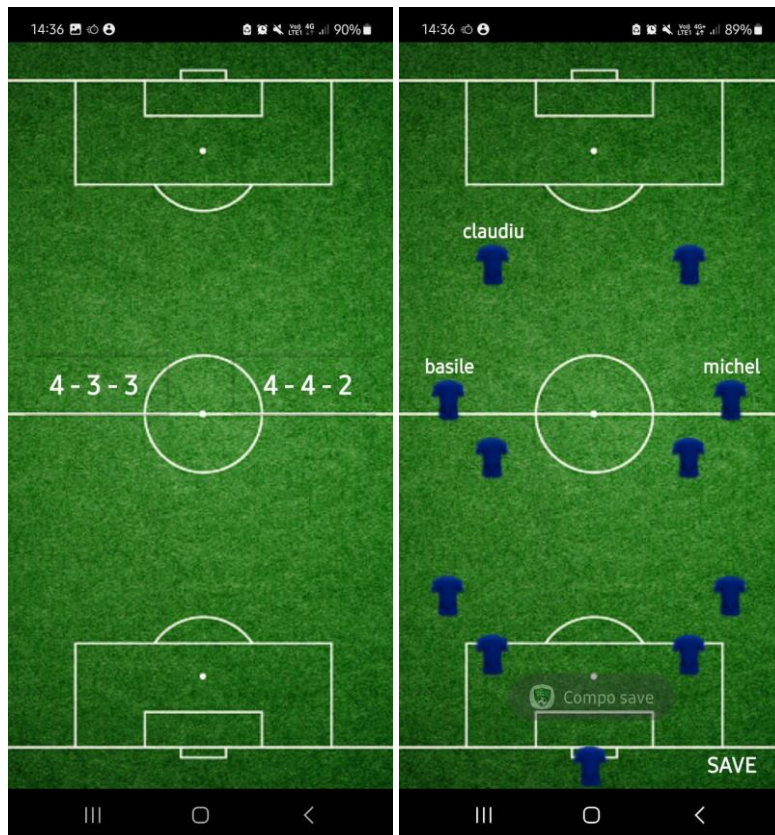
Pretty Raw Preview Visualize JSON

```

1  {
2    "coord": {
3      "lon": 4.3071,
4      "lat": 50.8366
5    },
6    "weather": [
7      {
8        "id": 500,
9        "main": "Rain",
10       "description": "light rain",
11       "icon": "03d"
12     }
13   ]
14 }

```

## Compo

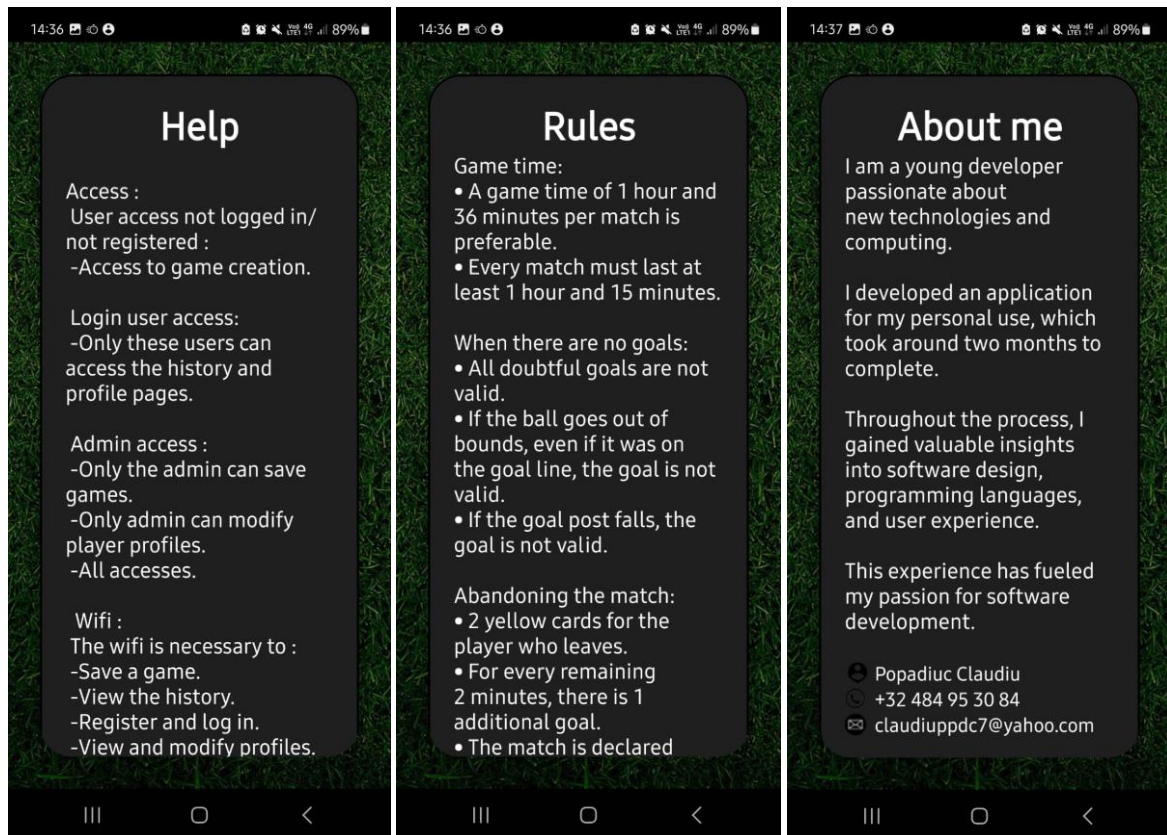


La composition est une fonctionnalité que j'ai ajoutée à mon application pour répondre à mes besoins personnels. Lorsque je souhaite jouer un match avec 11 personnes, il est important d'avoir une composition d'équipe définie. J'ai donc intégré les compositions classiques telles que le 4-4-2 et le 4-3-3, afin de pouvoir les utiliser facilement lorsque j'en ai besoin. Pour accéder à ces compositions, il suffit de cliquer sur le haut du maillot dans l'interface de composition. On peut alors saisir le nom des joueurs dans les positions correspondantes. Une fois tous les joueurs ajoutés, il est possible de sauvegarder la composition. Le système de récupération et d'ajout des compositions dans la base de données est le même que celui utilisé pour l'historique des matches.

En intégrant cette fonctionnalité, j'ai veillé à ce qu'elle soit intuitive et conviviale. L'utilisateur peut simplement cliquer sur les positions de joueur dans la composition, saisir les noms et enregistrer le tout en quelques étapes simples. Cette simplicité d'utilisation permet de gagner du temps et de rendre l'expérience utilisateur agréable.



## Info



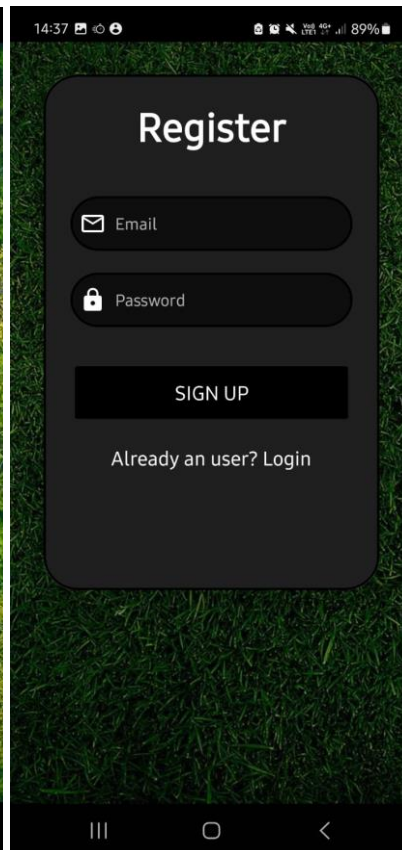
Les 3 Pages ne sont que des pages informatives, elles ne contiennent rien de spécial.

## Le processus d'inscription/Connexion.

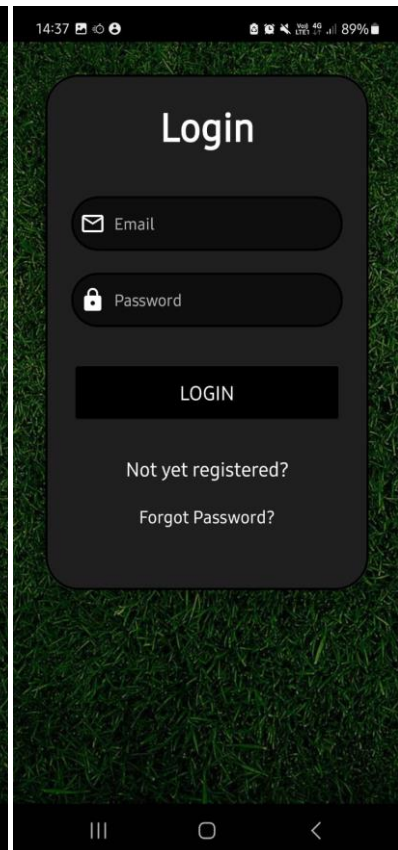
Page d'accueil :



Page d'inscription :



Page de connexion :



Sincèrement, je n'ai pas rencontré de difficultés pour implémenter le system de connexion et d'inscription. J'ai suivi un tutoriel sur YouTube qui m'a permis de comprendre et de développer cette fonctionnalité dont je vais parler plus tard. Au début, j'avoue que c'était compliqué et j'ai dû tester cela dans un nouveau projet avant de pouvoir l'implémenter dans mon projet actuel. Cependant, après quelques heures, j'ai réussi à faire fonctionner l'inscription et la connexion. Mais je vais rentrer dans les details pour cela.

Avant tout, il est important de préciser que mon application n'exige pas nécessairement une inscription. Cependant, si l'utilisateur décide de ne pas s'inscrire, certaines fonctionnalités de l'application lui seront inaccessibles (ce qui serait dommage, d'ailleurs). Par exemple, sur la page d'accueil, les boutons "Historique" et "Statistiques" ne seront disponibles qu'en étant connecté. Par conséquent, il sera obligatoire d'avoir un compte utilisateur pour profiter pleinement de l'application. Toutefois, si l'utilisateur souhaite réellement éviter de s'inscrire, il pourra créer un match qui ne pourra pas être sauvegardé.

Ainsi, si l'utilisateur désire d'avoir un avec un maximum de sensation sur l'application, il devra créer un compte. Comme vous pouvez le constater sur ma page d'inscription, il y a seulement

deux champs à remplir : l'adresse e-mail et un mot de passe pour créer un compte. Sur mon application, c'est très limité car l'application n'est pas poussée pour faire autre chose de plus complexe.

Pour mettre en place cette fonctionnalité, j'ai suivi un tutoriel disponible ici : <https://www.youtube.com/watch?v=TSsttJRAPH&t=650s>

J'ai dû utiliser Firebase qui est disponible directement dans Android Studio dans les Tools, j'ai connecté mon projet avec Firebase, dans la partie authentification pour cette partie d'inscription avec mail et mots de passe :

Identifiant	Fournisseurs	Date de création	↓	Dernière connexion	UID utilisateur
[REDACTED]	✉	7 avr. 2023		7 avr. 2023	oiPmxLYbEUbUk7sD2pp2kd7uYXp2
[REDACTED]	✉	7 avr. 2023		7 avr. 2023	C2mzRT7nJcUeVPe7xw5wpE8yEe...
[REDACTED]	✉	7 avr. 2023		7 avr. 2023	UclqHmRN2UOBHrbPMWKiThrCy...
[REDACTED]	✉	7 avr. 2023		23 avr. 2023	R4xTBC7scmPaCcdXT0gcnXmFF...
claudiuppd6@gmail.com	✉	15 mars 2023		21 avr. 2023	3Mv52KXheZZJ8ynV4SggV4mrlWI1
claudiuppd7@yahoo.com	✉	8 mars 2023		11 mai 2023	DHJzR1z2Uea5SpaUydKxSzG9n8...

Lignes par page : 50 1 – 6 of 6 < >

\*Afin de rester professionnel, j'ai voulu respecter la vie privée de mes utilisateurs alors j'ai baré en bleu leur adresse mail.

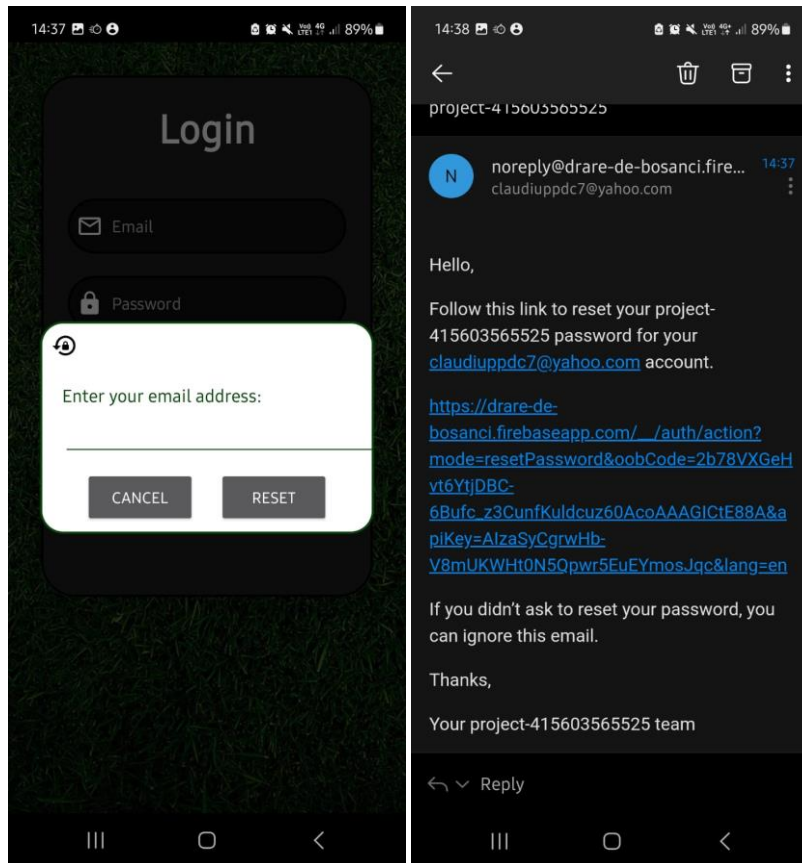
Ce qui a été compliqué pour réaliser cette fonctionnalité, c'est que j'aurais voulu utiliser la base de données en temps réel (Real Time Database) pour stocker mes utilisateurs, mais je n'ai pas réussi à le mettre en place. J'ai suivi plusieurs tutoriels et découpé mon projet en plusieurs parties, mais malheureusement, je n'ai pas réussi à le faire fonctionner. J'ai donc préféré me concentrer sur le système d'authentification, qui est amplement suffisant pour les besoins de mon application.

Pour entrer dans les détails, lorsque j'appuie sur le bouton " Sign Up", toute la partie du code s'exécute. Tout d'abord, je vérifie si le champ de l'adresse e-mail correspond bien à la structure d'une adresse e-mail avec le symbole "@" ; je vérifie également si le champ n'est pas vide et si le mot de passe contient au moins 8 caractères. Si toutes ces conditions sont remplies, j'ajoute l'utilisateur avec son adresse e-mail et son mot de passe dans le système d'authentification.

Pour la connexion, le principe est légèrement différent. Le système vérifie si l'e-mail et le mot de passe encodés correspondent aux utilisateurs disponibles dans le système d'authentification. Sinon, il affiche un message (Toast) indiquant une erreur de connexion.

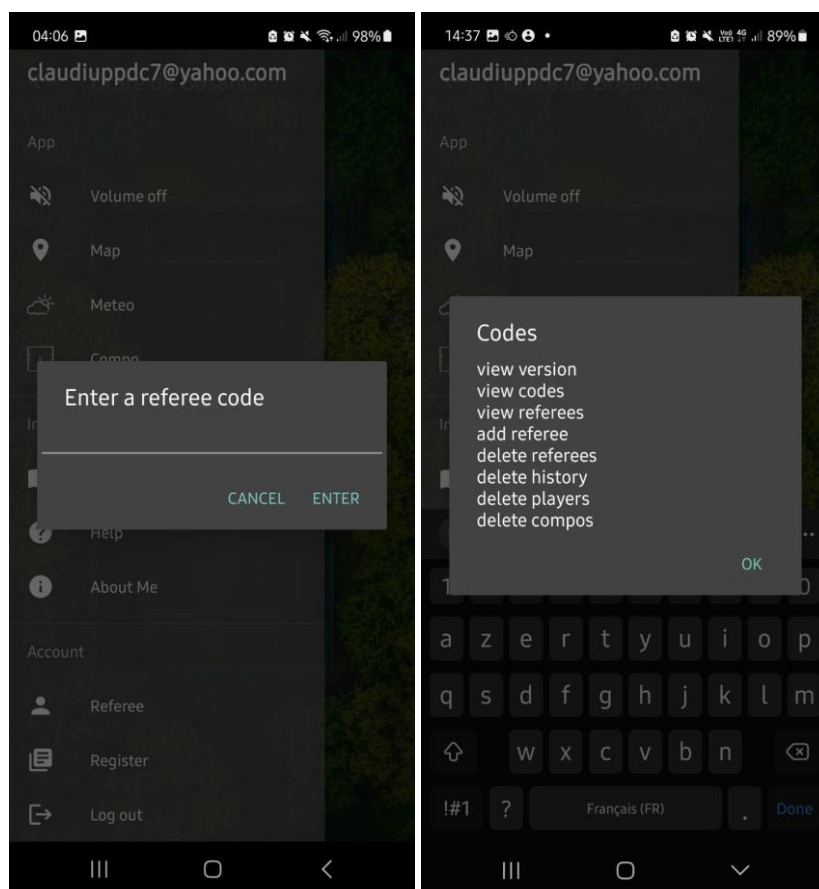


Si un utilisateur n'arrive plus à se connecter parce qu'il a peut-être oublié son mot de passe, un système de récupération de mot de passe est disponible sur la page de connexion. Voici ce qui apparaîtra lorsque j'appuierai sur " Forgot password ? " :



Un email de récupération de mots de passe sera alors envoyé, après avoir tester cette fonctionnalité, ça fonctionne bel et bien.

## Admin



Lorsqu'on appuie sur l'option "Arbitre", cela n'ouvre pas une nouvelle page mais affiche plutôt un générateur avec un champ de texte à remplir. Ce champ de texte est en fait un code, et même les caractères saisis sont cachés. Cette fonctionnalité permet simplement de devenir arbitre, c'est-à-dire administrateur de l'application. Comme expliqué précédemment, il y a deux types d'utilisateurs : les joueurs et les arbitres. Les joueurs qui créent un compte n'ont pas accès à toutes les fonctionnalités de l'application, telles que l'ajout ou la suppression d'un joueur. En effet, si tout le monde pouvait le faire, la saison de football deviendrait vite chaotique. Ainsi, j'ai veillé à ce que seuls les administrateurs aient accès à ces fonctionnalités, à savoir : la sauvegarde des matches, la suppression des matches dans l'historique, l'ajout et la suppression d'un joueur dans les statistiques et la modification des statistiques des joueurs.

Pour devenir administrateur, il faut entrer le code "add referee" et l'utilisateur devient directement administrateur. Pour toutes les fonctionnalités énumérées ci-dessus, je vérifie simplement dans ma base de données si l'utilisateur connecté fait partie de ma liste d'administrateurs. S'il est présent dans la liste, il aura accès à toutes les fonctionnalités.



```
|
└─ Referee
    ├── email11: "claudiupdc7@yahoo.com"
    └── email15: "test1@gmail.com"
```

C'est aussi simple que ça. J'ai ajouté plusieurs codes dans ce générateur qui permettent de réaliser plusieurs tâches directement depuis ce générateur sans avoir à passer par l'application, comme vous pouvez le voir sur le deuxième écran. Ces codes sont tous les codes disponibles.

## UI/UX

Cette section aborde en détail l'importance accordée à l'UI/UX, ainsi que la manière dont les besoins et préférences des utilisateurs ont été pris en compte, notamment lors de la séance de Speed Testing.

Avant d'aborder les retours du Speed Testing, j'avais déjà effectué des tests d'expérience de mon application auprès de certaines personnes de mon entourage. On m'a signalé quelques bugs d'affichage dans mon application, notamment dans le TextView du chronomètre qui n'affichait pas les unités de secondes sur les téléphones moins larges à mon avis. De plus, certaines images étaient décalées sur d'autres types de téléphones. J'ai donc apporté les modifications nécessaires et j'ai renvoyé l'application pour être testée à nouveau.

J'ai également rencontré un problème avec une personne qui ne recevait pas son mail de confirmation. Initialement, j'envoyais un mail de confirmation lors de l'inscription pour confirmer l'identité humaine de l'utilisateur et lui permettre de se connecter. La personne devait vérifier sa boîte mail et cliquer sur le lien reçu pour confirmer son compte. Cependant, une personne ne recevait tout simplement pas ce mail de confirmation, bien que son adresse mail soit correcte. Pour éviter ce type de problème à l'avenir, j'ai décidé de supprimer cette fonctionnalité, car elle n'était pas particulièrement utile dans mon application. À part ce problème, tout fonctionnait bien.

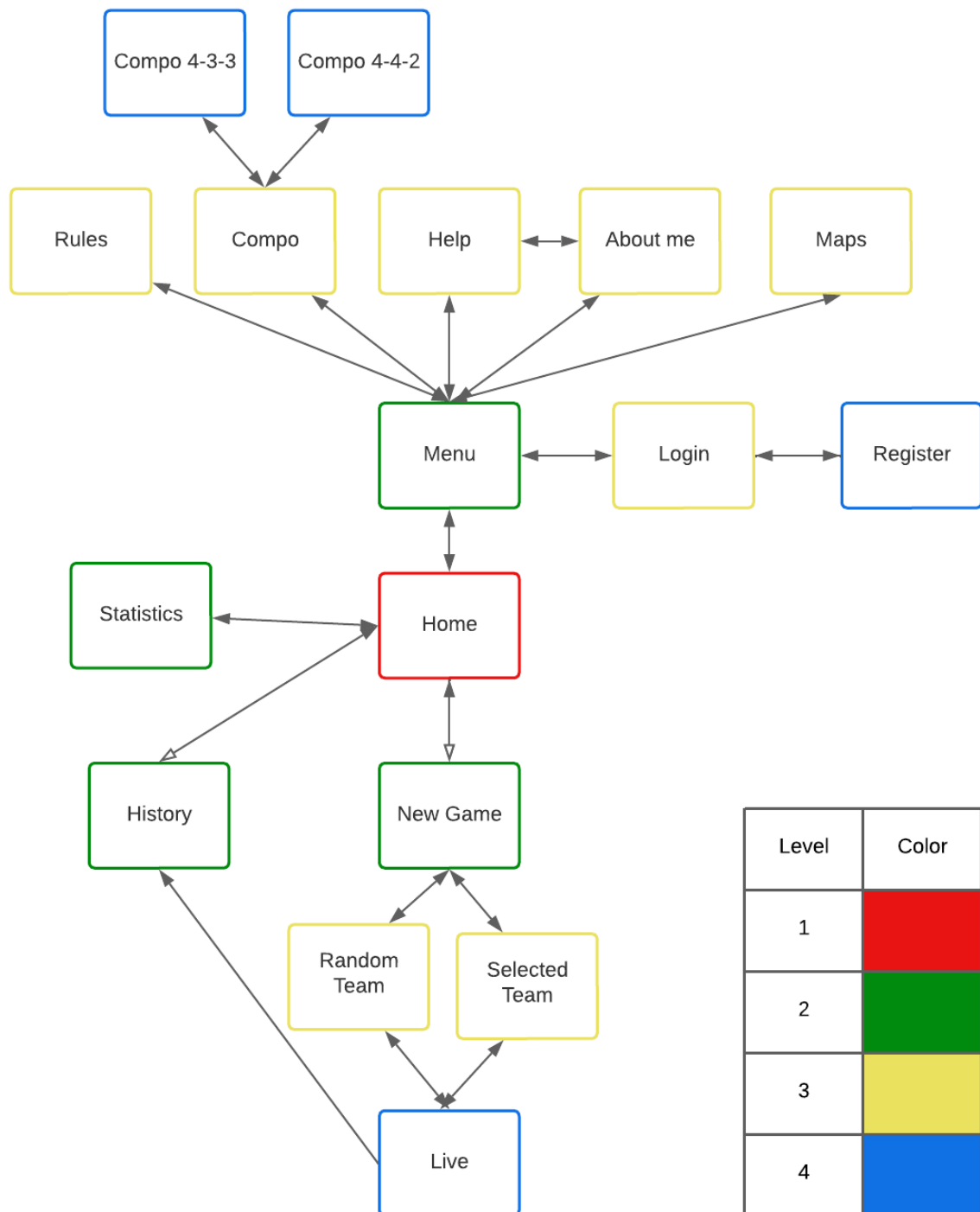
Lors d'une de nos itérations précédentes, vous m'avez fait remarquer que les fonds d'écran de l'application n'étaient pas adaptés, ce qui rendait le texte peu visible. J'ai donc pris en compte votre feedback et j'ai modifié presque tous les fonds d'écran de l'application, comme vous pouvez le voir dans les captures d'écran figurant dans le rapport.

Voici un résumé des retours reçus des testeurs via les formulaires Google lors de l'activité Speed Testing, ainsi que la manière dont ces retours ont été pris en compte dans l'implémentation de ma version finale de l'application :

- Plusieurs testeurs ont suggéré qu'il serait utile de savoir quand on clique sur un joueur de la liste des joueurs lors de l'ajout de joueurs aux équipes de football. J'ai donc modifié cela pour que, dès qu'un joueur est cliqué, il disparaisse de la liste afin qu'il ne soit plus cliquable.
- Lors de l'ajout du temps, j'ai limité le temps maximum à deux chiffres, c'est-à-dire qu'il n'y aura pas plus de 99 minutes pour une période de jeu.
- Les testeurs ne savaient pas où cliquer pour démarrer le match en direct, alors j'ai agrandi le bouton et ajouté un grand texte pour bien comprendre cela.
- Il a également été suggéré qu'il serait préférable de ne pas pouvoir commencer un match si les équipes ne sont pas encore déterminées pour le mode RANDOM. J'ai donc fait en sorte que le match ne commence pour la partie RANDOM que si l'on clique sur le grand bouton PLAY.

- L'un des problèmes rencontrés était que l'on pouvait quitter le live du match sans sauvegarder, ce qui supprimait complètement les données du live. Pour résoudre ce problème, j'ai ajouté un avertissement à l'utilisateur avant de quitter la page live, indiquant que s'il continue, cela risque de supprimer la session. S'il décide d'annuler, il revient à la page et peut sauvegarder le match avant de le quitter.
- Selon un testeur, il y avait trop de pop-up dans la partie météo, alors j'en ai enlevé.
- Un autre testeur a suggéré qu'il serait préférable d'indiquer les fonctionnalités de l'application et comment cela fonctionne avec des textes au-dessus des champs à compléter pour indiquer comment ils doivent être remplis et comment l'application fonctionne.

# Analyse



## Explication du diagramme :

Comme nous pouvons le constater, la première page que nous verrons sera en rouge, ce sera la page d'accueil qui permettra d'accéder à toutes les fonctionnalités. À partir de là, les utilisateurs pourront accéder au menu déroulant qui leur permettra de se connecter ou de s'inscrire, ou tout simplement utiliser les fonctionnalités de l'application, telles que la map, les pages de composition avec des schémas tels que 4-4-2 ou 4-3-3, ainsi que toutes les pages d'information comme "About me", "Rules" ou "Help".

Les fonctionnalités principales, telles que la création de matches, sont disponibles dans la section "Home". À partir de là, nous aurons le choix de l'équipe, et en fonction de ce choix, nous arriverons dans la diffusion en direct du match. À partir de là, nous pourrons accéder directement à l'historique de tous les matches, ou cela sera également possible depuis la page d'accueil. Il y aura également la possibilité de consulter les statistiques, où les performances de tous les joueurs de l'application seront affichées.

En résumé, la page d'accueil rouge est la porte d'entrée de toutes les fonctionnalités de l'application. Elle permettra aux utilisateurs de se connecter, de s'inscrire et d'utiliser les différentes fonctionnalités telles que la carte, les pages de composition et les informations générales. De plus, la section " Home " offrira la possibilité de créer des matches, de suivre leurs matchs en direct et de consulter l'historique des matches. Les statistiques des joueurs seront également disponibles pour une analyse approfondie.



## Limitations et développement futur

Voici quelques questions auxquelles je vais essayer de répondre au maximum pour mon application :

### 1. Dans quels cas d'utilisation mon application pourrait ne pas fonctionner ?

En ce qui concerne les crashes, normalement l'application ne devrait pas crasher. Cependant, il y a quelques points qui pourraient être extrêmement gênants, par exemple :

- L'enregistrement des matches dans l'historique. Dans mon cas, il n'y aura probablement pas plus de 50 matches par saison, mais si l'on imagine seulement 1000 matches, il deviendrait extrêmement fastidieux de faire défiler 1000 matches dans l'historique. Il aurait été préférable d'ajouter une barre de recherche par équipe, score ou buts.
- Si l'on imagine 1000 matches dans l'historique, il y aurait également un problème lorsqu'on souhaite supprimer un match en particulier, par exemple le 750ème. Étant donné que je ne peux supprimer que le dernier match ou tous les matches, il serait impossible de supprimer uniquement le 750ème match.
- De même pour les statistiques, j'ai seulement 12 joueurs dans mes statistiques. Je peux en ajouter autant que je le souhaite, mais imaginons que j'en aie 100. Il serait alors nécessaire de chercher parmi tous les joueurs pour en retrouver un, il faudrait donc également une barre de recherche par prénom. Si nous avons 10 000 joueurs, le classement de 1 à 10 000 serait très long, il faudrait donc le diviser en fonction des joueurs.
- Dans mes statistiques, je peux ajouter un joueur avec un nom de 1000 caractères, et l'application l'acceptera probablement, mais cela causerait des problèmes d'affichage dans le choix des joueurs avec un spinner énorme.
- Des problèmes similaires peuvent se produire pour tous les champs de mes statistiques, par exemple, si je mets plus de 3 chiffres dans les cases avec des chiffres tels que les cartons jaunes, seuls les 3 premiers chiffres seront affichés, ce qui donnera des valeurs incorrectes.
- Pour l'attribut "fame", si vous écrivez 10 000 caractères, la case s'agrandira et vous ne verrez plus ce qui se trouve en dessous, vous devrez donc faire défiler énormément.
- Les points réagissent en fonction des champs des statistiques, j'ai également remarqué que cela se produit pour "fame", la case des points s'agrandit si de grandes valeurs sont ajoutées.
- Mes compositions sont limitées aux plus connues, le 4-4-2 et le 4-3-3, mais si l'on souhaite une autre formation, il est impossible d'en créer une autre sans modifier le code.
- Pour la météo, je ne peux pas choisir certaines villes qui ne sont pas proposées par l'API, et si je fais une faute d'orthographe, cela ne fonctionne pas du tout.

- Mon API est limitée à 100 requêtes par jour, car seulement 100 sont permises gratuitement. Par conséquent, elle ne fonctionnera que 100 fois par jour. Si 100 utilisateurs consultent les statistiques une seule fois par jour, l'API sera déjà épuisée. Il faudrait donc payer chaque mois pour avoir droit à plus de requêtes.
- En ce qui concerne les codes admin, si une personne pirate mon application et trouve les codes admin, elle pourrait supprimer toute ma base de données et faire ce qu'elle souhaite avec les données de mon application.
- Lors de la création d'un match, si l'on ajoute un joueur manuellement, il peut entrer autant de caractères qu'il le souhaite. Par exemple, si on en met 5000, cela détruirait complètement l'affichage de mes équipes dans le LIVE et cela ne ressemblerait plus à rien. J'ai limité le temps d'ajout à la mi-temps et à la pause à 2 chiffres, mais il reste tout de même absurde d'avoir un match avec 99 minutes par mi-temps et 99 minutes de pause.
- Il peut également y avoir un gros problème de récupération des données si l'on n'a pas de connexion Wi-Fi ou 4G. Dans ce cas, on ne peut presque pas utiliser l'application, car tout fonctionne en Wi-Fi, que ce soit l'enregistrement des matches, l'affichage de l'historique, les statistiques des joueurs, les classements, etc.

2) Y a-t-il des aspects techniques non traités ?

Non, toutes les fonctionnalités demandées ont été réalisées.

3) Si j'avais plus de temps pour le projet, qu'aurais-je amélioré ?

L'un des points les plus importants que j'aurais améliorés aurait été la possibilité de personnaliser l'application pour un usage autre que le mien. Ainsi, n'importe qui aurait pu créer une saison, et chaque saison aurait eu son code d'accès avec ses participants, ses matches, son historique, sa carte et ses statistiques, et tout ce qui va avec. Cela aurait permis d'avoir une véritable application utilisable par n'importe qui qui aime le football.

## Play Store

Avant d'aborder le sujet principal, je tiens à mentionner que j'ai essayé de mettre mon application sur le Google Play Store. J'ai consacré du temps à suivre un tutoriel sur YouTube :

<https://www.youtube.com/watch?v=7h6aouRXpWM&t=947s>.

Tout se déroulait bien lors de mes tests pour mettre mon application sur la plateforme, jusqu'à ce que j'arrive à l'étape d'upload de mes bundles et APK. Je les avais correctement préparés, mais j'ai rencontré un problème lors de l'ajout de mon application. Une erreur est apparue, indiquant que je ne pouvais pas importer l'application car mon package était "com.example". Selon ChatGpt, voici pourquoi cela n'est pas possible :

Le Google Play Store a des règles strictes concernant les packages d'application, dans un souci de sécurité et pour éviter toute confusion ou conflit entre les applications. Le package d'application, généralement au format "com.example", sert à identifier de manière unique chaque application sur le Play Store. L'utilisation du package "com.example" est généralement réservée aux exemples de code ou aux démonstrations fournies par les développeurs pour aider les autres à comprendre comment créer des applications. Cependant, il n'est pas autorisé pour les applications réelles destinées à être publiées sur le Play Store.

Lorsque vous soumettez une application au Play Store, vous devez fournir un package d'application unique qui n'est pas déjà utilisé par une autre application existante. Cela garantit que chaque application a son identifiant unique et peut être téléchargée et mise à jour indépendamment. En utilisant un package d'application générique comme "com.example", il y aurait un risque de conflit avec d'autres applications qui pourraient utiliser le même package. Cela pourrait entraîner des problèmes de sécurité, des erreurs de mise à jour ou d'autres complications indésirables. Il est donc important de choisir un package d'application unique et spécifique pour votre application afin de pouvoir la publier en toute sécurité sur le Play Store et éviter tout problème potentiel avec d'autres applications existantes.

Ainsi, l'idée est simplement de modifier le nom de mon package. Cependant, cela s'est avéré plus compliqué que prévu. Lorsque j'ai essayé de le faire, cela a provoqué de nombreux problèmes, l'application ne voulait plus run du tout. J'avais push trop rapidement la dernière version sur GitHub, ce qui a également affecté le fonctionnement de GitHub. J'ai passé une bonne partie d'une nuit à résoudre ce problème en testant différentes méthodes, et j'ai finalement réussi à le résoudre sans même savoir exactement comment. En tout cas, je ne changerai plus jamais le nom de mon package s'il y a autant de risques dans un projet d'envergure. Le package a en réalité de nombreuses dépendances qui ne permettaient pas de simplement modifier son nom.

J'ai donc abandonné l'idée de publier mon application sur le Google Play Store. J'aurais au moins essayé.

## Conclusion

Je commence ma conclusion en exprimant ma fierté pour ce projet. Je pense que c'est la création la plus importante que j'ai réalisée en informatique. J'ai attendu longtemps le cours sur le développement mobile, depuis mes débuts en informatique.

Ce qui me passionnait le plus, c'était de réaliser l'application que j'ai créée, et j'ai pu le faire grâce à vous. En effet, il est parfois difficile de découvrir seul une nouvelle plateforme comme Android Studio, c'est pourquoi votre cours m'a été d'une grande aide.

Au cours de ce projet, j'ai pu apprendre énormément de choses, notamment en ce qui concerne le développement mobile. J'ai également approfondi mes connaissances du langage Java, notamment toutes les liaisons possibles avec le XML et tout ce qui concerne l'affichage. Je me rends compte que j'apprécie davantage le front-end que le back-end, et ce projet m'a permis de progresser considérablement dans ce domaine. J'ai découvert de nombreuses astuces qui m'ont permis d'améliorer l'interface graphique au fil des jours.

J'attends avec impatience de suivre le cours de Mobile 2 afin de poursuivre le développement mobile et d'apprendre encore plus de choses. Si possible, j'aimerais utiliser une plateforme comme React pour créer des applications pour Apple, car d'après mes recherches, c'est ce qui est recommandé.

Je tiens à vous remercier sincèrement d'avoir accepté ce sujet et de m'avoir supervisé tout au long de ce projet. Je ne dis pas cela pour vous impressionner, car je sais que j'ai réalisé toutes les fonctionnalités et, sans arrogance, je suis convaincu que je vais réussir. Cependant, je tiens à souligner que vous m'avez énormément aidé dans ce projet, et sans votre soutien, je n'aurais jamais pu aller aussi loin. Les itérations ont été d'une grande aide, car vous me guidiez et me donniez des conseils pour améliorer mon application, en particulier lors des sessions de test de vitesse, qui m'ont permis de recueillir plusieurs retours constructifs.

Je termine ce rapport en vous donnant un conseil :

Continuez à vous intéresser aux élèves comme vous l'avez fait jusqu'à présent !