**Rapport TP Transversal**

**Pari Local**

**Aout 2021**

**Membres de l’équipe :**

**Elliot Kevin RAKOTOMANANA**

**Mahefarivo Sombiniaina RAVOAHANGY**

**Toavina Fanasina RAZAFY**

**LISTE DES FIGURES**

[Figure 1 – Planification développement serveur Grails 14](#_Toc80029656)

[Figure 2 – Planification développement serveur node 15](#_Toc80029657)

[Figure 3 - Planification développement du back office 15](#_Toc80029658)

[Figure 4 - Planification développement du client web 15](#_Toc80029659)

[Figure 5 - Planification développement du client lourd 15](#_Toc80029660)

[Figure 6 - Planification développement de l'application mobile 15](#_Toc80029661)

[Figure 7 - Login back office 17](#_Toc80029662)

[Figure 8 - gestion paris disponible 17](#_Toc80029663)

[Figure 9 - Gestion paris disponibles sous forme d'un tableau 18](#_Toc80029664)

[Figure 10 - Gestion des équipes 18](#_Toc80029665)

[Figure 11 - Liste demandes de jeton 18](#_Toc80029666)

[Figure 12 - Map 19](#_Toc80029667)

[Figure 13 - Client web liste des paris disponibles 19](#_Toc80029668)

[Figure 14 - Client web palcer un pari 20](#_Toc80029669)

[Figure 15 - Client web information de l'utilisateur 20](#_Toc80029670)

[Figure 16 - Client web faire une demande de jeton 20](#_Toc80029671)

[Figure 17 - Client web historique des paris 21](#_Toc80029672)

[Figure 18 - Client web détail d'un pari 21](#_Toc80029673)

[Figure 19 - Client lourd paris disponibles 22](#_Toc80029674)

[Figure 20 - Client lourd parier 22](#_Toc80029675)

[Figure 21 - Client lourd modification information utilisateur 23](#_Toc80029676)

[Figure 22 - Client lourd faire une demande de Jeton 23](#_Toc80029677)

[Figure 23 - Client lourd historique des paris 24](#_Toc80029678)

[Figure 24 - Client lourd détails d'un pari 24](#_Toc80029679)

[Figure 25 - Client mobile accueil 25](#_Toc80029680)

[Figure 26 - Client mobile login 25](#_Toc80029681)

[Figure 27 - Client mobile inscription 26](#_Toc80029682)

[Figure 28 - Client mobile paris disponibles 26](#_Toc80029683)

[Figure 29 - Client mobile saisie pari 27](#_Toc80029684)

[Figure 30 - Client mobile placer un pari 27](#_Toc80029685)

[Figure 31 - Client mobile faire une demande de jeton 28](#_Toc80029686)

[Figure 32 - Architecture 28](#_Toc80029687)

[Figure 33 - Exemple code Grails 29](#_Toc80029688)

[Figure 34 - Exemple code Node 30](#_Toc80029689)

[Figure 35 - Exemple code vue Back Office 30](#_Toc80029690)

[Figure 36 - CU Parier 31](#_Toc80029691)

**LISTE DES ACRONYMES**

**IDE :** Integrated Development Environment

**IHM :** Interface Home Machine

**SOMMAIRE**

[**1.** **INTRODUCTION GENERALE** 6](#_Toc80040993)

[**2.** **PRESENTATION DU PROJET** 6](#_Toc80040994)

[**3.** **ETAT DE L’ART** 7](#_Toc80040995)

[**4.** **ETUDE DE L’EXISTANT ET SOLUTION ENVISAGEE** 8](#_Toc80040996)

[4.1 Etude de l’existant 8](#_Toc80040997)

[4.2 Solution envisagée 8](#_Toc80040998)

[**5.** **DEMARCHE PROJET** 8](#_Toc80040999)

[5.1 Principes de la démarche projet 8](#_Toc80041000)

[5.1.1 Activités d’ingénierie logicielle 8](#_Toc80041001)

[5.1.2 Démarche projet adoptée 9](#_Toc80041002)

[5.1.3 Rôles et responsabilités 9](#_Toc80041003)

[5.1.4 Outils 9](#_Toc80041004)

[5.2 Contrainte et risques 10](#_Toc80041005)

[5.3 Démarche projet mise en œuvre 10](#_Toc80041006)

[5.4 Planification 10](#_Toc80041007)

[**6.** **EXIGENCES REALISEES DANS LE PROJET** 11](#_Toc80041008)

[6.1 Exigences fonctionnelles - Cas d’utilisation 12](#_Toc80041009)

[6.1.1 Cas d’utilisation pour le back office 12](#_Toc80041010)

[6.1.2 Cas d’utilisation pour les clients web, exécutable et mobile 12](#_Toc80041011)

[6.2 Exigences non fonctionnelles transverses 12](#_Toc80041012)

[6.3 Interfaces détaillées 12](#_Toc80041013)

[6.3.1 IHM 12](#_Toc80041014)

[Back Office : 12](#_Toc80041015)

[Client web : 15](#_Toc80041016)

[Client lourd : 17](#_Toc80041017)

[Application mobile : 21](#_Toc80041018)

[**7.** **CONCEPTION DU SYSTEME LOGICIEL REALISE DANS LE PROJET** 24](#_Toc80041019)

[7.1 Plate-forme technique 24](#_Toc80041020)

[7.2 Conception du logiciel développé 25](#_Toc80041021)

[7.2.1 Conception du code source 25](#_Toc80041022)

[7.1.1 Les composants et leur déploiement 28](#_Toc80041023)

[**8** **Tests du système logiciel** 29](#_Toc80041024)

[**9** **CONCLUSION** 29](#_Toc80041025)

[**10** **REFERENCES ET BIBLIOGRAPHIE** 30](#_Toc80041026)

# **INTRODUCTION GENERALE**

Un pari sportif est une opération qui consiste à miser de l’argent sur un potentiel vainqueur ou vaincu d’une rencontre sportive gérée par un bookmaker. Contrairement à ce qu’on pourrait penser, les paris sportifs ne datent pas d’aujourd’hui. En effet, cette activité existe depuis à peu près aussi longtemps que le sport existe. En ces temps, l’activité se fait davantage sur internet avec de nouvelles opportunités. Cela nous mène alors à voir ce projet intitulé « PariLocal ».

Ce projet sera réalisé pour tous les amateurs de pari sur tout plateforme afin qu’ils puissent faire leurs paris n’importe où ou ils veulent.

# **PRESENTATION DU PROJET**

L’application PariLocal est une plateforme de pari en ligne qui se base sur les paris sportifs. Ce projet a été conçu dans le but de faciliter le pari pour les parieurs mais aussi de diminuer les taches d’un bookmaker. Les parieurs auront un accès direct sur les matchs disponibles sur toutes plateformes. Cette application permet alors aux amateurs de pari de parier directement sur le site grâce à des jetons qu’il aurait acheté via les mobiles banking, le flux d’argent en liquide sera alors diminué et les transactions se feront du site, en jeton, vers les mobiles banking ou vice versa.

A cet effet, la plateforme PariLocal possède de nombreuses fonctionnalités mises à la disposition des parieurs et une interface dédiée à la demande.

Pour ce faire, l’application est divisée en cinq grands partis. Il est constitué de 3 clients et un back office à savoir :

* Client web est fait en angular
* Client mobile est fait sur Android
* Client Lourd est fait sur windows en winform
* Le back office en Angular
* Back end sera fait en Java (Grails) et en Node

En généralité, les backends vont recevoir et traiter les données des partis clients. Ils se communiqueront grâce à des webservices Rest

# **ETAT DE L’ART**

Nous allons prendre un site intitulé « Betplanet » pour comparer avec notre projet PariLocal.

Les critères de comparaisons sont :

* Les fonctionnalités
* Utilisabilité et accessibilité

|  |  |
| --- | --- |
| Pari Local | Bet Planet |
| Pari Local est un bookmaker spécialisé pour le football | Bet Planet est site de pari sportif, avec des jeux de casino |
| Pari Local est disponible en version mobile et application lourd exécutable sur un ordinateur | Bet Planet est disponible seulement sur le web |

Figure 1 - Tableau comparatif avec un autre site pari sportif

# **ETUDE DE L’EXISTANT ET SOLUTION ENVISAGEE**

## **Etude de l’existant**

Un pari sportif consiste à miser de l’argent sur la réalisation d’un évènement lors d’une rencontre sportive. Dans le pays, une grande partie de la population est intéressée par le pari sportif, notamment les chômeurs. Certains considèrent le pari sportif comme un travail pour nourrir la famille.

Actuellement, pour effectuer un pari sportif, le joueur doit se rendre à un cash point. Dans un cash point, les listes des matchs du jour avec les différentes cotes sont affichées sur un écran. Le joueur choisie le ou les matchs sur qui il veut parier et prend note sur un bout de papier. L’agent du cash point l’accueille ensuite, reçois l’argent misé sur le pari, enregistre les informations du pari sur une back office, et retourne un bout de papier ou ticket avec un code d’identification. Après visualisation des résultats des matchs sur son ticket, le joueur retourne vers le cash point et présente son ticket à l’agent. L’agent check le ticket grâce au code du ticket, si le ticket est gagnant alors l’agent paye la somme gagnée au joueur.

Le système de pari sportif est n’est pas très pratique car, les joueurs ont l’obligation de se déplacer vers un cash point et les paiements ne sont pas automatiques.

## **Solution envisagée**

En guise de solution envisagée, le pari sportif doit être disponible en ligne. Il est aussi favorable de créer des applications version mobile et desktop afin que tout le monde puisse être à l’aise avec l’appareil de son choix. Créer un back office pour faciliter les taches des admins.

# **DEMARCHE PROJET**

## **Principes de la démarche projet**

### Activités d’ingénierie logicielle

La première méthode applicable est la méthode Agile, qui divise le projet en plusieurs itérations appelées « sprint ». Chaque objectif est déterminé à chaque début de sprint, les tâches sont décrites avec ordre de priorité que les développeurs peuvent suivre ou non selon les circonstances de codage. Le planning du sprint peut ne pas être suivi à la lettre, le but est de finir les tâches à la fin du sprint.

La deuxième méthode applicable est la méthode en cascade qui consiste à déterminer les tâches dès le début et de mettre en place la conception après, puis le développement. Ensuite il y a la phase de test, de déploiement et de maintenance.

### Démarche projet adoptée

Pour la gestion de notre projet, nous avons choisi la méthode itérative. Pour chaque itération (séquence d'[instructions](http://www.dicofr.com/cgi-bin/n.pl/dicofr/definition/20010101002601) destinée à être exécutée plusieurs fois), nous faisons la conception, le développement et les tests de validation du lot à livrer.

Chaque semaine, une réunion est tenue au sein de l’équipe pour connaître l’avancement du projet.

### Rôles et responsabilités

Notre équipe est composé de 3 personnes qui sont : Elliot, Toavina et Mahefa. Nous avons tous contribués sur la conception des différents projets. Ci-dessous le dispatch sur les projets effectués :

* Elliot :
  + Création du serveur Node
  + Création de l’application client web
  + Création de l’application winform exécutable
* Toavina :
  + Création du serveur Grails
  + Création du back office
* Mahefa :
  + Création de l’application mobile

### Outils

Pour la réalisation de ce projet, nous avons utilisés les outils suivants :

* Power amc : pour la conception de la base de données
* Sqldevelopper : pour manipulation de la base de données oracle
* Sourcetree : pour la gestion du repository Git
* IntelliJ IDE : pour le développement de la partie Back End en Grails
* WebStorm : pour le développement Angular du back Office
* Visual Studio Code : pour le développement de la partie Back End Node js et du client web (Angular)
* Visual Studio : pour le développement du client exécutable (WinForm)
* Android Studio : pour le développement de l’application mobile

## **Contrainte et risques**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N° des risques | Libellé du risque | Facteur contribuant | Solution proposée | Statut |
| 1 | Hébergement d’une base de données oracle | Cout d’hébergement élevé | Utiliser un serveur Oracle fourni par l’université | Résolu |
| 2 | Service Grails fréquemment indisponible | Crash récurrent de l’application Grails déployé sur Heroku | Faire des recherches de débogage sur des forums et des tests monitoring | En cours |
| 3 | Déploiement sur Heroku | Les 6 projets sont combinés dans les branches sur GIT | Séparer les projets par branche | Résolu |

Figure 2 - Tableau des contraites et risques

## **Démarche projet mise en œuvre**

* Création du document de cadrage
* Estimation de charges
* Document technique
* Développement
* Déploiement

## **Planification**

Le développement des projets était palanifié en fonction de nos temps.

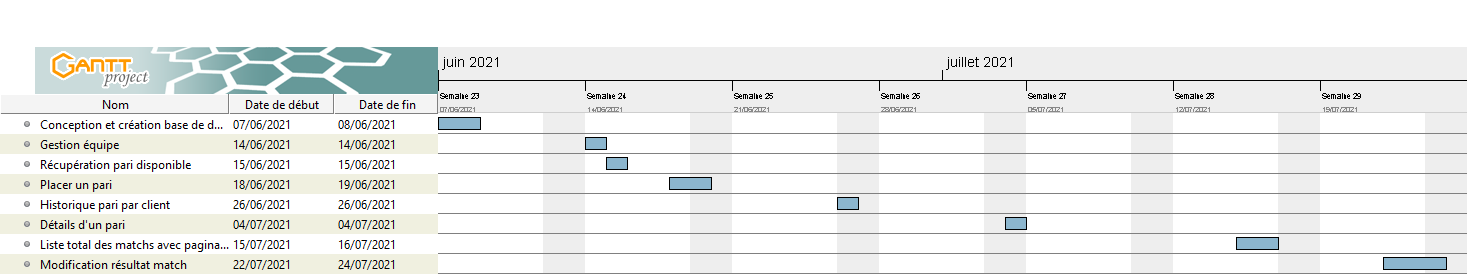


Figure 3 – Planification développement serveur Grails

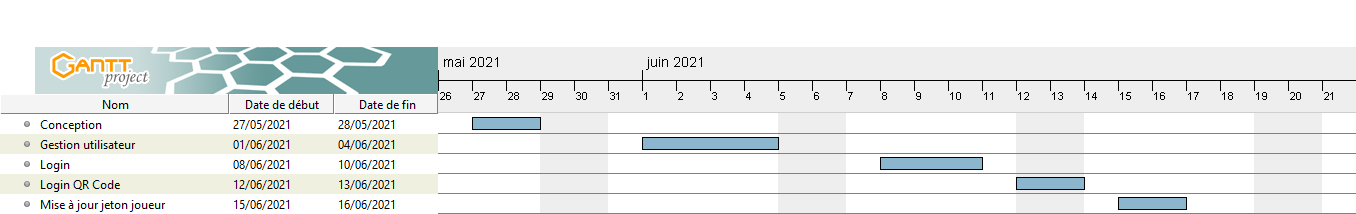


Figure 4 – Planification développement serveur node

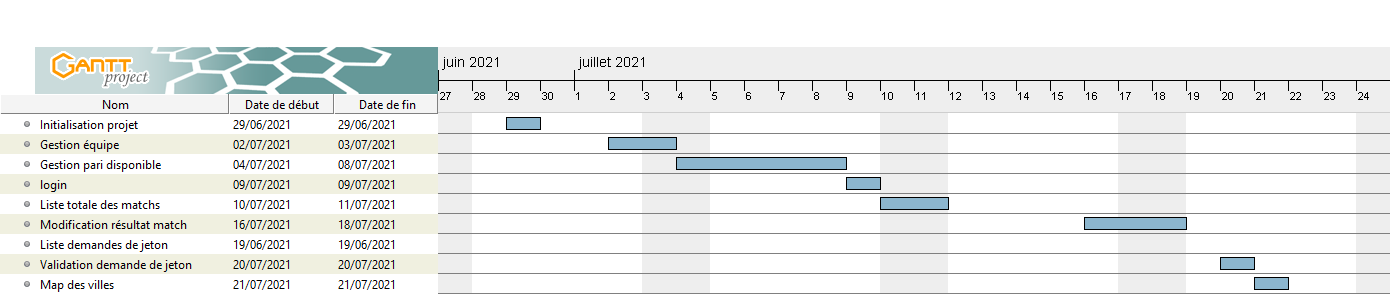


Figure 5 - Planification développement du back office

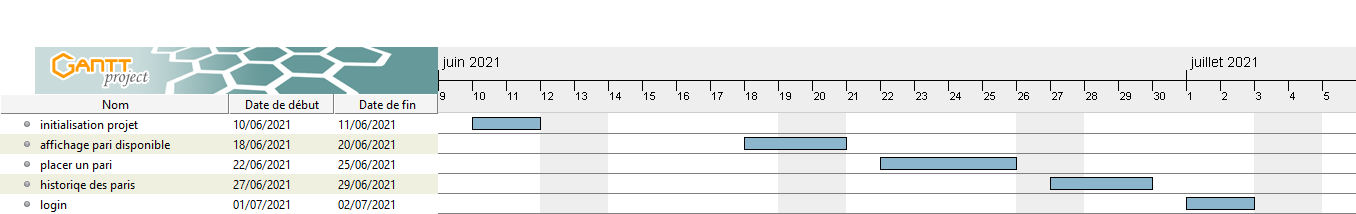


Figure 6 - Planification développement du client web

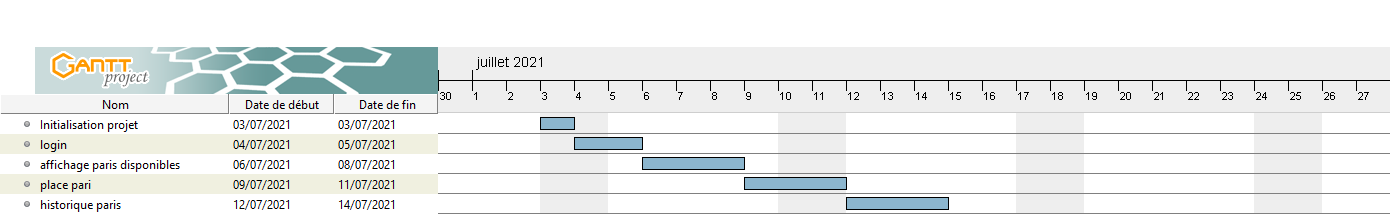


Figure 7 - Planification développement du client lourd

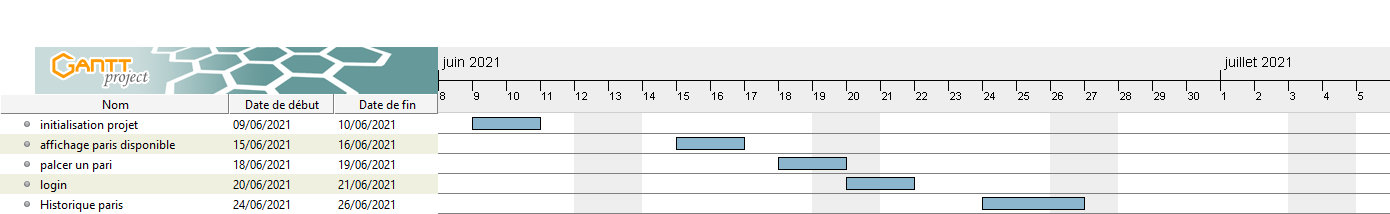


Figure 8 - Planification développement de l'application mobile

# **EXIGENCES REALISEES DANS LE PROJET**

## **Exigences fonctionnelles - Cas d’utilisation**

### Cas d’utilisation pour le back office

L’administrateur peut :

* S’authentifier à l’application
* Gérer les équipes
* Gérer les paris disponibles
* Visualiser la totalité des matchs
* Valider les demandes de Jetons

### Cas d’utilisation pour les clients web, exécutable et mobile

Le client ou joueur peut :

* S’authentifier via un formulaire
* S’authentifier via un QR Code généré par le client web sur le client mobile
* Effectuer une demande de jetons
* Parier sur des matchs
* Voir l’historique des paris effectués

## **Exigences non fonctionnelles transverses**

* **Utilisabilité**

Tout type d’ordinateur avec un système d’exploitation windows peut exécuter le client lourd.

* **Robustesse**

La panne de serveur Grails dépend de Heroku le site de déploiement et la liaison vers la base de données Oracle distant.

* **Performance**

La configuration requise pour l’application mobile est une version d’android plus de 6.0

## **Interfaces détaillées**

### IHM

### Back Office :

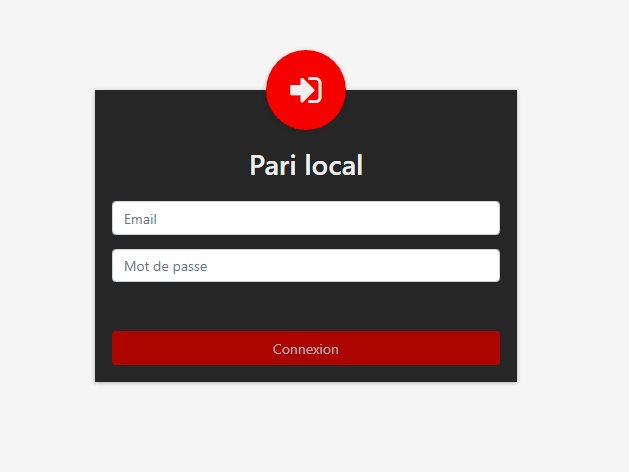


Figure 9 - Login back office

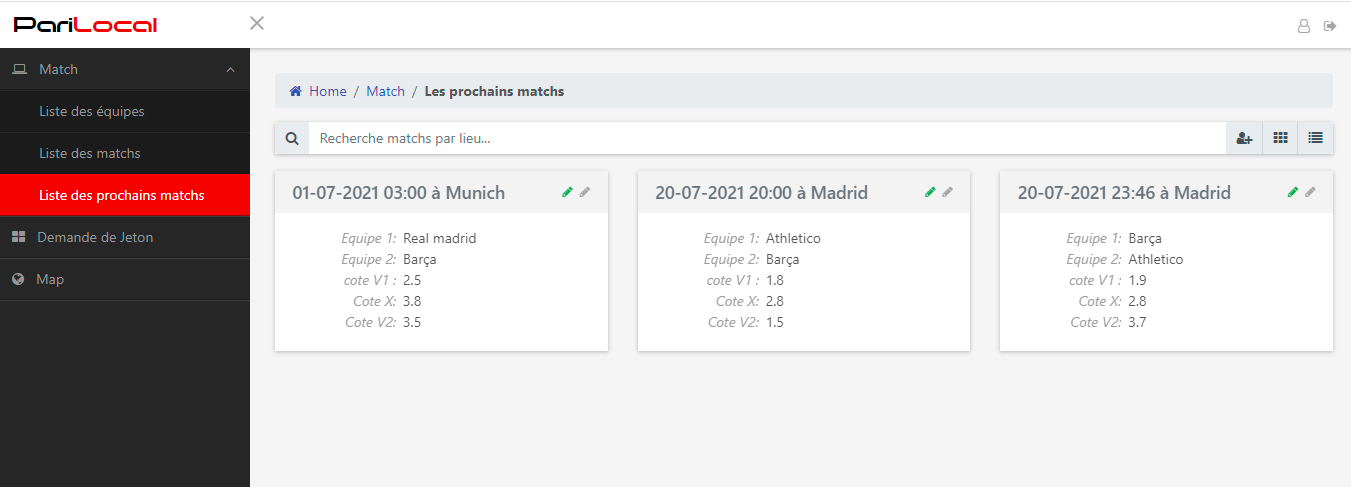


Figure 10 - gestion paris disponible

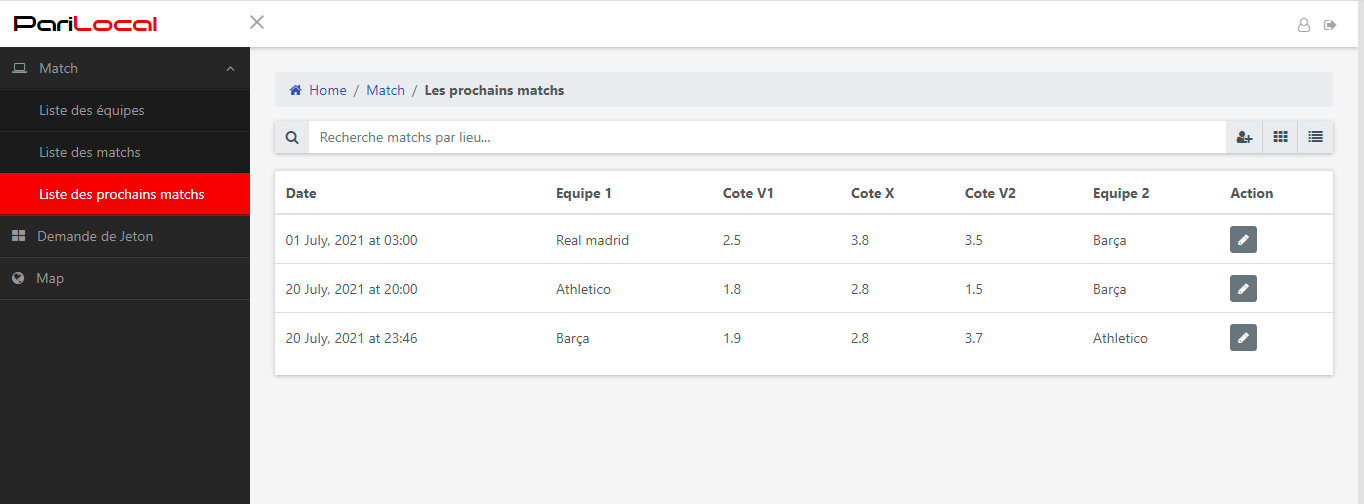


Figure 11 - Gestion paris disponibles sous forme d'un tableau

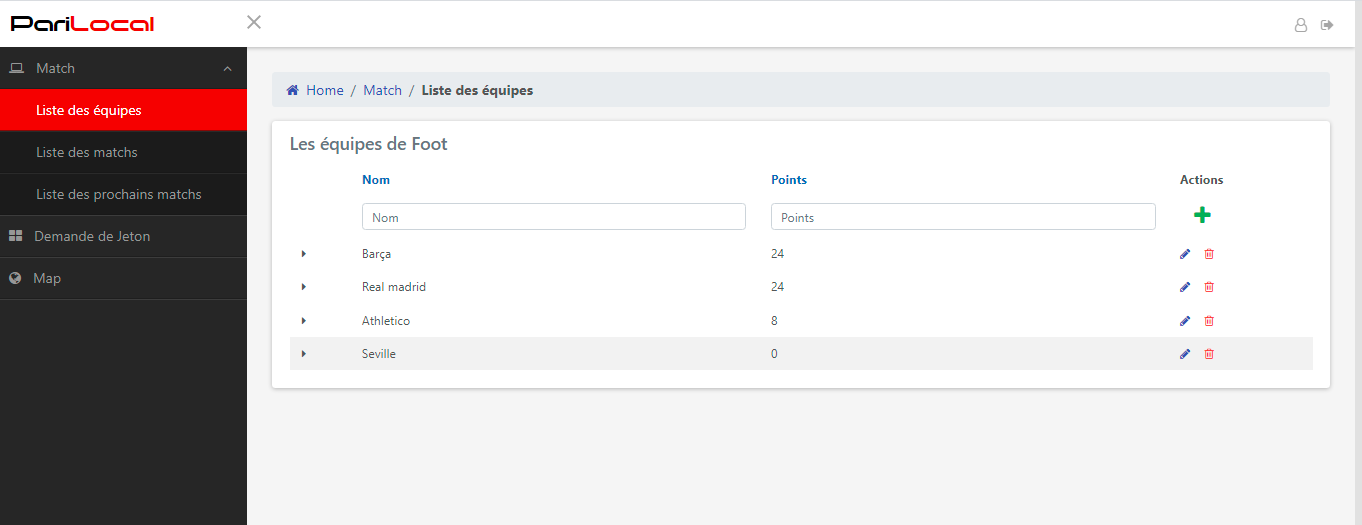


Figure 12 - Gestion des équipes

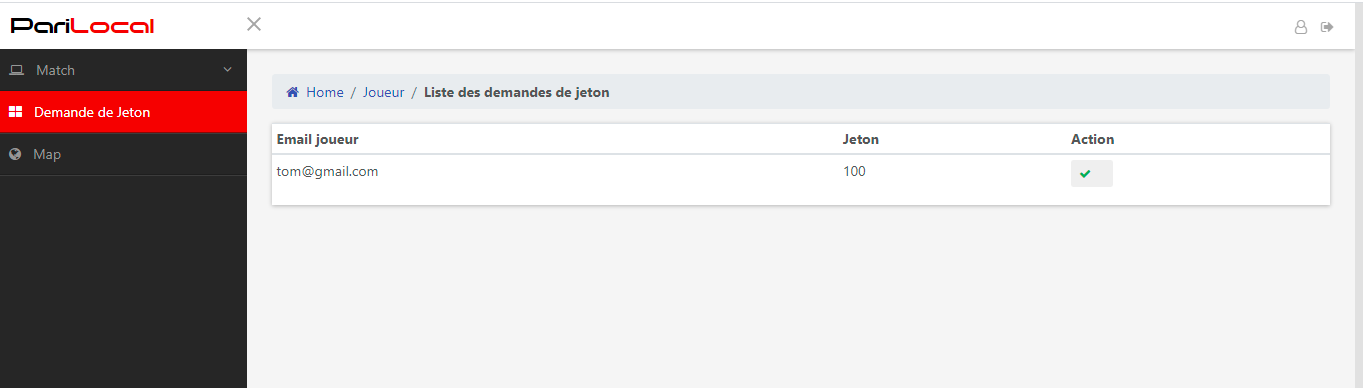


Figure 13 - Liste demandes de jeton

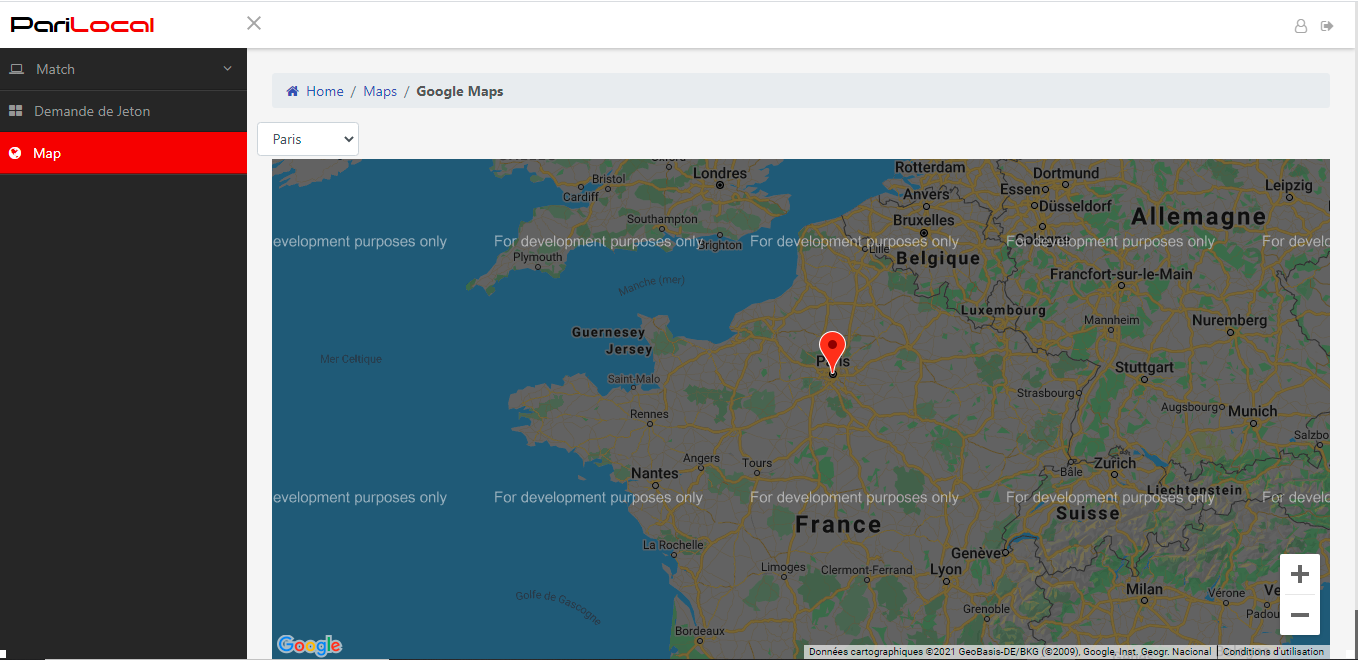


Figure 14 - Map

### Client web :

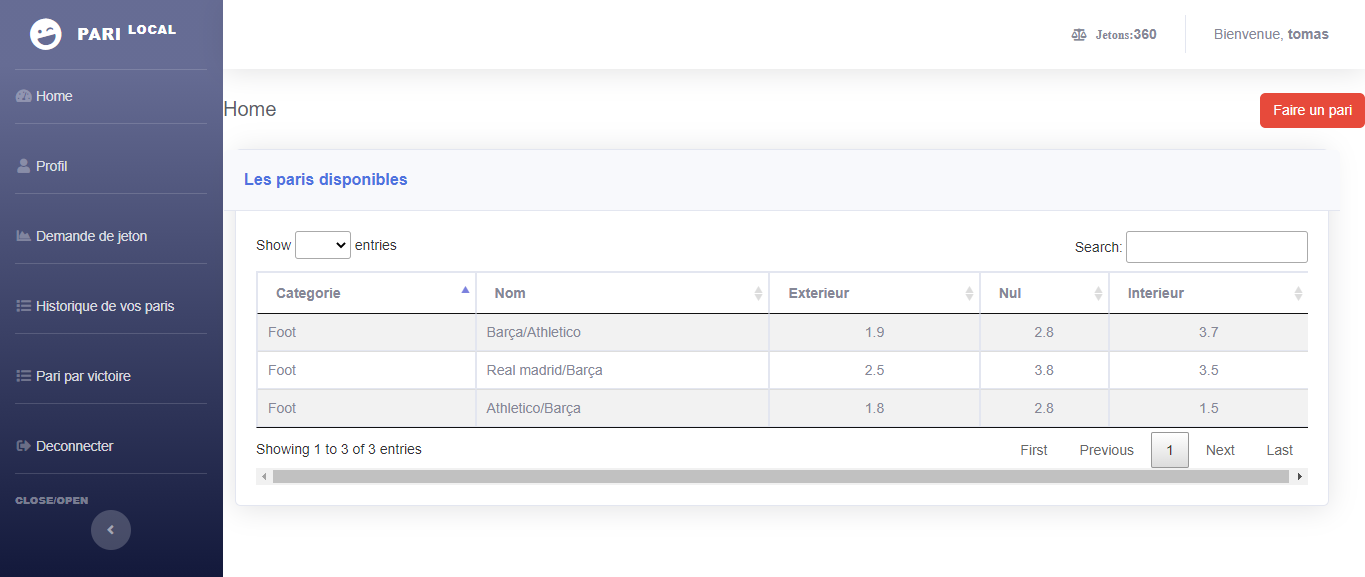


Figure 15 - Client web liste des paris disponibles

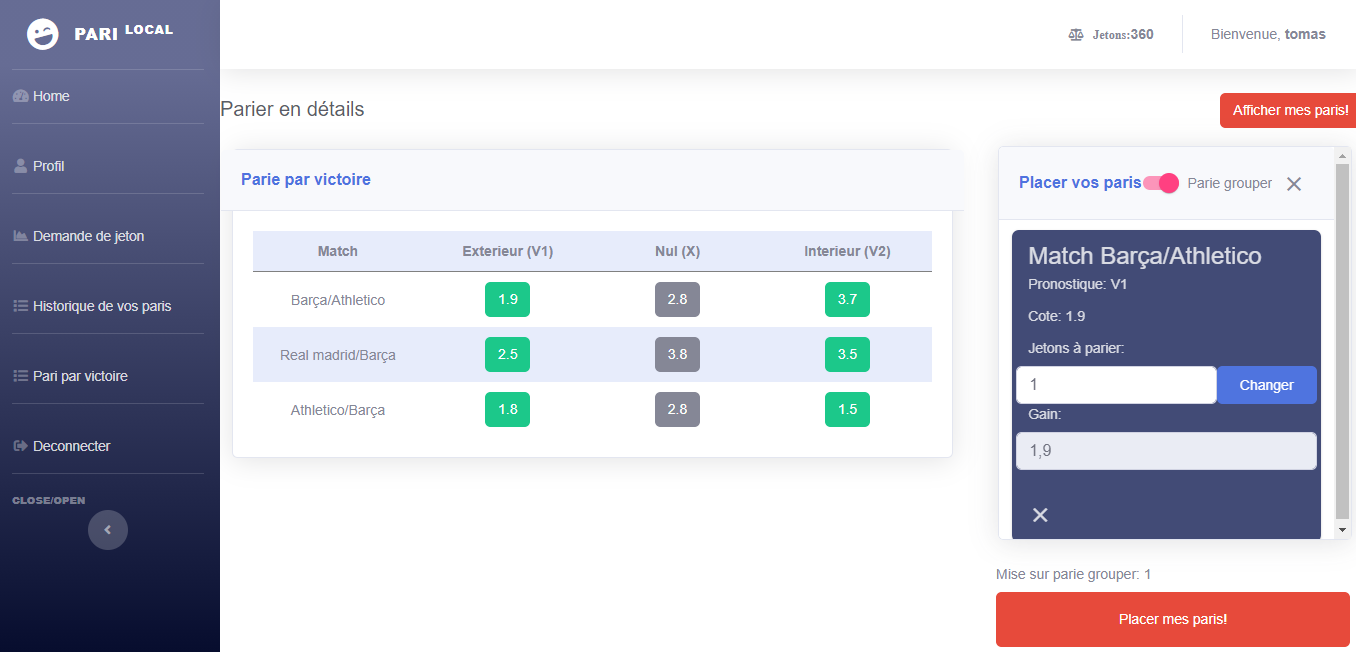


Figure 16 - Client web palcer un pari

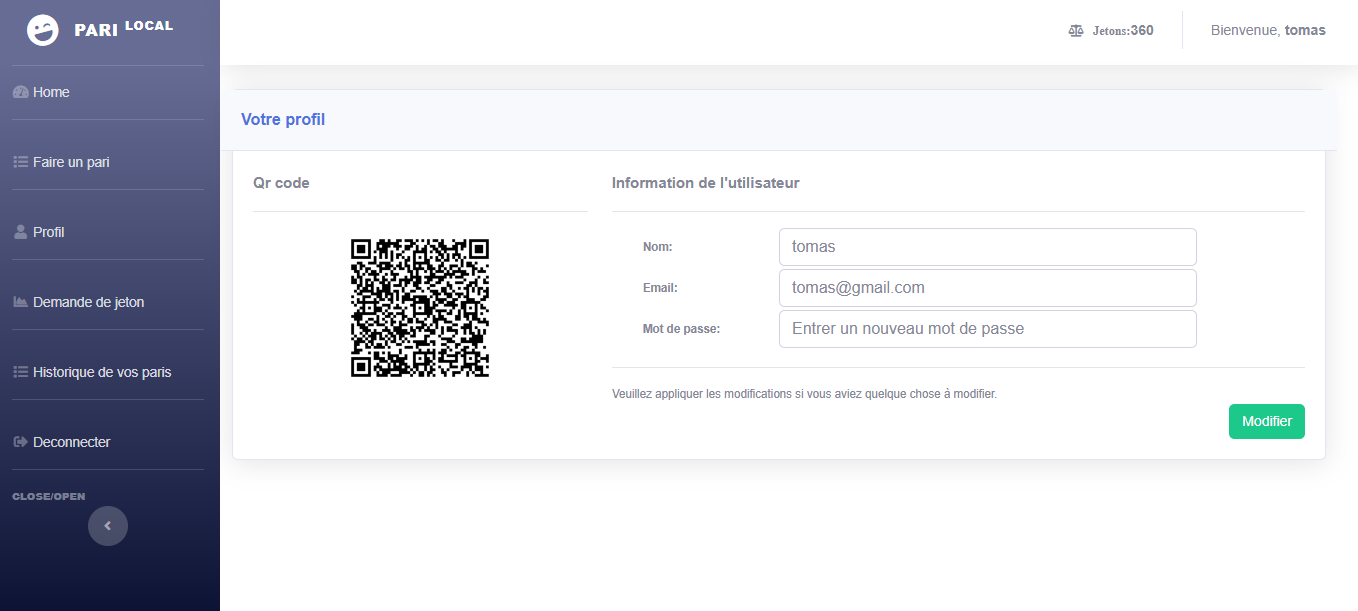


Figure 17 - Client web information de l'utilisateur



Figure 18 - Client web faire une demande de jeton

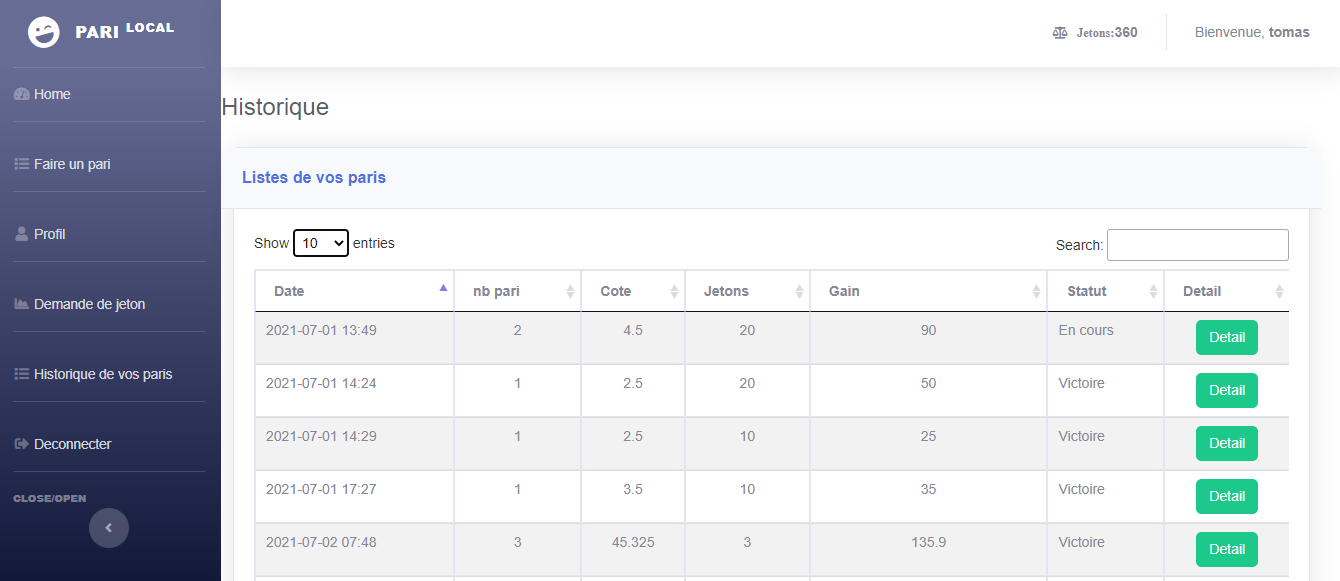


Figure 19 - Client web historique des paris

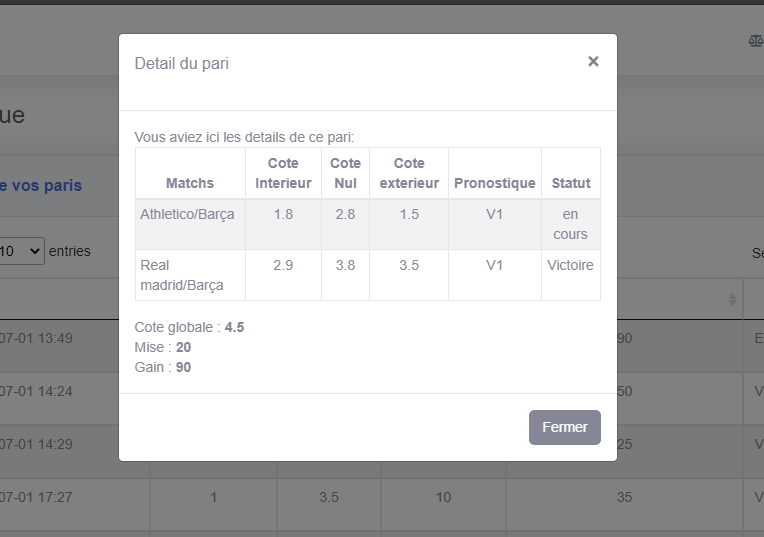


Figure 20 - Client web détail d'un pari

### Client lourd :

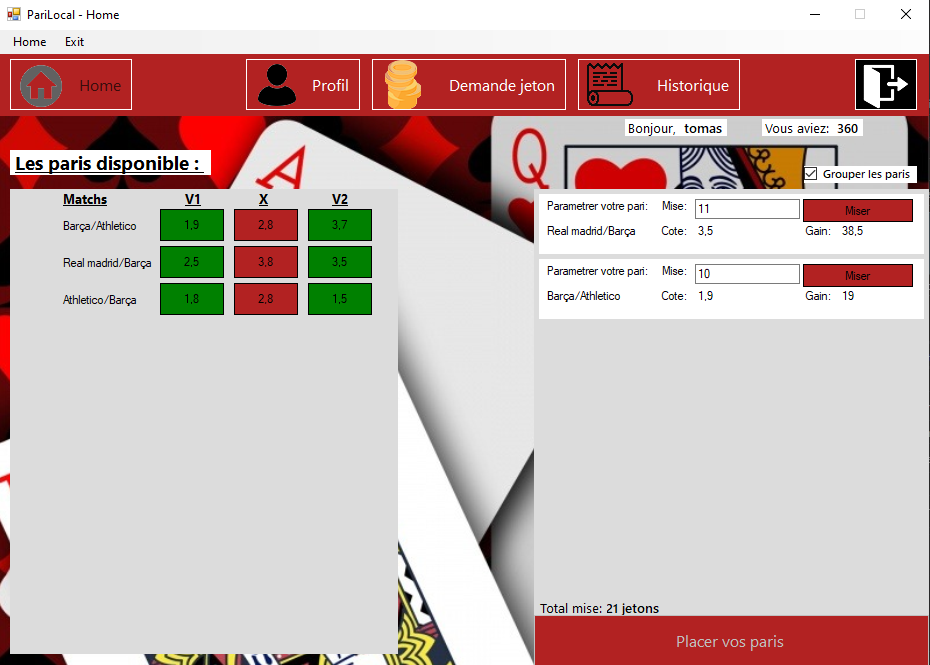


Figure 21 - Client lourd paris disponibles

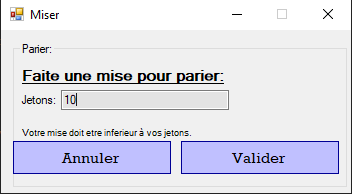


Figure 22 - Client lourd parier

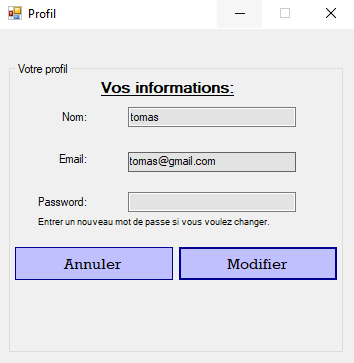


Figure 23 - Client lourd modification information utilisateur

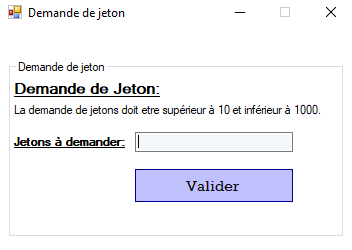


Figure 24 - Client lourd faire une demande de Jeton

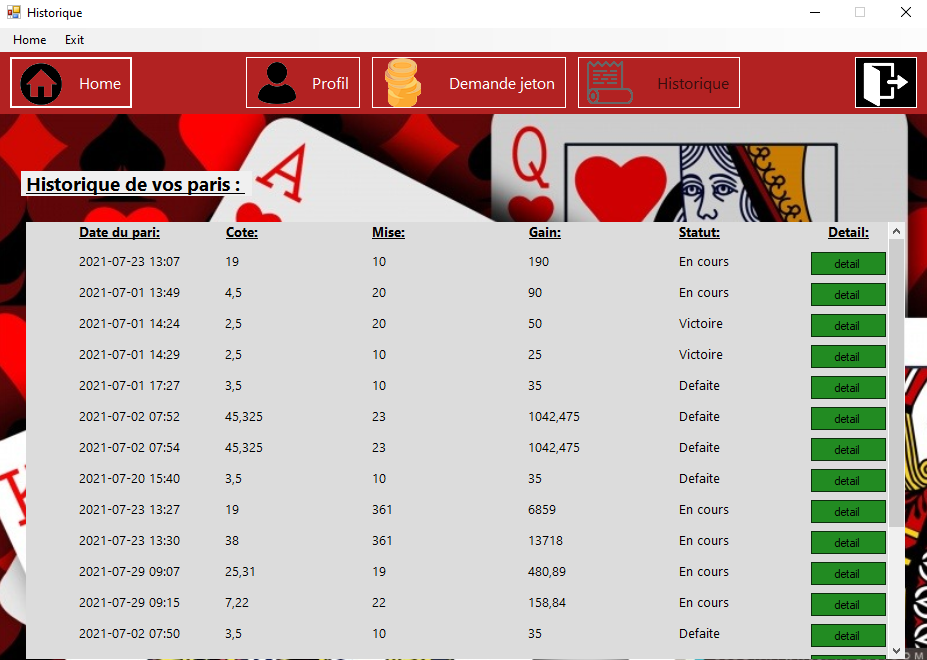


Figure 25 - Client lourd historique des paris



Figure 26 - Client lourd détails d'un pari

### Application mobile :

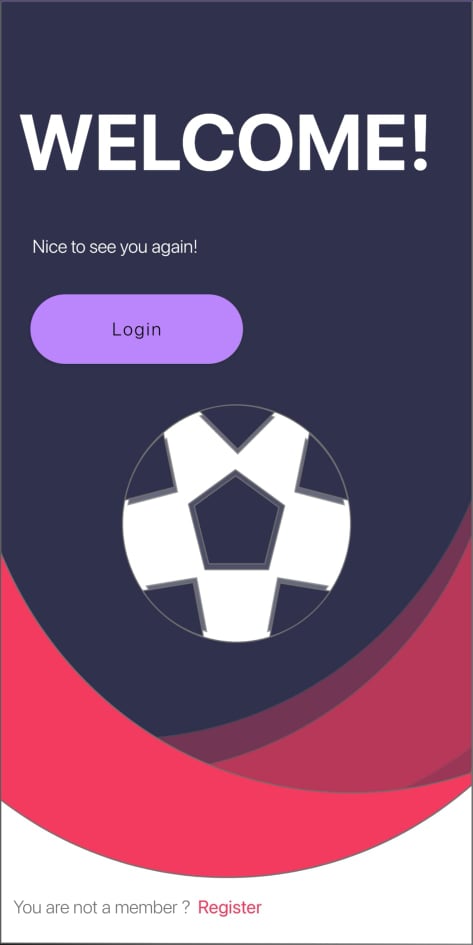


Figure 27 - Client mobile accueil

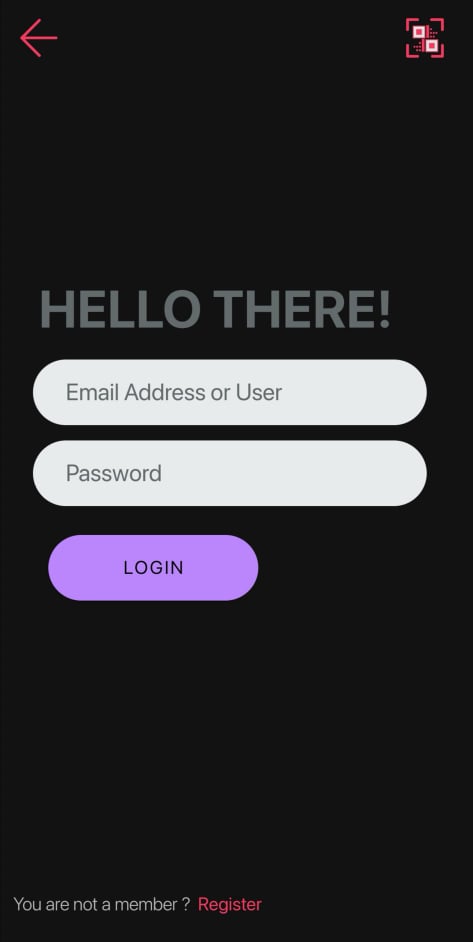


Figure 28 - Client mobile login

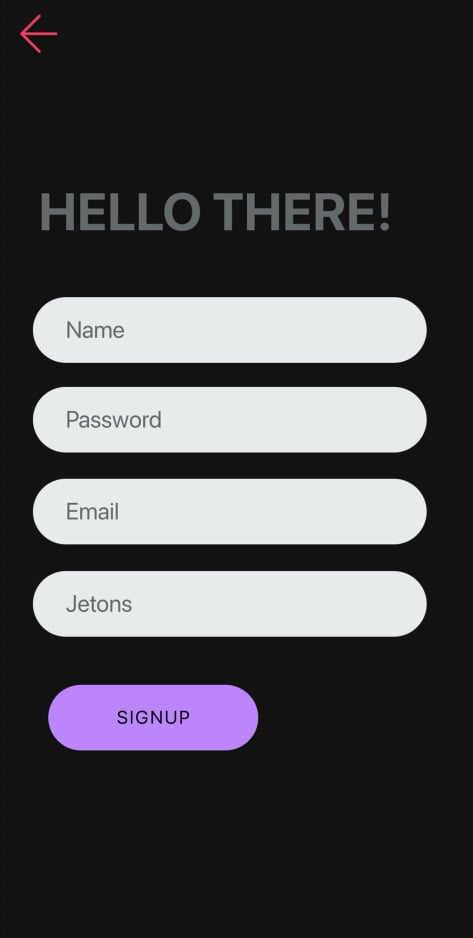


Figure 29 - Client mobile inscription

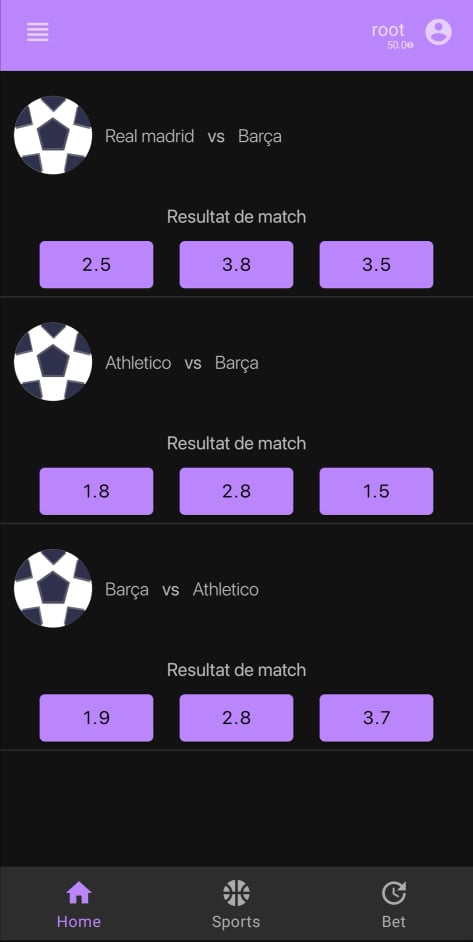


Figure 30 - Client mobile paris disponibles

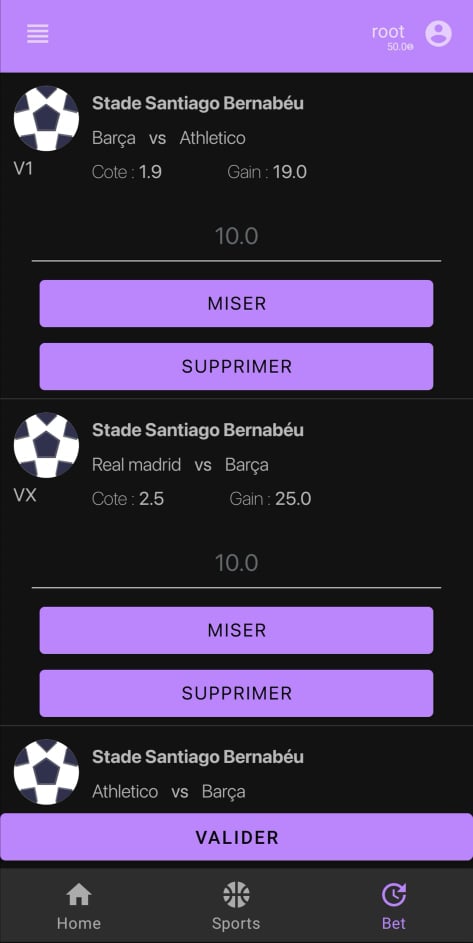


Figure 31 - Client mobile saisie pari

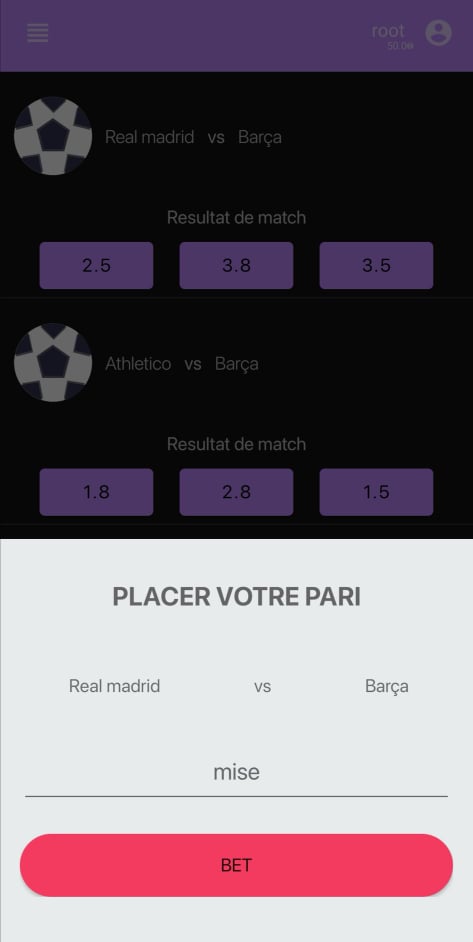


Figure 32 - Client mobile placer un pari



Figure 33 - Client mobile faire une demande de jeton

# **CONCEPTION DU SYSTEME LOGICIEL REALISE DANS LE PROJET**

## **Plate-forme technique**

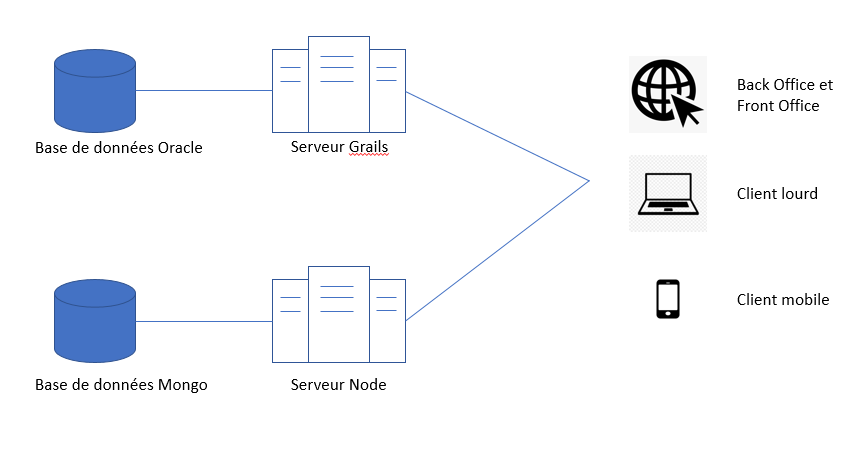


Figure 34 - Architecture

Le serveur Grails est lié à la base de données Oracle et le serveur Node est lié à la base de données Mongo. Le serveur Grails gère les paris et l’historique des joueurs. Le serveur Node gère les informations des joueurs : l’accès aux applications et les jetons.

Pour les frontends, le back-office est une application web. Les clients ont la possibilité de se connecter via application web, client lourd ou client mobile.

## **Conception du logiciel développé**

### Conception du code source

La structure d’un projet Grails est déjà en place automatiquement Les domains, service, et controller sont déjà séparés. Les fichiers groovy sont placés sous les packages itu.mbds.tpt.

Pour le projet Back Office Angular, les modeles sont situés dans le dossier « /modele/ » et les services dans « /service/ ». Un module principal « PageModule » a été généré pour stocker les modules :

* « MatchModule » : pour les vues concernant la gestion des matchs
* « JoueurModule » : pour les vues concernant les joueurs : DemandeJetonComponent

Les projets a été envoyés sur un repository GIT. Les projets Grails, ServeurNode, Admin, Angular client, Winform, Mobile ont chacun un branch sur le repository GIT.

#### **Le code source - vue statique**

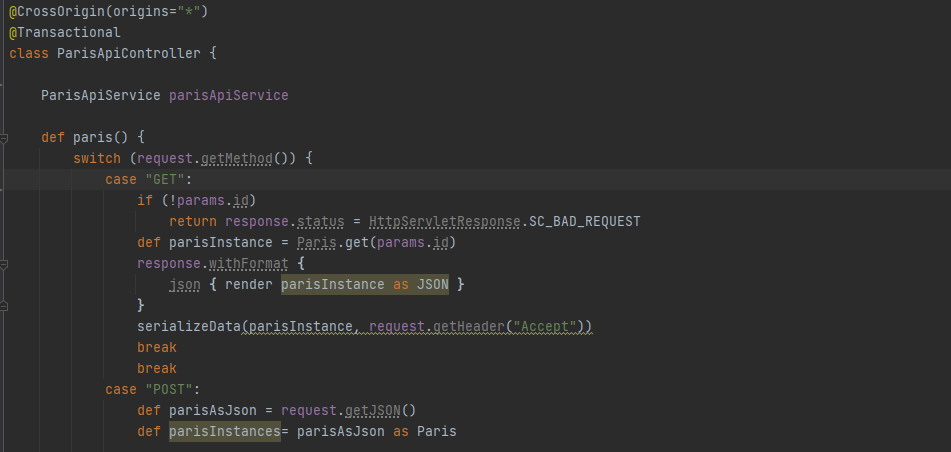


Figure 35 - Exemple code Grails

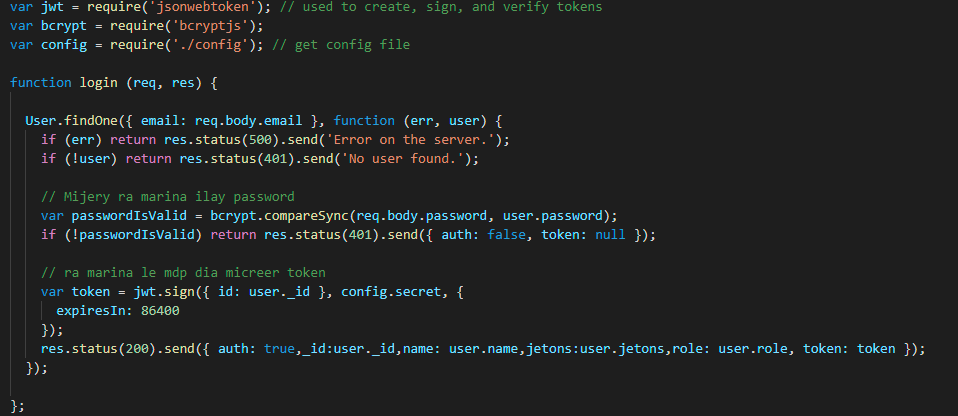


Figure 36 - Exemple code Node

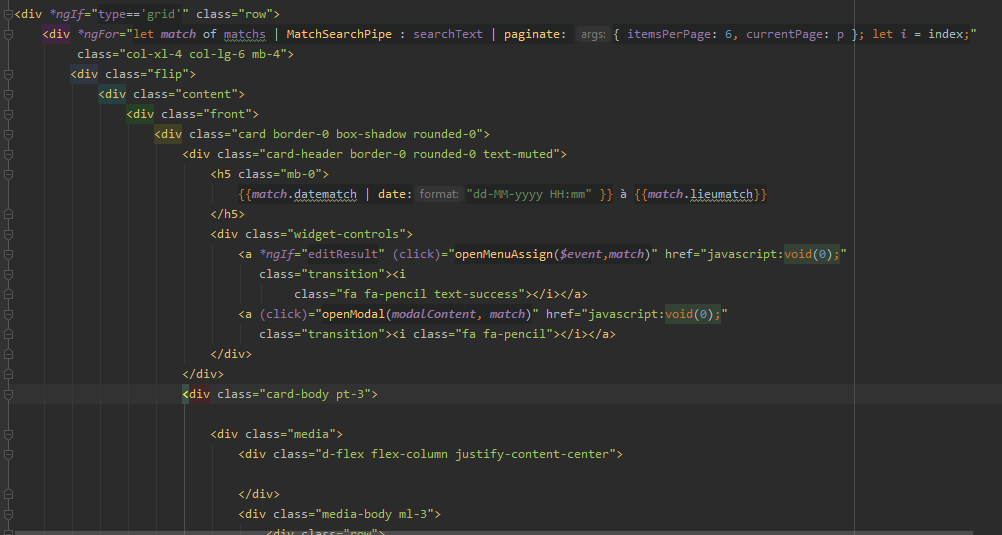


Figure 37 - Exemple code vue Back Office

#### **Modélisation de données**

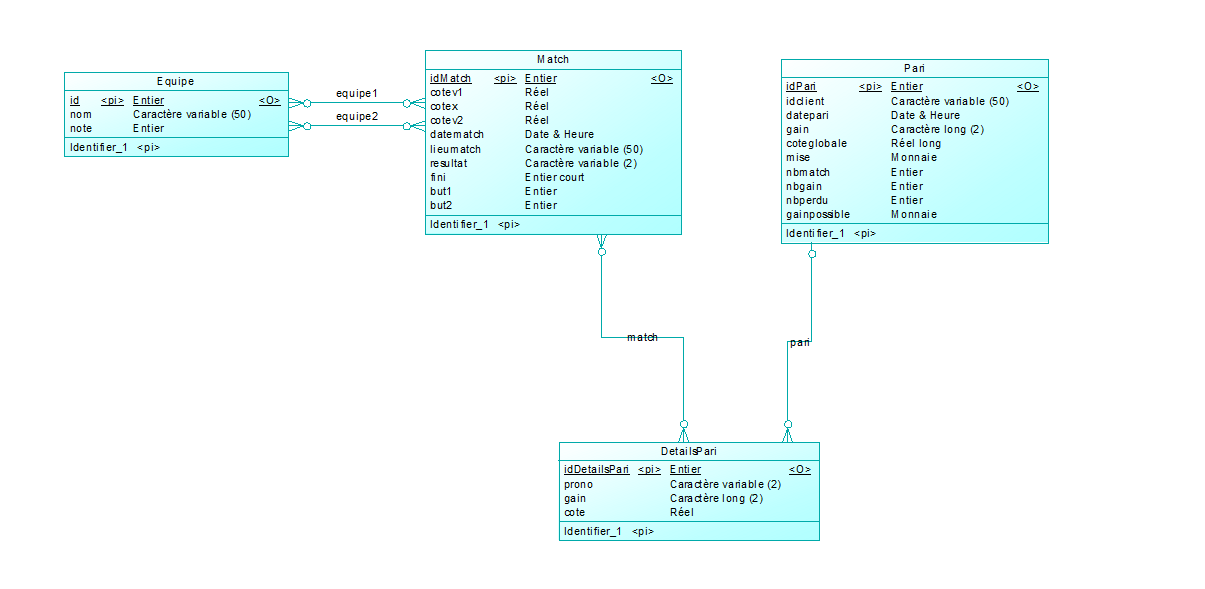


Figure 38 - Model conceptuel de données

Sur la base de données Oracle, nous avons les tables :

* Equipe
* Match
* Pari
* DetailPari

Pour l’optimisation des requêtes, nous avons les indexs sur :

* Nom (table Equipe)
* Equipe1\_id (table Match)
* Equipe2\_id (table Match)
* Idclient (table Pari)

#### **Réalisation des cas d’utilisation**

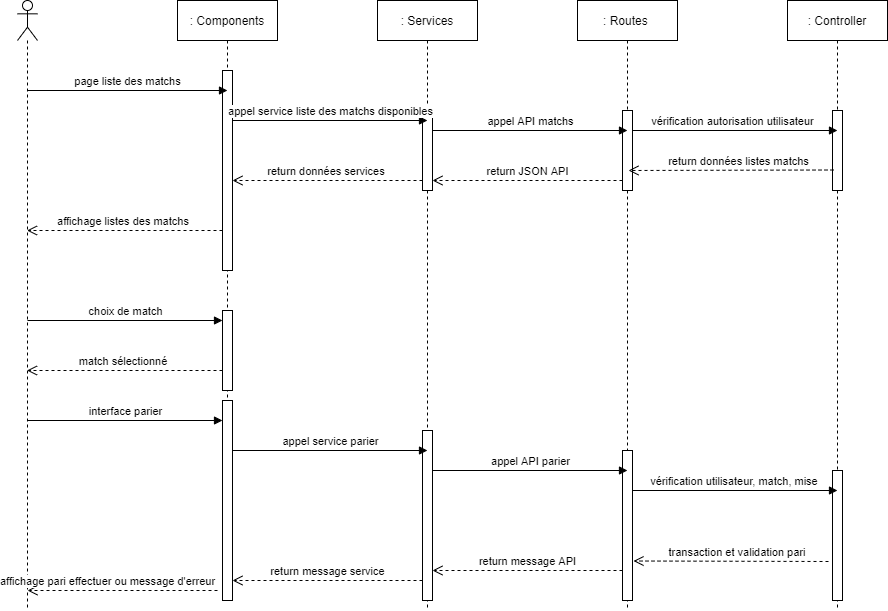


Figure 39 - CU Parier

**Acteur** : Joueur.

**Objectif** : Placer un pari

**Scénario** :

* l’acteur navigue sur l’affichage des paris disponibles
* l’acteur fait son choix
* l’acteur place son pari

### Les composants et leur déploiement

L’API Grails :

Serveur base de données : Oracle en ligne fourni par l’université.

Déploiement : sur le site Heroku

L’API Node :

Serveur base de données : Mongo Atlas

Déploiement : sur le site Heroku

Le back office :

Déploiement : sur le site Heroku

Le client web :

Déploiement : sur le site Heroku

L’application mobile :

Fichier de livraison : fichier .apk

Le client lourd :

Fichier de livraison : fichier .exe

# **Tests du système logiciel**

|  |  |  |  |
| --- | --- | --- | --- |
| Fonctionnalité testé | Données utilisées | Résultat attendu | Résultat obtenu |
| Placer un pari | Matchs :  Real de Madrid vs Barcelone  Prono : V2, cote : 2,5  Psg vs Athletico  Prono : V1, cote : 1,5  Mise : 15 jetons | Cote totale : 3,75  Gain possible : 56,25 | Cote totale : 3,75  Gain possible : 56,25 |
| Donner résultat match | Match :  Real de Madrid vs Barcelone  Score : 3 -1 (V1) | Pari précédent perdu ou KO | Pari précédent KO |

Figure 40 - Tableau de test

# **CONCLUSION**

Le but de ce stage est de mettre en place un système informatique pour faciliter le pari sportif. Le sport qui a été choisi est le football qui est le sport le plus regardé dans le monde

Lors de ce travail de groupe, nous avons rencontré des difficultés notamment sur la manipulation de la base de données Oracle, le déploiement du serveur Grails sur le site Heroku, le développement de l’interface winform. Pour pallier à ces difficultés, nous avons approfondi les recherches des tutoriels et de débogage.

Ce TP Transversal nous a ainsi permis de mettre en pratique les connaissances théoriques acquises lors de notre formation académique à l’IT University Andoharanofotsy et d’en acquérir de nouvelles relatives à l’utilisation de techniques de création de web service, d’application web, lourd et mobile. Cette expérience à mis en évidence des points positifs.

Le bilan de ce projet est dans l’ensemble positif. La plus grande partie du travail demandée a été réalisée.

Comme toute application, Que fera l’objet d’amélioration et de modification.

Enfin, ce TP Transversal constitue est une expérience valorisante et enrichissante pour notre avenir professionnel.

# **REFERENCES ET BIBLIOGRAPHIE**

Documentation officielle du framework Angular :

<https://angular.io/>

Documentation winform :

<https://docs.microsoft.com/en-us/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2019>

Site de Heroku :

<https://www.heroku.com/>