DOCUMENT DE CONCEPTION

Projet TPT

Par

Rakotohanitriaina Maharo Rivomahefa

Ranarivelo Voahangy Antsa

Lala Thong Sang Irchad

Table des matières

[1. Conception 4](#_Toc79956818)

[1.1. Contraintes techniques 4](#_Toc79956819)

[1.2. Plateforme technique 4](#_Toc79956820)

[1.2.1. Architecture logicielle 4](#_Toc79956821)

[1.3. Conception du logiciel développé 5](#_Toc79956822)

[1.3.1. Conception du code source 5](#_Toc79956823)

[1.3.1.1. Le code source – vue statique – web 5](#_Toc79956824)

[1.3.1.2. Le code source – vue statique – mobile 5](#_Toc79956825)

[1.3.1.3. Le code source – vue statique – Application de bureau 6](#_Toc79956826)

[1.3.1.4. Le code source – vue statique – API 7](#_Toc79956827)

[1.3.2. Modélisation des données 8](#_Toc79956828)

[1.3.2.1. Base oracle 8](#_Toc79956829)

[1.3.2.2. Base MongoDB 9](#_Toc79956830)

[1.3.3. Réalisation des cas d’utilisation 10](#_Toc79956831)

[1.3.4. Les composants et leur déploiement 15](#_Toc79956832)

[1.3.4.1. Application web 15](#_Toc79956833)

[1.3.4.2. Client lourd 15](#_Toc79956834)

[1.3.4.3. Application mobile 15](#_Toc79956835)

# Conception

# Contraintes techniques

**Langage de développement**

* Langage Java pour le serveur d’application
* Utilisation de Javascript côté serveur
* Utilisation de composant Angular pour la partie front
* Client lourd Winform

**Bases de données**

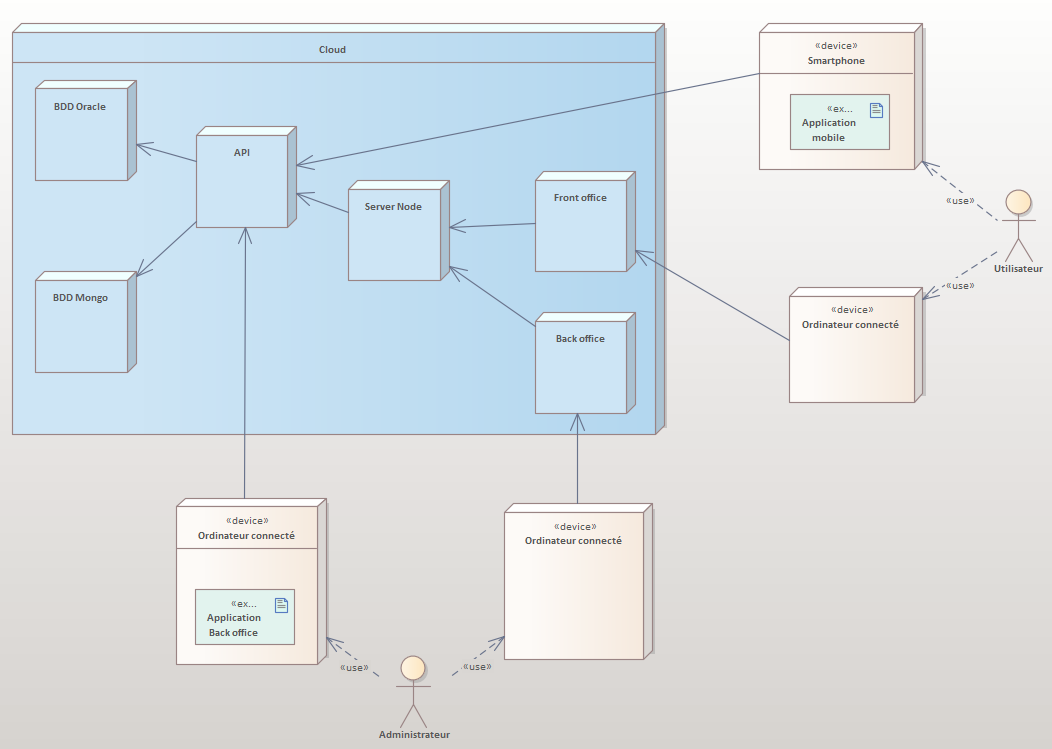
* Utilisation d’une Base Oracle
* Utilisation d’une base MONGODB
* Base de données locale sur le téléphone

**Modules**

* Utilisation des WebServices REST
* Implémentation d’un module mapping
* QRCode sur l’application mobile

# Plateforme technique

# Architecture logicielle



La base de données Orcale est hébergée sur le serveur Oracle Cloud. La base Mongo est une base en ligne appelée MongoDBAtlas. Le front-office, l’application back office web ,l’API et le serveur node sont hébergés sur la plateforme Heroku. Ainsi le front office et le back office communiquent tous deux avec le Server Node qui appellera l’API , intermédiaire entre les applications et les bases de données. Le back office client lourd et l’application mobile communiqueront directement avec l’API.

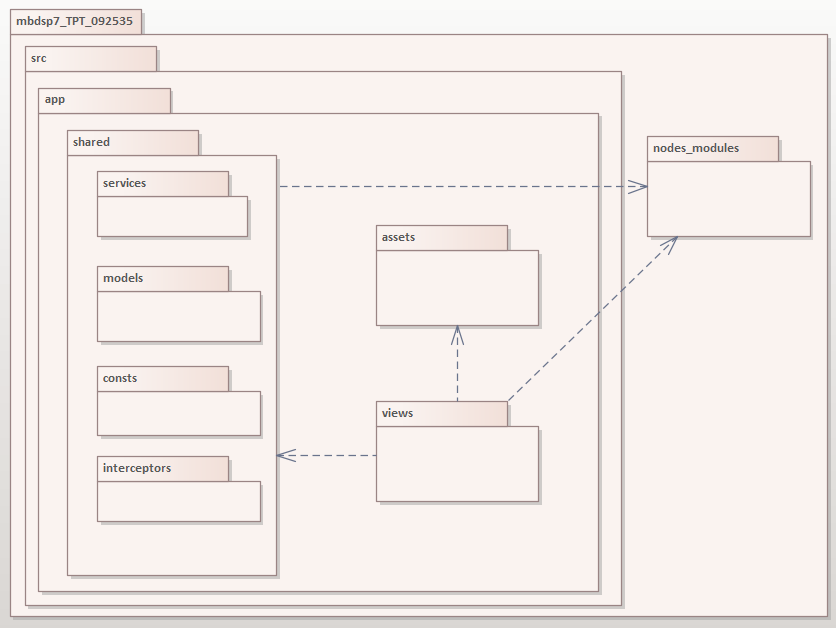
# Conception du logiciel développé

# Conception du code source

# Le code source – vue statique – web

L’architecture d’Angular se base sur la construction de l’application comme un ensemble de blocs fonctionnels appelés des composants Angular. Ces composants définissent des ***vues*** qui sont des ensembles d’éléments d’écran.

Les composants utilisent des services qui fournissent des fonctionnalités spécifiques. Les fournisseurs de services peuvent être injectés dans les composants en tant que dépendances, rendant ainsi le code modulaire, réutilisable et efficient.



# Le code source – vue statique – mobile

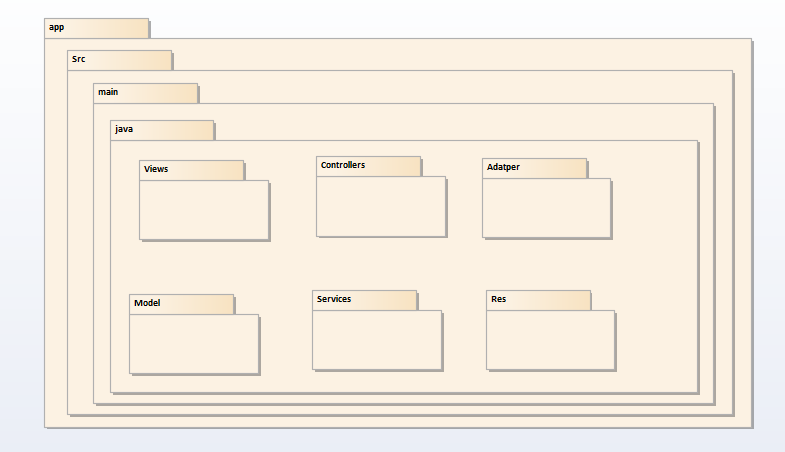
L’architecture de notre application mobile avec java suit le type MVC (Mode-View-Controller).

Dans la partie la partie Views il y a les différents pages qui seront utilisés sur le projet, il est composé des fragments et des activités.

Ensuite la partie Services est celle qui fait l’interprète entre l’API et l’application mobile, elle contient toutes les fonctions qui seront utilisés pour envoyer les requêtes et récupérer les données vers l’API.

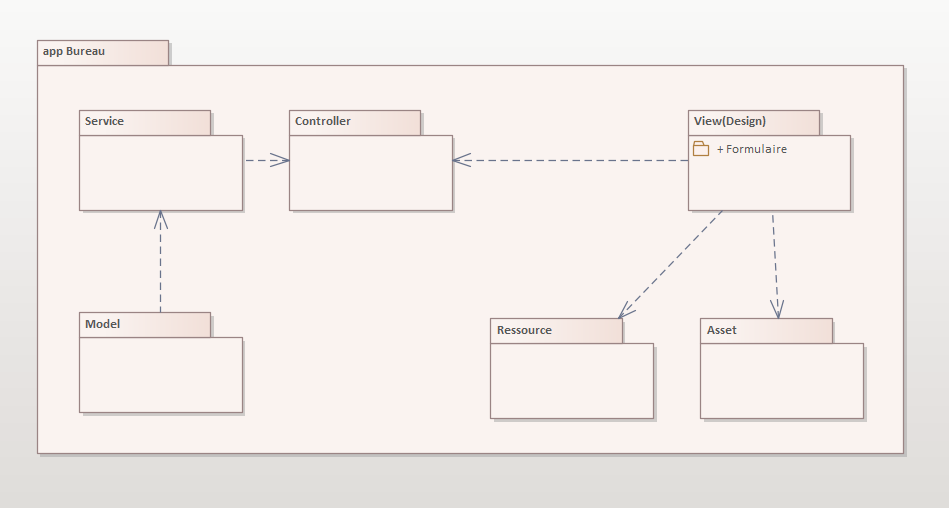
La partie Model est utilisé pour adapter les données venant de l’API en objets utilisables dans l’application.

C’est dans la partie Res qu’est contenu tous les ressources nécessaire et utilisés par l’application.



# Le code source – vue statique – Application de bureau

L’architecture de notre application bureau avec C# est du type MVC (Model – View – Controller). Nous pouvons complètement découpler nos couches applicatives métier et de présentation. De plus, nous aurons un objet complètement indépendant pour contrôler la couche de vue(view). L'indépendance entre les objets/couches de notre projet fourni par le MVC rendra la maintenance un peu plus facile et la réutilisation du code plus facile. Les vues sont contrôlées par les contrôleurs qui utilisent des API pour échanger avec la base. Les données recues sont traités par les modèles avant d’être renvoyées aux contrôleurs qui retourneront les vues pour l’affichage.



# Le code source – vue statique – API

L’API développé avec Spring boot suit une architecture en couches dans laquelle chaque couche communique avec la couche directement en dessous ou au-dessus (structure hiérarchique).

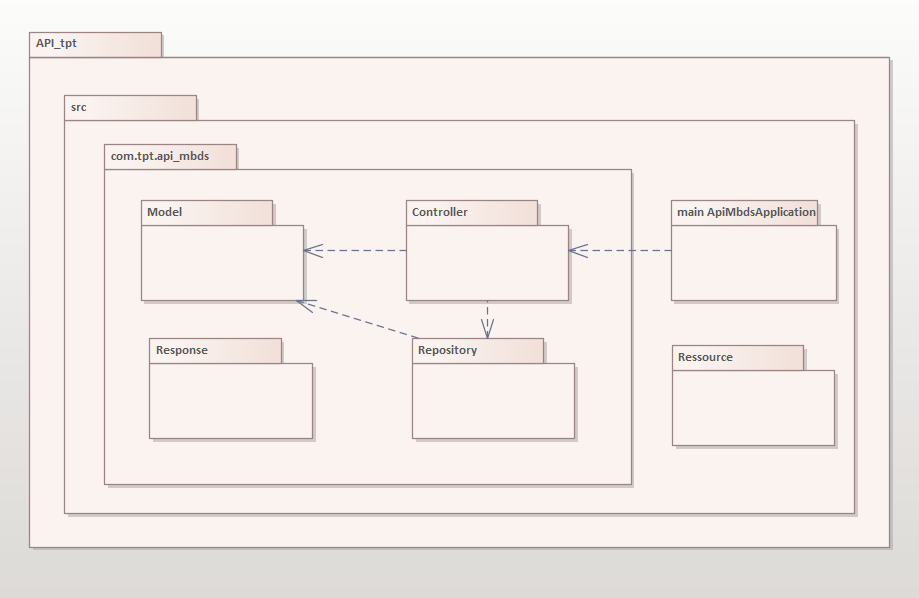
Ces couches sont (Couche de présentation, contrôleur, model et de base de données).

La couche de présentation gère les requêtes HTTP, traduit le paramètre JSON en objet, authentifie la requête et la transfère à la couche contrôleur.

La couche contrôleur : gère toute la logique métier. Il se compose de classes de services et utilise des services fournis par des couches d'accès aux données. Il effectue également l’autorisation et la validation.

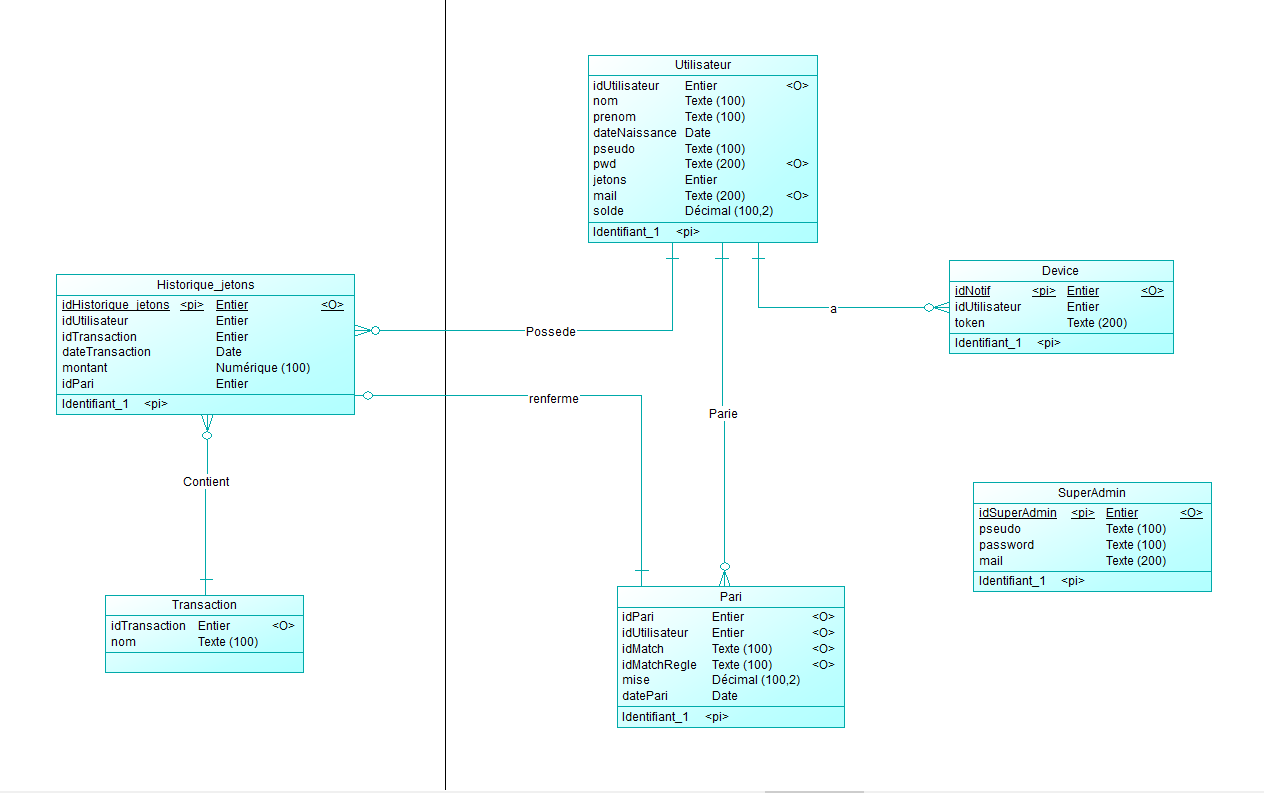
La couche model : traduit les objets venant du contrôleur vers les lignes de la base de données.

La couche de base de données : avec les « repositories » qui s’occupent des opérations CRUD (créer, récupérer, mettre à jour, supprimer) sont effectuées.

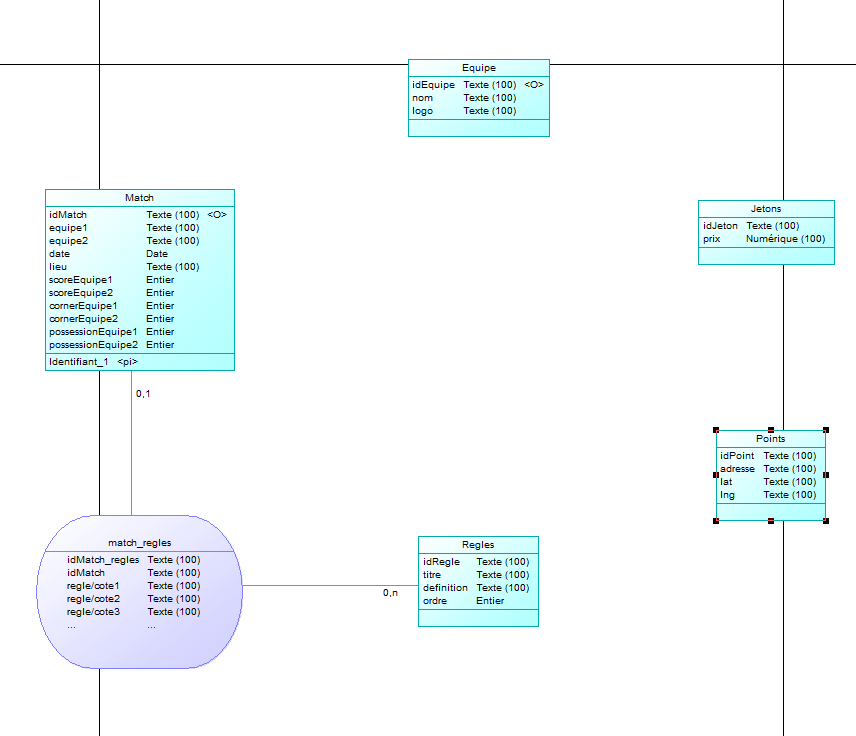


# Modélisation des données

# Base oracle



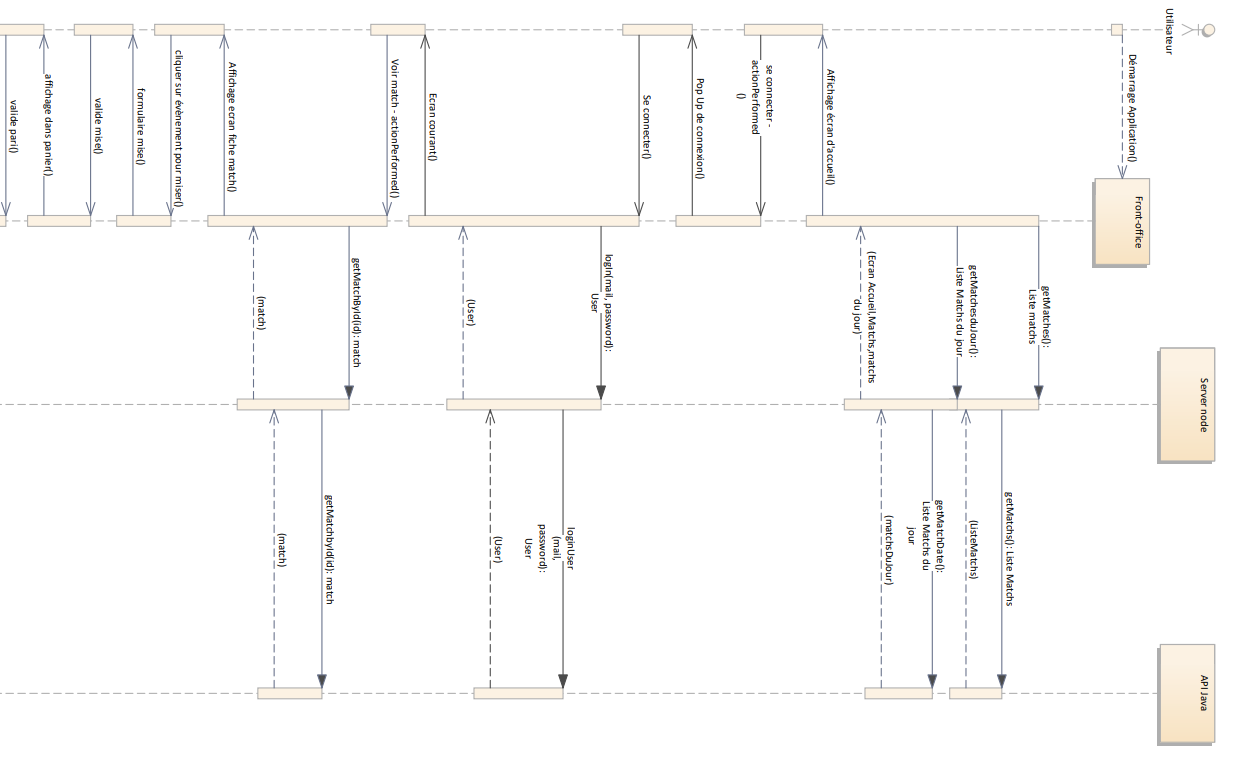
# Base MongoDB

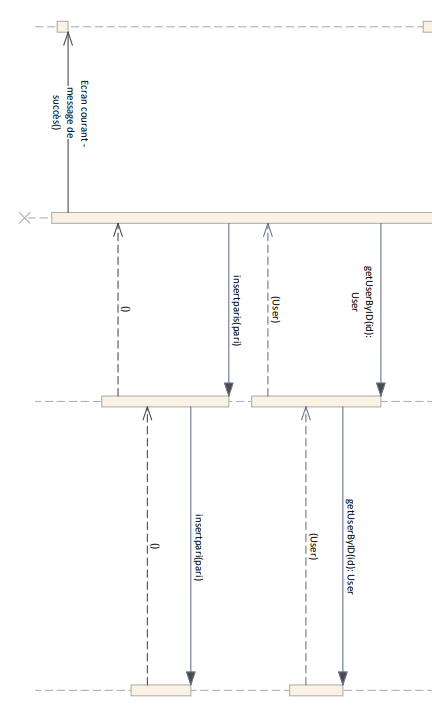


# Réalisation des cas d’utilisation

**Cas d’utilisation : Faire un pari sur Web**

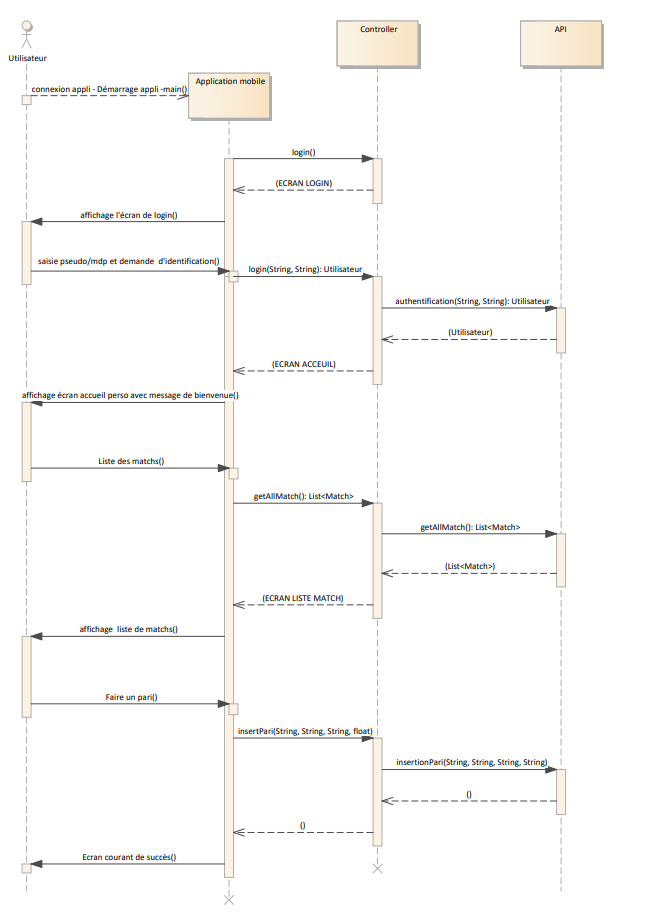
**Scénario normal**





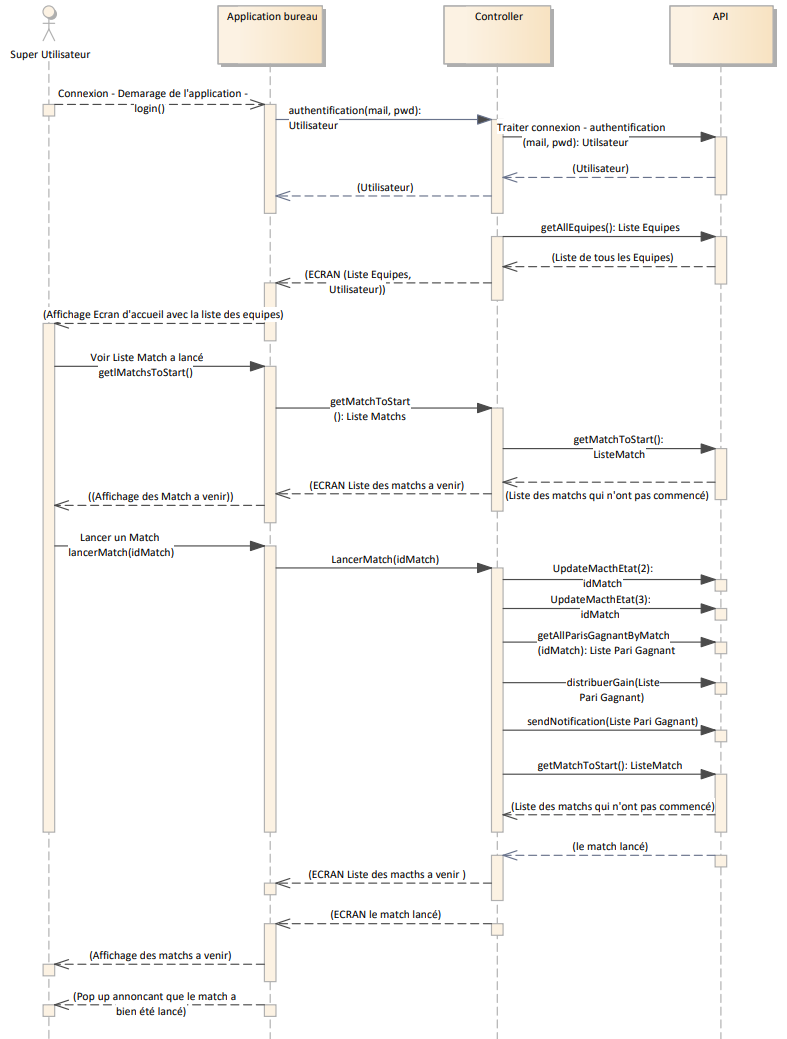
**Cas d’utilisation : Faire un pari sur mobile**

**Scénario normal :**



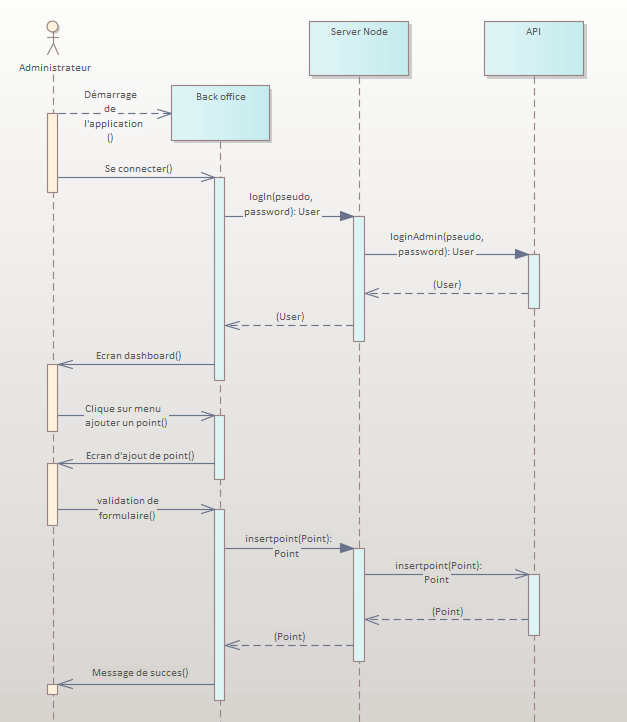
**Cas d’utilisation : Lancer un match sur l’application de bureau (backOffice)**

**Scénario normal :**



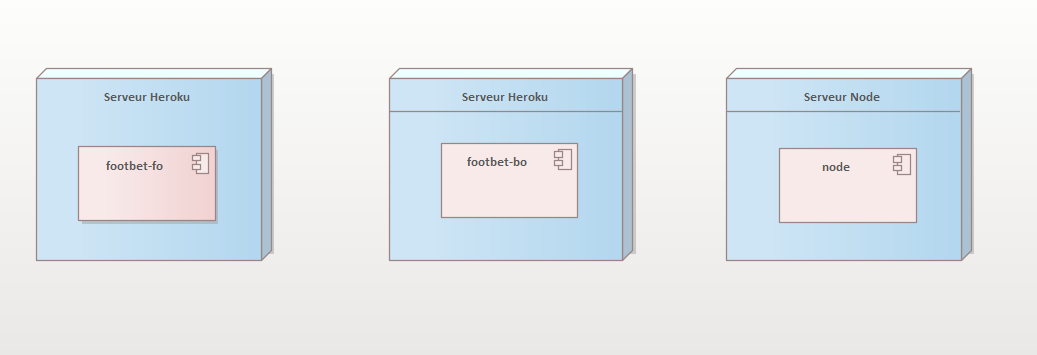
**Cas d’utilisation : Ajout point de vente sur web (backOffice)**

**Scénario normal :**



# Les composants et leur déploiement

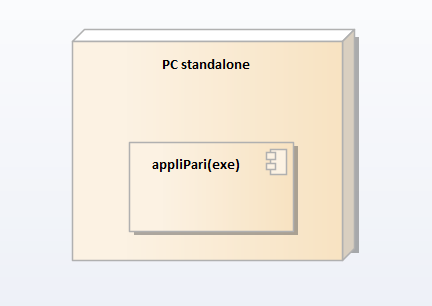
# Application web



# Client lourd

Le système est produit sous la forme d'un seul composant exécutable : un module ".exe" appelé « appliPari » :

Il est déployé sur l’unique nœud, le PC standalone :



# Application mobile

Le système est produit sous la forme d'une application mobile exécutable" appelée « appliPari » :

Il est déployé sur un téléphone mobile :

