Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance ▸ KICHENIN
Nom d'usage ▸ KICHENIN
Prénom ▸ EMELINE
Adresse ▸ 1, Allée Marie Curie Bat N Appât 338 ZUP 2
97420 LE PORT

Titre professionnel visé

Concepteur(trice) Développeur(se) Informatique

MODALITÉ D'ACCÈS :

☒ Parcours de formation
☐ Validation des Acquis de l'Expérience (VAE)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.
Ce titre est délivré par le Ministère chargé de l'emploi.

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen**.

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- une déclaration sur l'honneur à compléter et à signer ;
- des documents illustrant la pratique professionnelle du candidat (facultatif)
- des annexes, si nécessaire.

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

http://travail-emploi.gouv.fr/titres-professionnels

Sommaire

Exemples de pratique professionnelle

EXEMPLES DE PRATIQUE PROFESSIONNELLE

Activité-type 1 Développer des composants d'interface

Exemple n°1 ▸ ***Développement d'une application de protection d'animale – ENVOL TOIT***

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

ENVOL TOIT est une association de lutte contre la maltraitance animale de la Réunion. Á leur demande une application web nommée ENVOIL TOIT a été conçue.

Elle a pour but de faciliter la communication sur le suivi de l'animal (bilan de santé, rendez-vous chez le vétérinaire, antécédent de maladie …) confier à une famille d'accueil, avant de pouvoir par la suite être adopté.

Après le recueil des besoins de l'association, j'ai défini un ensemble de scénarios utilisateurs [1]en décrivant en détail ce que les utilisateurs pourraient faire sur le site et pourquoi ils le font. Qui m'ont servi de base pour mettre en place la navigation entre les interfaces :

- Sur la page d'accueil : les animaux à charge de la famille d'accueil connecté.
- Un menu de navigation est mis à la disposition de l'utilisateur « Accueil », « rendez-vous », « contact », « déconnexion », et la possibilité de prendre en charge des nouveaux pensionnaires dans la rubrique « Seconde chance »
- L'utilisateur peut accéder à son profil et le modifier.
- Quand l'utilisateur clique sur un animal à sa charge, ça le dirige la fiche détail de l'animal.
- Dans les détails de l'animal, sont affichés le nom de l'animal, son profil avec ses caractéristiques (nom, robe, numéro de puce, date de naissance, date trouvé, date recueillie par la famille d'accueil), liste des informations lié au rendez-vous chez le vétérinaire (la date du rendez-vous, la nature du rendez-vous, la prochaine date de visite, le montant, le poids …)
- Dans la rubrique « Rendez-vous », un calendrier est à la disposition de l'utilisateur. Lorsqu'un rendez-vous est pris, une notification email et message est envoyer au vétérinaire sélectionné par l'utilisateur. Après validation du rendez-vous par le vétérinaire, l'utilisateur est informé par SMS avec la date et heure.

À la suite de la rédaction du cahier des charges, j'ai commencé par me renseigner sur le type de technologie à utiliser pour mener à bien ce projet. Je voulais une technologie que je maîtrisais un

[1] Est comme une courte histoire d'une personne qui visite un site avec une certaine motivation et objectif précis en tête. Un scénario utilisateur inclut tous les renseignements qui sont pertinents pour le processus suivi par l'utilisateur en vue d'atteindre son objectif

peu. Ainsi mon choix s'est porté sur Symfony 3.4.22, qui est un Framework PHP, vu lors d'une formation sur PHP. Symfony dispose d'une bibliothèque appelée **Doctrine**, c'est un ORM (mapping object-relational mapping) qui est un type de programme informatique qui se place en interface entre le programme applicatif et une base de données relationnelle pour simuler une base de données orientée objet.
Ensuite, j'ai réalisé le diagramme de cas d'utilisation. Qui permet de donner une vision globale du comportement fonctionnel de notre application web.
Au début du projet, le logo ENVOIL TOIT a été fournie par le formateur. Après quelques navigations rapides sur les différents sites référencé sous le mot-clé « ENVOL TOIT », j'ai élaboré une feuille de style proposant 4 couleurs : Blanc, Orange, Beige crème, Marron.

Figure 1 : LOGO

#f4eade #ffe
antiquewhite
hsl(32,50,91)
rgb(244,234,222)

#f08e21 #f92
goldenrod
hsl(31,87,53)
rgb(240,142,33)

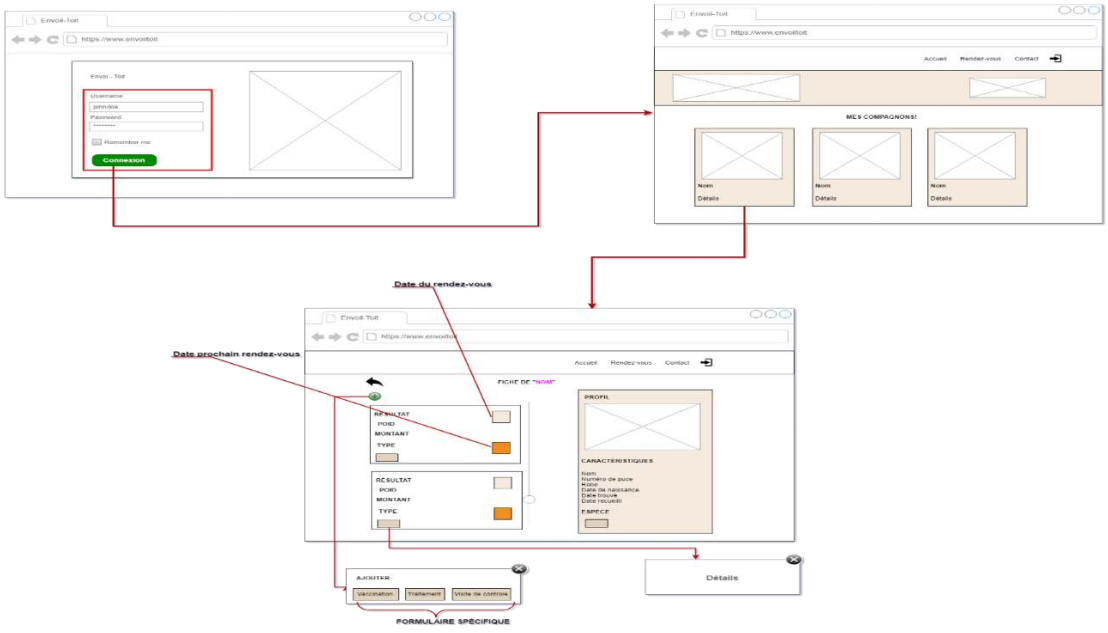
#ffffff #fff
white
hsl(0,0,100)
rgb(255,255,255)

#261b0d #221
black
hsl(33,49,10)
rgb(38,27,13)

Exemple de forme de texte :

Titre de page (h**1**) : 48px
Titre de section (**h3**) : 18px
Texte de paragraphe : 12px

Une fois tous ces éléments validés, j'ai mis en place ces éléments dans l'application pour proposer des enchaînements de maquettes :

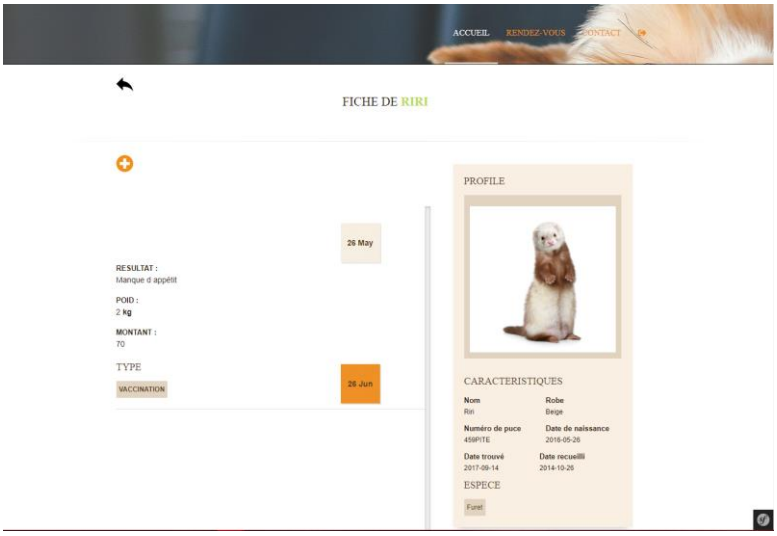


Et voici, le rendu de l'application :

- Vue de connexion :

- Vue de l'accueil :

- Vue sur la fiche d'information sur l'animal :



Pour permettre la prise en main rapide concernant les utilisateurs, j'ai rédigé une documentation utilisateur. Pour plus de détail veuillez consulter l'annexe 1.

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Editeur de texte**

J'ai utilisé **NetBeans**, qui est un environnement de développement intégré (IDE). Il permet la prise en charge native de divers langages d'où **Symfony** qui est un Framework **PHP**.

- **Création des maquettes**

Pour réaliser les maquettes, j'ai utilisé **Draw.io** qui est un outil disponible en ligne, gratuit avec la possibilité de l'enregistrer directement dans votre espace google drive. Il est rapide et simple d'utilisation.

- **Mise en forme des vues**

Pour avoir une continuité vis-à-vis des interfaces pensées durant le maquettage de l'application. J'ai utilisé du **Bootstrap** qui est une librairie utile à la création du design d'une application web.

- **Documentations**

- J'ai rédigé un **cahier des charges** à l'aide de *google docs* pour décrire le projet de manière technique à partir des quelques besoins fournis par le formateur (HOARAU Jérémy).
- J'ai utilisé comme documentations le livre par EXPERNET qui s'intitule « **HTML5 et CSS3** » qui regroupe les bonnes pratiques sur les mises en forme, l'utilisation de différents composants tels que les liens, les tableaux….
- Je me suis documenté sur Symfony, voici deux sources qui me paraissent pertinents :
  - **https://symfony.com/doc/current/index.html#gsc.tab=0** : ce site met à disposition une documentation complète sur les bonnes pratiques et l'utilisation de Symfony et Doctrine.
  - Livre fournis par École EXPERNET qui s'intitule **« SYMFONY 2 Développez des sites web PHP structurés et performants »** : qui s'adresse aux développeurs, chef de projets, directeurs techniques ou, plus largement. Mets à disposition l'apprentissage du Framework (Routage et contrôleur, les Templates avec Twig, les bases de données avec Doctrine.

3. Avec qui avez-vous travaillé ?

J'ai travaillé avec :

- ☞ M. HOARAU Jérémie (Formateur), afin de définir ensemble les besoins. Il représentait le maitre d'ouvrage, et était force de propositions quant aux habitudes utilisateurs.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ ENVOL TOIT

Chantier, atelier, service ▸ Dans le cadre de la formation : Développement d'une application web

Période d'exercice ▸ Du : 01/10/2018 au : 19/10/2018

5. Informations complémentaires *(facultatif)*

Activité-type 1 Développer des composants d'interface

Exemple n°2 ▸ *Développement d'une application d'évaluation à froid – EVAL'F*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

EVAL'F est une plateforme destinée aux responsables du centre de formation EXPERNET. Cette plateforme leur permet de personnaliser des questionnaires de type 'à froid' ayant pour destinataire les formateurs et les étudiants. Ainsi, cela leurs permettrait d'évaluer une formation.

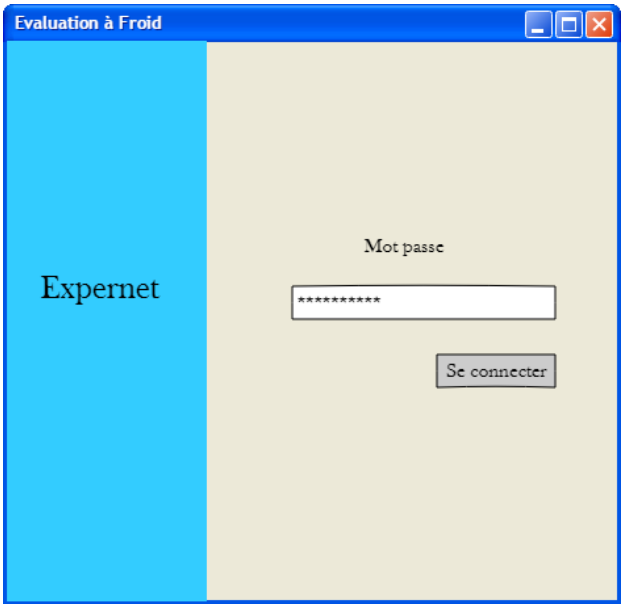
À partir de ces quelques informations et d'une fiche numérique que nous a fourni notre formateur, j'ai déterminé les spécifications techniques. La plateforme doit avoir :

- ☞ Une interface de connexion
- ☞ Une interface où il y a une liste de questionnaires avec la possibilité de supprimer, ajouter, et de voir les statistiques lier au questionnaire.
- ☞ Une interface permettant de visualiser le questionnaire et l'envoyer par mail.
- ☞ Une interface de statistique.
- ☞ La forme visuelle du questionnaire se base sur la fiche numérique fournie précédemment.

J'ai déterminé par la suite, la charte graphique de l'application. En respectant, la couleur du logo du centre de formation EXPERNET.

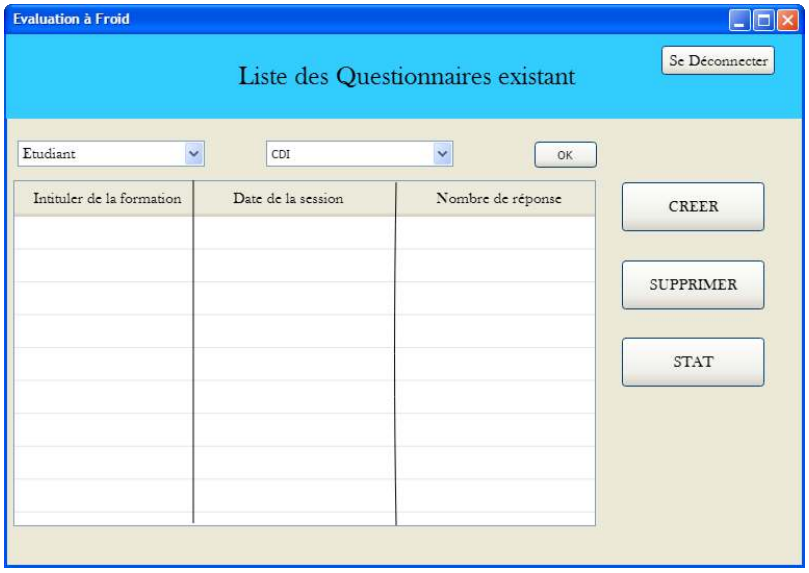
Voici les maquettes liées :

Après mure réflexion avec l'équipe, nous avons estimé que la plateforme n'exigera qu'un mot de passe pour l'authentification.

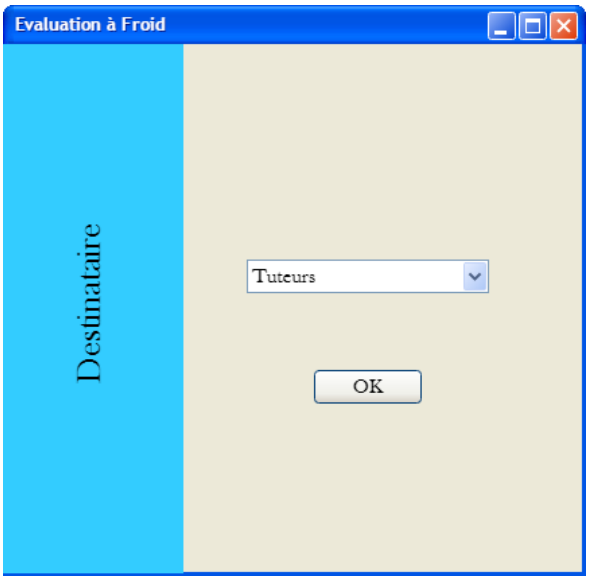


La redirection se fait vers l'interface de liste des questionnaires. Avec 3 boutons, possibilité de :

- ☞ Créer
- ☞ Supprimer
- ☞ Consulter les statistiques



Au moment de la création d'un questionnaire, l'utilisateur est dirigé vers cette fenêtre qui demande à qui est adressé ce questionnaire ?' Deux possibilités, par le biais d'une liste déroulante (Tuteur ou Étudiant).
Chaque choix a une interface distincte, composer d'un formulaire.



J'ai rédigé **le dossier technique**, qui regroupe les maquettes des écrans, des formulaires en précisant chaque type de composant que je vais utiliser durant la phase de conception, le schéma des enchaînements et le diagramme de classes.

Dans l'étape de réalisation, j'ai codé avec le langage de programmation orienté objet les interfaces utilisateur, c'est-à-dire j'ai mis en place les différentes classes présente dans mon diagramme de classes (en ne négligeant pas le principe d'encapsulation) et dont j'ai interagi avec chacun des composants d'interface.

Après avoir créé les interfaces graphiques, j'ai établi la **connexion à la base de données.** Sous NetBeans, j'ai intégré le driver MySQL. Lorsque la connexion est réussie, je me suis mise à coder les différentes actions permettant d'interagir avec la base de données. L'exemple suivant montre l'interface de connexion, je précise que le mot de passe est préalablement renseigné en base :

Mes composants d'interface et leur nommage :

- JPasswordField pour le mot de passe – **jPasswordField1**
- Bouton « Se connecter » - **jButton1**

```
private void jButton1ActionPerformed (java.awt.event.ActionEvent evt) {
  if (jPasswordField1.getPassword().length==0) // Vérification si le champ vide
    JOptionPane.showMessageDialog(null, " Champs vides détectés ! Veuillez remplir tous les champs ");
  else {
     char [] pass = jPasswordField1.getPassword(); // Récupère les entrées
     String pwd = String.copyValueOf(pass) ; // conversion du tableau en chaîne
     if(validate_login(pwd))
       JOptionPane.showMessageDialog(null, "Identifiant de connexion correcte") ;
     else
       JOptionPane.showMessageDialog(null, " Identifiant de connexion incorrecte ") ;
  }
}
```

```
private boolean validate_login(String password) {
  try {
     Class.forName("com.mysql.jdbc.Driver");  // Connexion à la base de données MySQL
     Connection conn = DriverManager.getConnection("jdbc: mysql://localhost:3306/evaf?" +
"user=root&password=");
     PreparedStatement pst = conn.prepareStatement("Select * from login where password=?");
     pst.setString(1, password);
     ResultSet rs = pst.executeQuery();
     if(rs.next())
        return true;
     else
        return false;
  }
  catch (Exception e) {
```

Lorsque l'application fut conçue, j'ai rédigé la documentation utilisateur.

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- ☞ **Editeur de texte**

J'ai utilisé **NetBeans**, qui est un environnement de développement intégré (IDE). Il permet la prise en charge native de divers langages d'où **Java**. Pour cette application est du type **Java Application**.

- ☞ **Création des maquettes**

Pour réaliser les maquettes, j'ai utilisé **PENCIL** qui est un logiciel libre et gratuit de création de maquette développé par Évolution Solution. Il est rapide et simple d'utilisation.

- ☞ **Mise en forme des vues**

Pour avoir une continuité vis-à-vis des interfaces pensé durant le maquettage de l'application. J'ai utilisé du **JFrame form**.

- ☞ **Outils collaboratifs**

Pour ce projet, nous avons utilisé les outils de Google pour faciliter nos échanges. Avec l'utilisation de Sheets de Google nous nous sommes fait un tableau de répartition des tâches, ce rapprochant du concept du logiciel Trello.

- ☞ **Test Unitaires**

Pour vérifier mes tests unitaires, j'ai utilisé Cucumber qui est un outil qui permet de tester notre code.

- ☞ **Documentations**
  - J'ai rédigé un **dossier technique** et **une documentation utilisateur** à l'aide de ***google docs*** pour décrire l'application et son utilisation.
  - J'ai utilisé comme documentations le livre par EXPERNET qui s'intitule « **JAVA** » ;
  - Je me suis documenté sur la base de Java Application combiné au JFrame Form, voici quelques sources qui me paraissent pertinents :
    - **https://netbeans.org/kb/docs/java/gui-functionality.html**
    - Livre fourni par École EXPERNET qui s'intitule « **Java Application** ».
  - Pour l'utilisation de Cucumber le site officiel https://cucumber.io/docs est une excellente référence.

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec des étudiants de la promotion CDI (Concepteur Développeur Informatique) travaillant dans différentes entreprises suivantes :

- ☞ Hoarau John : Groupe MDSI
- ☞ Bastide Sébastien : TEEO

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Dans le cadre de la formation : Développement en couche avec Java

Période d'exercice ▸ Du : 08/01/2018 au : 19/01/2018

5. Informations complémentaires *(facultatif)*

Activité-type 2 Développer la persistance des données

Exemple n° 1 ▸ ***Développement d'une application de protection d'animale – ENVOL TOIT***

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour réaliser cette activité type, j'ai divisé l'activité en plusieurs phases qui sont les suivantes :

1. **Phase d'analyse**

Après le recensement des besoins vus précédemment. J'ai réalisé le **diagramme de cas d'utilisation**. Á l'aide de draw.oi.
Ce qui m'a emmené à établir des **règles de gestion** suivantes :

- Un administrateur ne gère aucun ou plusieurs animaux ;
- Un animal est géré par un seul et unique administrateur ;
- Un animal est accueilli par une seule et unique famille ;
- Un animal est transporté par un seul et unique transporteur ;
- Un animal possède une seule et unique espèce ;
- Une espèce est liée à aucun ou plusieurs animaux ;
- Un animal est associé à aucun ou plusieurs événements ;
- Un événement est associé à un seul et unique animal ;

Après plusieurs informations données par le maître d'œuvre. J'ai déterminé les données élémentaires, les éventuelles données calculées, et les données composées en précisant leurs types. Ce qui m'a permis de composer rapidement le **dictionnaire de données**.

2. **Phase de conception technique**

Durant cette phase, J'ai mis en place **les différents modèles**, en m'appuyant et épurant le dictionnaire de données.
Voici les différents modèles que j'ai réalisés dans l'ordre suivant :

I. Le modèle **conceptuel** de données du type **modèle entité-association**. Qui permet de décrire le système d'information à l'aide d'entité.

II. Modèle **logique de données**, qui consiste à décrire la structure selon laquelle les données seront stockées en dans la base de données. Voici le modèle logique de données modélisant notre projet ENVOL-TOIT.

ek_personne(**id**, nom, prenom, adresse, telephone, code_postale, nature)

ek_transporteur(***id_transporteur***, dateDepart, specificiter)

ek_veterinaire(***id_veterinaire***)

ek_admin(***id_admin***)

ek_famille(***id_famille***)

ek_espece(**id**, libelle)

ek_animal(**id**, nom, robe, numero_puce, image, dateNaissance, poid, #id_personne, #id_espece)

ek_calendrier(**id**, dateD, dateF, sujet, #id_personne)

ek_evenement(**id**, date, date_suivante, montant, nature)

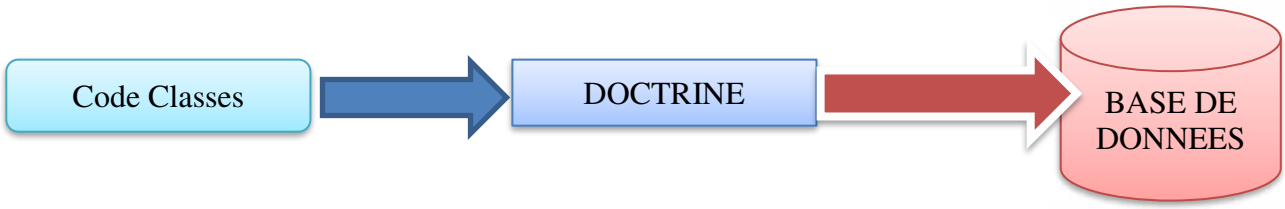
ek_visite(***id_visite***, *bilan, poid)*

ek_traitement(***id_traitement***, symptome, maladie, traitement)

ek_vaccination(***id_vaccination***, produit)

III. Le **modèle physique** est un mixte entre le ***modèle conceptuel*** de données et le ***modèle logique*** de données. Je représente chaque entité et leurs attributs en précisant leur **type** (varchar(..), int …). Mais je représente aussi **les clés étrangères**.

IV. Le **diagramme de classes**, à partir de toutes ces modélisations. Je réalise le plus important d'entre eux car il représente l'ensemble des objets de notre système. De plus, j'ai conçu l'application en **Code First**.



3. Phase de réalisation

Comme je l'ai précisée précédemment, j'ai utilisé le mécanisme du Code First. C'est-à-dire, que tout d'abord j'ai codé toutes mes classes types '***@Entity'*** et généré automatiquement ma base de données et mes tables par le biais de ***Doctrine***.
Ensuite, lorsque ma base et mes tables furent construites. J'ai mis en place **mes scripts de jeux d'essaies**. Sous Symfony cette action se nomme « ***Fixtures*** ».

```php
<?php

namespace AppBundle\DataFixtures;

use AppBundle\Entity\ek_Species;
use Doctrine\Bundle\FixturesBundle\Fixture;
use Doctrine\Common\DataFixtures\OrderedFixtureInterface;
use Doctrine\Common\Persistence\ObjectManager;

class SpeciesFixtures extends Fixture implements OrderedFixtureInterface {

    /**
     * Get the order of this fixture
     *
     * @return integer
     */
    function getOrder() {
        return 1;
    }

    //put your code here
    public function load(ObjectManager $manager) {
```

Ce script permet de remplir la table '*Species*' (Espèce) la fonction ***getOrder ()*** est l'ordre dans lequel le script doit être exécuté. Dans la fonction ***load (ObjectManager $manager)*** je crée un tableau associatif contenant des chaînes de caractères. Je fais un ***foreach ()*** sur ce tableau, je crée un nouvel objet de type ***Species(),*** je ***set()*** l'attribut que je souhaite remplir et j'itère sur chaque ligne de mon tableau**.** Pour finir, je déclenche l'enregistrement avec un ***flush ().***
Par la suite, je mets en place mon **script de sauvegarde** :

- Je détermine une route où mon composant d'interface pourra l'appelé.
- Je les données de connexion à la base de données (Host, Base de données, Utilisateur, Mot de passe).
- Je regroupe les données de connexion au serveur FTP (Serveur, Utilisateur, Mot de passe, le chemin absolu de destination du fichier de base de données.
- J'ouvre en création ou en modification le fichier log de la journée.
- Je démarre la connexion.
- Je liste d'abord l'ensemble des tables.
- J'exclus éventuellement les tables indiquées.
- La variable $return contiendra le script de sauvegarde. J'englobe le script de backup dans une transaction et je désactive les contraintes de clés étrangères.
- Je boucle sur l'ensemble des tables à sauvegarder, je valide la transaction et je réactive les contraintes de clés étrangères.
- J'enregistre maintenant le script SQL dans un fichier au format gzip.
- J'envoie maintenant le fichier gzip au serveur FTP indiqué.
- J'écris le log dans le fichier de logs de la journée.

À partir des règles de gestion concernant la prise de rendez-vous, j'ai mis en place un trigger qui permet de vérifier si, un créneau que souhaite avoir un utilisateur est libre.

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Outil de modélisation**

J'ai utilisé **JMerise**, qui est un outil de modélisation de base de données. Il met à disposition la possibilité de créer des dictionnaires de données, de faire des MCD (Modèle Conceptuel de Données) et de vérifier si celui-ci est correcte. À partir, d'un MCD correct JMerise peut générer automatiquement le MLD (Modèle Logique de Données). C'est un outil de gain de temps.

- **SGBDR**

J'ai utilisé comme système de gestion de base de données relationnelles **PhpMyAdmin** qui incorpore le **langage SQL**.

- **Outil de gestion de projet**

Pour gérer ce projet, j'ai utilisé **GitLab**. Qui propose un hébergement gratuit, un système de versionning, et surtout un tableau de bord qui permet de planifier chacune des fonctionnalités de l'application sous forme d'étiquette (À faire, En cours, Fermer).

- **Documentations**

Je me suis documenté sur Symfony, voici deux sources qui me paraissent pertinentes :

- **https://symfony.com/doc/current/index.html#gsc.tab=0** : ce site met à disposition une documentation complète sur les bonnes pratiques et l'utilisation de Symfony, Doctrine et la création des classes qui permette le remplissage des tables (**Fixture**).
- Livre fourni par École EXPERNET qui s'intitule **« SYMFONY 2 Développez des sites web PHP structurés et performants »** : qui s'adresse aux développeurs, chef de projets, directeurs techniques ou, plus largement. Met à disposition l'apprentissage du Framework (Routage et contrôleur, les Templates avec Twig, Les bases de données avec Doctrine.

3. Avec qui avez-vous travaillé ?

J'ai travaillé avec :

- M. HOARAU Jérémie (Formateur), afin de définir ensemble les besoins. Il représentait le maitre d'ouvrage, et était force de propositions quant aux habitudes utilisateurs. Mais aussi pour la validation du Modèle Conceptuel de données.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Cadre de la formation : Développement Web côté serveur avec PHP et Symfony

Période d'exercice ▸ Du : 09/07/18 au : 20/07/18

5. Informations complémentaires *(facultatif)*

Activité-type 2 Développer la persistance des données

Exemple n° 2 ▸ Développement d'une application d'organisation de jeux– TINTOUR

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

TINTOUR est une application planification de jeux. Le concept est de permettre aux utilisateurs d'organiser des tournois et ainsi permettre une rencontre avec une date, une heure, un lieu et le jeu concerner (Billard, Baby-foot).

Dans un premier temps, après le recensement des besoins nous avons établi les différents cas d'utilisation. Ce qui nous a emmené à définir des règles gestion voici quelques-unes :

- **Un utilisateur (joueur) peut être propriétaire d'un à 5 salons.**
- **Un salon peut contenir un ou plusieurs utilisateurs (joueurs).**
- **Un utilisateur peut avoir un ou plusieurs ami(s) joueur.**
- **…**

Après plusieurs échanges avec les membres de l'équipe, j'ai recueilli diverses données, déterminé les données élémentaires, les données calculées, et les données composées en précisant la typologie de chaque donnée. Ce qui m'a permis de mettre en place rapidement un dictionnaire de données.

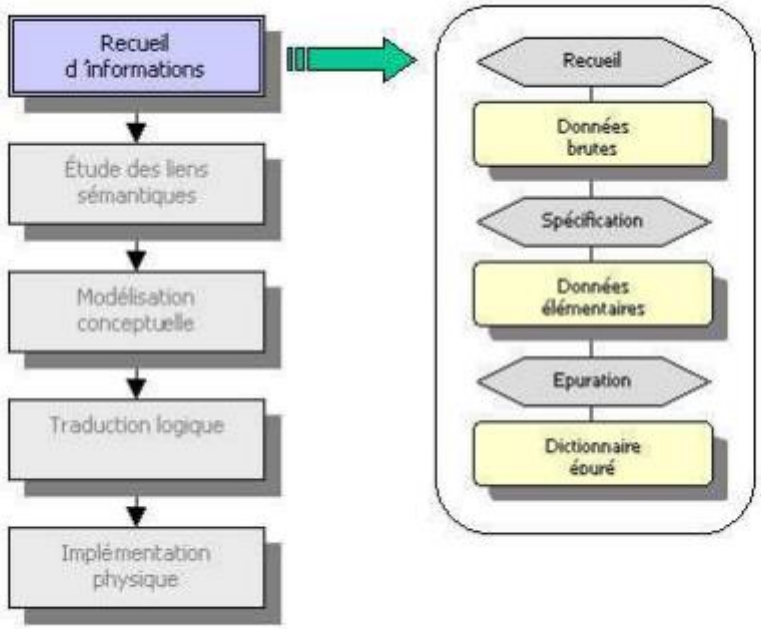
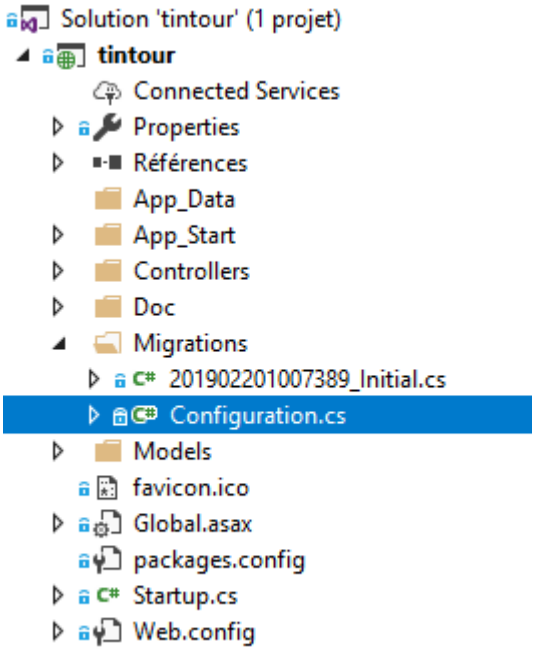


Figure 2 : Les phases de conception BDD

Dans un second temps, dans la conception technique, j'ai mis en place la phase de modélisation des données, en épurant ce dictionnaire mis en place précédemment.
Comme le montre la Figure 2, j'ai réalisé tout d'abord la modélisation conceptuelle, ensuite la modélisation logique et par la suite la modélisation physique des données.
Pour finir cette conception technique, j'ai représenté l'abstrait des objets du système qui vont interagir pour réaliser le cas d'utilisation par le biais d'un diagramme de classes.

Durant la phase de réalisation, j'ai créé les différentes classes de type entité. Puis, j'ai généré la base de données opérationnelle avec Entity Framework sous ASP.NET de Microsoft Visual Studio. Les tables sont créées sous SQL Server[2]. C'est un mécanisme qui se nomme Code First.
L'activation de la migration pour la création de la base de données, créée un nouveau dossier Migrations dans l'application contenant un fichier Configuration.cs. Permettant de définir le comportement de la migration pour le DbContext utilisé.



Dans la fonction ***Seed()*** de la classe Configuration, j'ai mis en place le script de jeux d'essai de la base de données. Au lancement d'une nouvelle commande de migration, les tables se remplissent automatiquement grâce à ce scripte.

[2] Microsoft SQL Server est un système de gestion de base de données en langage SQL incorporant entre autres un SGBDR développé et commercialisé par la société Microsoft.

```
protected override void Seed(tintour.Models.DbCtxt context)
{
    //  This method will be called after migrating to the latest version.

    //  You can use the DbSet<T>.AddOrUpdate() helper extension method
    //  to avoid creating duplicate seed data.

    //Players
    context.Player.AddOrUpdate(x => x.Id,
        new Player() { LastName = "Austen", FirstName = "Jane", Email = "jane@gmail.com" , Password = "jane", InRoom
        new Player() { LastName = "Tiphanie", FirstName = "Tremayne", Email = "ttremayne1@blogger.com", Password = "
        new Player() { LastName = "Gil", FirstName = "Hirth", Email = "ghirth2@themeforest.net", Password = "FvirRRB
        new Player() { LastName = "Ofilia", FirstName = "Tythe", Email = "otythe3@umich.edu", Password = "oTsj4S", I
        new Player() { LastName = "Zsazsa", FirstName = "Stockle", Email = "zstockle4@nationalgeographic.com", Passw
        new Player() { LastName = "Isiahi", FirstName = "Antuk", Email = "iantuk5@printfriendly.com", Password = "ca
        new Player() { LastName = "Caldwell", FirstName = "Faltin", Email = "cfaltin6@sakura.ne.jp", Password = "WKk
        new Player() { LastName = "Carry", FirstName = "Dowman", Email = "cdowman7@tinypic.com", Password = "HRhB5P"
        new Player() { LastName = "Dorothea", FirstName = "Thews", Email = "dthews8@squarespace.com", Password = "10
        new Player() { LastName = "Tybi", FirstName = "Seiler", Email = "tseiler9@google.es", Password = "QKekuNFBaw
    );

    //Rooms
    context.Room.AddOrUpdate(r => r.Id,
       new Room() { CreatedAt = new DateTime(2019,01,01), Label = "Team974", MaxPlayers = 6, Location = "St Pierre",
       new Room() { CreatedAt = new DateTime(2019,01,15), Label = "Les Aigles", MaxPlayers = 4, Location = "St Leu",
       new Room() { CreatedAt = new DateTime(2019,01,05), Label = "LéBoss", MaxPlayers = 8, Location = "Le Port", Is
       new Room() { CreatedAt = new DateTime(2019,02,10), Label = "LéZab", MaxPlayers = 2, Location = "St Joseph", I
    );

    //Game
    context.Game.AddOrUpdate(g => g.Id,
        new Game() { Label = "Billard" },
        new Game() { Label = "Baby-Foot" }
    );

    //Friends
    context.FriendsInPlayers.AddOrUpdate(f => f.Id,
        new FriendsInPlayers() { PlayerId = 1, FriendId = 2 },
        new FriendsInPlayers() { PlayerId = 2, FriendId = 3 },
        new FriendsInPlayers() { PlayerId = 3, FriendId = 4}
        );
```

Côté client, je réalise deux boutons vue administrateur « Sauvegarde » et « Restauration ». Ensuite deux services.ts avec un appel direct à l'API.

```
Private _url = 'http://localhost:8080/api/BackupConfiguration';
Private _url = 'http://localhost:8080/api/RestoreConfiguration';
```

Côté serveur, je crée un nouveau contrôleur BackUpController.cs, j'implémente une nouvelle route.

```
[Route("api/BackupConfiguration")]
[Route("api/RestoreConfiguration")]
```

À partir de la règle de gestion suivante « **Un utilisateur (joueur) peut être propriétaire d'un à 5 salons** », avec l'équipe on a décidé de mettre en place un trigger qui se déclenche lors de la création d'un salon par un utilisateur, qui va compter le nombre de salon dans la table Rooms dont l'identifiant du joueur propriétaire est égal à celui de l'utilisateur qui réalise l'action de création du salon. Si, le résultat obtenu est inférieur à 5, le salon est créé. Sinon, un message d'erreur est retourné « nombre de salon trop important ».
Comme procédure stocker, nous avons opté pour une simple mise en majuscules des intitulés après création d'un salon.

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Editeur de texte**

J'ai utilisé **Visual Studio Code**, côté *Client* de l'application qui propose des extensions **Angular**, de bonne pratique. Et permet de se connecter au l'hébergeur **GitLab** ou **GitHub**.
Côté *Serveur*, j'ai utilisé **Microsoft Visual Studio**. Célèbre IDE, qui propose une suite de logiciels de développement. Dont dans ce projet, on a utilisé **ASP.NET** qui est un Framework de gestion à la demande des pages web.

- **Chrome**

Évidemment, pour visualiser le résultat du développement et valider des tests utilisateurs j'utilise comme navigateur **Chrome**.

- **SGBDR**

J'ai utilisé comme système de gestion de base de données relationnelles **Microsoft SQL Server** qui incorpore le **langage SQL**.

- **Outils collaboratifs et de gestion**

Ce projet a été réaliser en équipe, pour faciliter nos échanges.
Nous avons utilisé :

- **GitLab** : qui a permis d'avoir une base de projet identique. Avec le système de branche, chacun a pu travail sur différentes fonctionnalités. De plus, le système de CI/CD (système d'intégration continue) réduit les marges d'erreur de l'application. GitLab propose un tableau de bord, ce qui nous permet de faire la gestion du projet, planifier les différentes tâches à réaliser et les répartir.
- **Google Drive** : nous avons permis d'échanger les différents documents techniques, de travail collaborativement sur chaque présentation.

- **Documentations**

ASP.NET
http://www.eni-training.com/cs/expernet/olr

Entity framework
https://docs.microsoft.com/en-us/ef/core/

Angular 7.3.5
https://angular.io/docs

Angular material
https://material.angular.io/components/categories

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec des étudiants de la promotion CDI (Concepteur Développeur Informatique) travaillant dans différentes entreprises suivantes :

- Hoarau John : Groupe MDSI
- Bastide Sébastien : TEEO
- Coli-Jun-Ong Kenny : De la mutualité de la Réunion

Nous constituons les maîtres d'œuvre, avec Hoareau John comme chef de projet et Bastide Sébastien, Coli-Jun-Ong Kenny, moi-même en tant qu'équipiers de projet.

L'application s'adresse aux étudiants du centre de formation d'Expernet, qui sont donc les maîtres d'ouvrage et nos utilisateurs Béta.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Dans le cadre de la formation : Développement d'une application mobile ECF 3

Période d'exercice ▸ Du : 23/04/19 au : 26/04/19

5. Informations complémentaires *(facultatif)*

Activité-type 3 Développer une application n-tiers

Exemple n° 1 ▸ *Développement d'une application d'organisation de jeux– TINTOUR*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

L'application Tintour, présenter dans l'activité précèdent à une portée numérique. Pour démontrer l'activité-type 3, je m'appuierai sur cette version mobile.

Analyse :
Nous avons utilisé le cahier des charges de l'application web, pour établir les cas d'utilisation (**Figure 3**) avec les scénarios de cette application mobile.

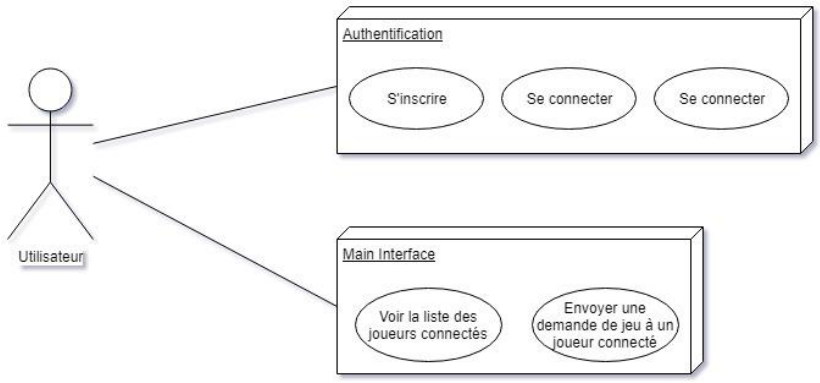


Figure 3 – Diagramme de cas d'utilisation

- Scénario :

L'utilisateur se connecte, si l'utilisateur n'a pas de compte, il peut s'inscrire. Lorsque l'utilisateur c'est enfin connecté, il accède à la liste de personne en ligne à proximité. De ce fait, l'application utilise le système de géolocalisation du téléphone.

Conception technique :
Dans cette étape, nous avons mis en place le diagramme de séquence détaillés (**Figure 4**), le diagramme de classes et le diagramme de déploiement (**Figure 5**). De plus, nous avons définit l'architecture physique et logicielle de l'application.
L'application ce base sur le design pattern MVC (Model – View - Controller), dont la couche model représente la partie de l'application qui exécute la logique métier, ce qui signifie qu'elle est responsable de la récupération des données. La vue retourne une présentation des données venant du model. Elle est responsable de l'utilisation des informations dont elle dispose pour produite une interface de présentation. Et enfin la couche Controller qui gère les requêtes des utilisateurs, il est responsable de retourner une réponse avec l'aide mutuelle des couche Model et Vue.
Comme technologies, nous avons décidé d'utiliser Kotlin qui est un langage de programmation car il permet de rendre une meilleure productivité et améliorer la pérennité du développeur dace à certains types d'erreurs récurrentes en Java comme par exemple le « NullPointerException ». Pour la base de données nous avons décidé d'utiliser Firebase.

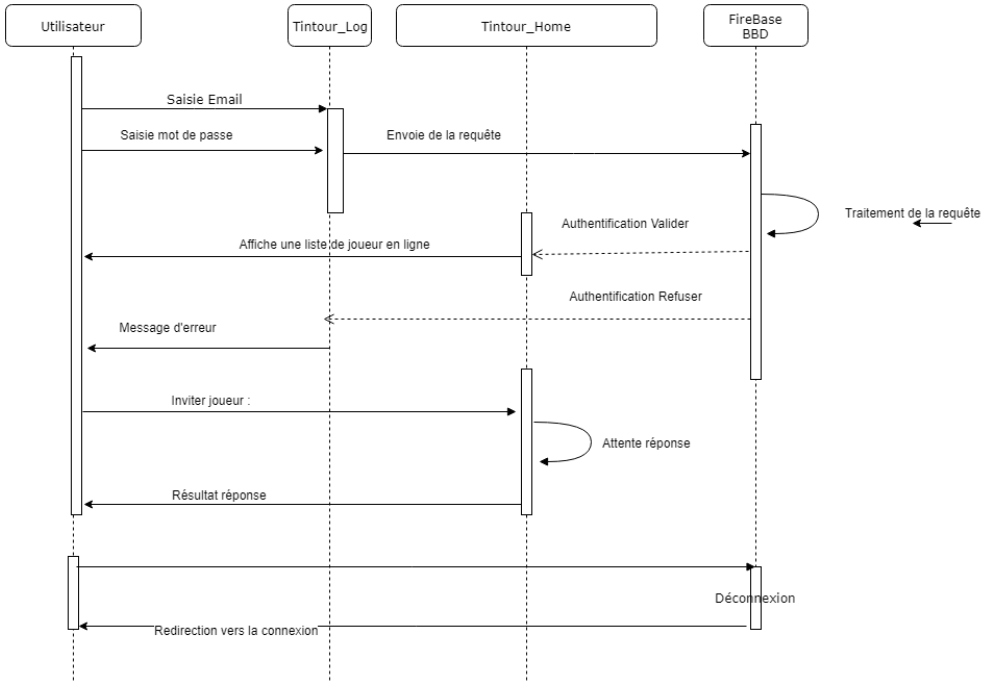


Figure 4 - Diagramme de séquence

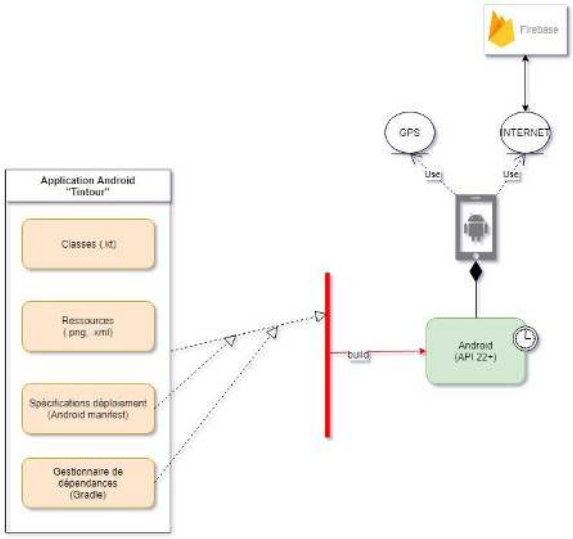


Figure 5- Diagramme de déploiement

Réalisation :

- Composants métier :

Nous avons développé, tout d'abord notre classe Player (Figure 5). Ensuite, nous avons mis en place le système de communication a Firebase (Figure 6). Étant donné que nous utilisons le système de géolocalisation de l'appareil mobile, il était indispensable de développer une classe permettant de récupérer cette distance et de déterminer la proximité des joueurs.

```
package com.expernet.tintour.models


/**
 * @class Player
 * Represents a player
 */
data class Player(
    val id          : String,
    val email       : String,
    var password    : String,
    val lastName    : String,
    val firstName   : String,
    var isOnline    : Boolean
) {
```

Figure 6 - Classe player

```
package com.expernet.tintour.models

import ...


class LocationListener : LocationListener {
    override fun onLocationChanged(location: Location?) {
        val long : Double? = location?.longitude
        val lat : Double? = location?.latitude

        if(long != null && lat != null) {
            updateLocation(long, lat)
        } else {
            updateLocation(MainActivityInterface.lastPosition.longitude, MainActivityInterface.lastPosition.latitude)
        }

        Log.d( tag: "LOC_CHANGED_1", OnlineUser.onlineUsers.toString()) // DEBUG
    }
```

Figure 7 - Classe de Géolocalisation

```
object OnlineUserManager{

    // Write a message to the database
    val database = FirebaseDatabase.getInstance()
    val myRef = database.getReference(OnlineUser::class.java.simpleName)

    /**
     * Saves OnlineUsers.onlineUsers to firebase
     */
    fun save() {
        myRef.setValue(OnlineUser.onlineUsers)
    }

    fun add(user: OnlineUser) {
//        myRef.child(user.id).setValue(user)
        myRef.child(user.id).setValue(user)
    }

    fun delete(uid: String) {
        myRef.child(uid).removeValue()
    }
```

Figure 8 -Connexion et interaction avec la base de données

Outil collaboratif et de version :

Durant ce projet, étant donné qu'ont travaillé en groupe, il était indispensable de se répartir les tâches pour le bon avancement du projet. Dans cette optique, nous savons mit en place un repositorie sur GitLab du projet, GitLab est un outil de gestion de projet qui permet de gérer ces

dépôts Git. A partir de cela, nous von créer des branches spécifiques pour chacun des membres de l'équipe (Figure 9). On peut voir en premier lieu les noms des branches crées parles développeurs qui compose l'équipe qui don « ek », « sb », « jh », « kc ». L'icône représentent les développeurs et le texte sur la même ligne que l'icône est une description explicative des fonctionnalités développé. Sous cette forme d'organisation des fonctionnalités cela apporte une gestion des versions de l'application.

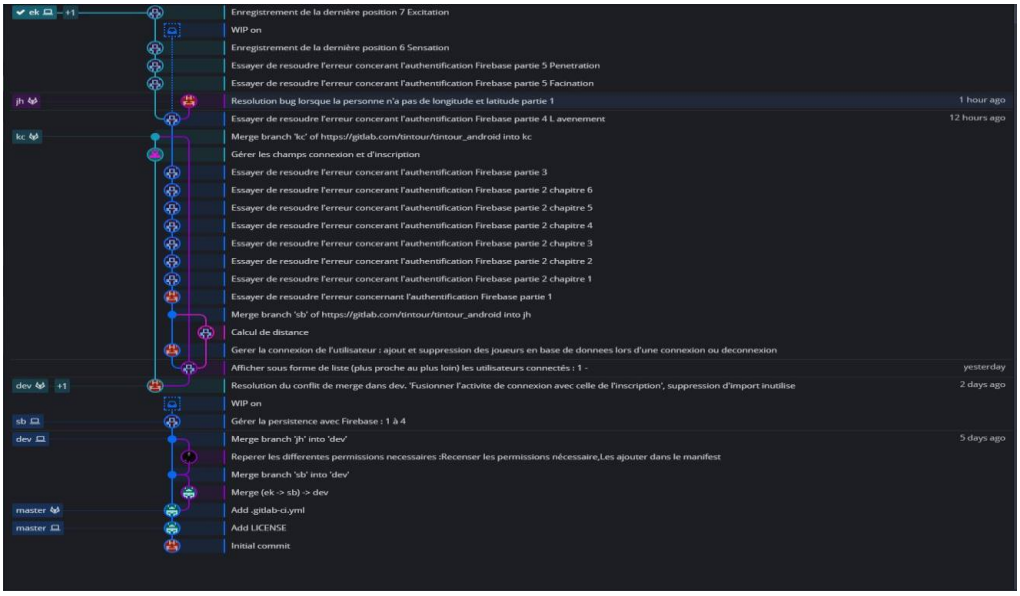


Figure 9 - Les différents commit et push

Gestion du projet :

Je souligne que durant ce projet, je n'étais pas chef de projet. Nous avions régulièrement chaque matin des **réunions brèves** pour définir l'avancement des fonctionnalités attribuer à chacun de nous.
Nous utilisons GitKraken glo pour définir la liste des fonctionnalités à développer sous forme de tickets, et à les attribuer à chacun des membres de l'équipe (Figure 10).

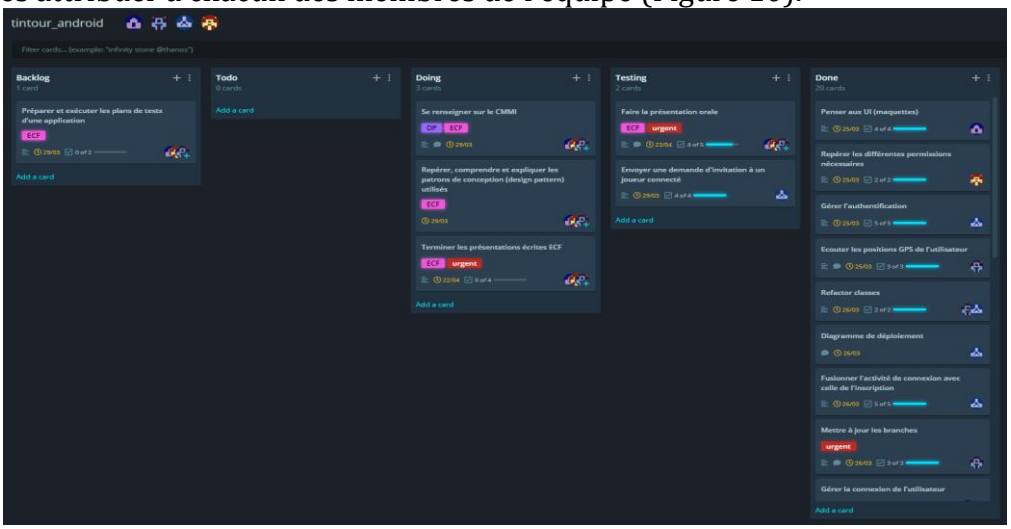


Figure 10 - Gestions des features

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Editeur de texte**

J'ai utilisé **Android Studio** qui est un environnement de développement pour développer des applications mobiles Android. Il intègre le langage **Kotlin**, langage que nous avons décidé d'utiliser pour ce projet. De plus Android Studio met à disposition des émulateurs pour avoir un visuel de notre application.

- **Base de données**

J'ai utilisé comme système de gestion de base de données **Cloud Firestore** (outil intégré à **Firebase**[3]) qui incorpore le **langage NoSQL** basé sur le type documentaire.

- **Outils collaboratifs et de gestion**

Ce projet a été réaliser en équipe, pour faciliter nos échanges.
Nous avons utilisé :

- **GitLab** : qui a permis d'avoir une base de projet identique. Avec le système de branche, chacun a pu travail sur différentes fonctionnalités. De plus, le système de CI/CD (système d'intégration continue) réduit les marges d'erreur de l'application. Et nous permet de faire du versionning.
- **Google Drive** : nous a permis d'échanger les différents documents techniques, de travail collaborativement sur chaque présentation.
- **GitKraken** : est une couche cliente de Git. Il propose un tableau de bord **Glo Boards**, ce qui nous permet de faire la gestion du projet, planifier les différentes tâches à réaliser et les répartir

- **Documentations**

KOTLIN
Je me suis initiée à Kotlin avec Openclassrom https://openclassrooms.com/fr/courses/5353106-initiez-vous-a-kotlin

ANDROID STUDIO
Pour la prise en main de Android Studio je me suis référer à la documentation officielle https://developer.android.com/docs.

[3] Est un ensemble de services d'hébergement pour n'impote quel type d'application. Il propose d'héberger en NoSQLet en temps réel des bases de données, de contenu, de l'authentification socialeet des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel.

FIREBASE
Pour la prise en main de Firebase je me suis tourné vers la documentation officielle https://firebase.google.com/docs

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec des étudiants de la promotion CDI (Concepteur Développeur Informatique) travaillant dans différentes entreprises suivantes :

- Hoarau John : Groupe MDSI
- Bastide Sébastien : TEEO
- Coli-Jun-Ong Kenny : De la mutualité de la Réunion

Nous constituons les maîtres d'œuvre, avec Hoareau John comme chef de projet et Bastide Sébastien, Coli-Jun-Ong Kenny, moi-même en tant qu'équipiers de projet.

L'application s'adresse aux l'étudiants du centre de formation d'Expernet, qui sont donc les maîtres d'ouvrage et nos utilisateurs Béta.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Dans le cadre de la formation : Développement d'un application mobile

Période d'exercice ▸ Du : 18/03/19 au : 29/03/19

5. Informations complémentaires *(facultatif)*

Activité-type 3 Développer une application n-tiers

Exemple n° 2 ▸ ***Développement d'une application de souhaits– LZZ974***

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

L'application LZZ974 (Liste Zote Zaffaire 974) est une application mobile permettant aux utilisateurs de lister leurs vœux.

L'objectif est de s'appuyer sur une base de données SQLLite qui permet de créer des catégories de de souhaits (Voitures, Cadeau de Noel…). Par le biais d'un bouton d'ajout présent sur l'interface, un pop-up s'ouvre et demande à l'utilisateur de saisir le nom de la catégorie qu'il souhaite créer.
Lorsque la catégorie est créée, l'utilisateur peut la supprimer ou la modifier. En cliquant, sur le nom de la catégorie, l'utilisateur est dirigé vers l'interface d'article, il peut par un bouton d'ajout d'article, ajouter un nouvel article où il doit renseigner le nom, le prix et ajouter une photo directement en utilisant l'appareil photo de son mobile ou par un bouton d'accès à sa galerie de téléphone.

Par la suite, j'ai élaboré le cas d'utilisation en utilisant draw.io (**Figure 11**).

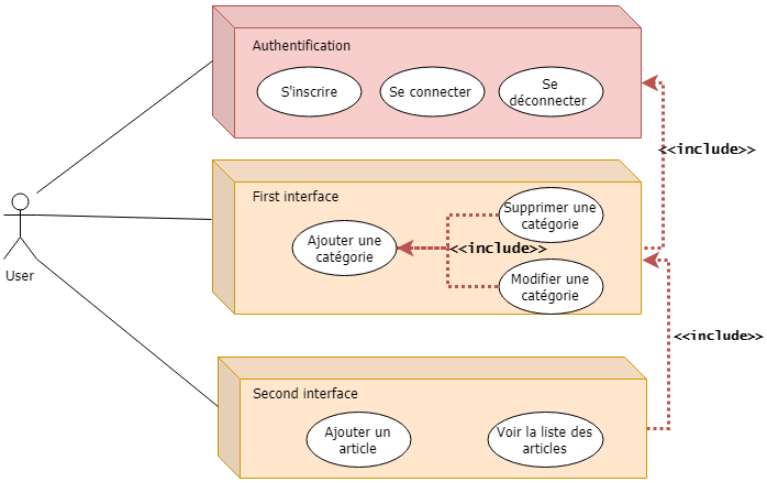


Figure 11 - Diagramme de cas d'utilisation

Conception technique :
J'utilise pour cette application flutter qui est un Framework de Google pour la création d'application mobile multiplateforme. L'avantage de ce genre de technologie est de pouvoir développer l'application qu'une fois et de le déployer sur PlayStore mais également sur AppStore. Flutter utilise le langage Dart qui est un langage créé par Google. Dart permet de générer des applications native s pour iOS et Android ce qui est intéressant car il peut s'optimiser pour les plateformes sur lesquelles il est exécuté. Pour ce projet, j'ai utilisé comme design pattern d'architecture logicielle MVSP (Model View Scoped Présenter), c'est un modèle qui permet de gérer l'état des widgets et ainsi rendre mon application plus flexible, maintenable (**Figure 12**).

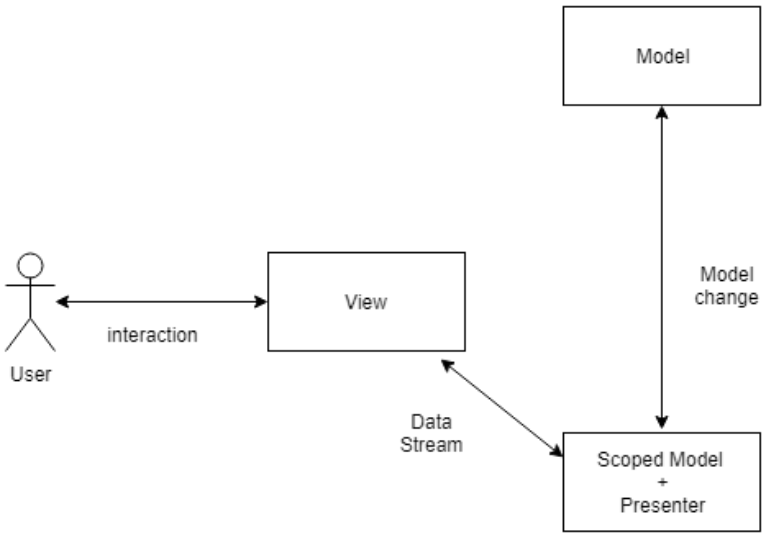


Figure 12 - Modèle de conception MVSP

Réalisation

- Composants métier

J'ai codé les différents modèles de l'application, c'est-à-dire les classes Category (**Figure 13**), Article.

```
class Category {
  int id;
  String nom;

  //Constructeur vide
  Category();

  //configurer catégorie grâce un map
  void fromMap(Map<String , dynamic> map) {
    this.id = map ['id'];
    this.nom = map['nom'];
  }

  //Convertir catégory en map
  Map<String, dynamic> toMap(){
   Map<String, dynamic> map = {
     'nom': this.nom
   };
   if (id != null){
     map['id'] = this.id;
   }
  }
}
```

Figure 13- Classe Category

```dart
class DatabaseClass {
  //déclaration en private de la variable _database de type Database
  Database _database;

  //fonction d'accès à la variable _database
  Future<Database> get database async {
    if  (_database != null){
      return _database;
    }else{
      //créer base de données
      _database = await create();
      return _database;
    }
  }

  // fonction creation de la base de données
  Future create() async{
    Directory directory = await getApplicationDocumentsDirectory();
    //récupération du chemin vers le dossier de mes documents et ajout de 'database.db'
    String database_directory = join(directory.path,'database.db');
    var bdd =  await openDatabase(database_directory, version: 1, onCreate: _onCreate);
    return bdd;
  }

  Future _onCreate(Database db, int version ) async {
    await db.execute('''
     CREATE TABLE category (
     id INTEGER PRIMARY KEY,
     nom TEXT NOT NULL)
     ''');
  }
```

Figure 14 - Script de création de la base de données

Dans ce même fichier, j'ai mis en place différence fonction pour pouvoir faire la lecture et l'écriture des données. La **figure 15** montre l'**ajout** d'une category.

```dart
/* Ecriture des données */
Future<Category> addCat(Category category) async{
  //vérifie base de donnée existe
  Database maDatabase = await database;
  //renvoi un int = id
  category.id = await maDatabase.insert('category',category.toMap());
  return category;
}
```

Figure 15 - Ajout d'une category

Pour faire la gestion de l'appareil photo et de la galerie photo, j'ai utilisé la librairie image_picker de flutter. (**Figure 16**)

```
  Future getImage(ImageSource source) async{
    var nouvelleImage = await ImagePicker.pickImage(source: source);
    setState(() {
      image = nouvelleImage.path;
    });
  }
}
```

Figure 16 - Utilisation de ImagePicker

Le code source est disponible sur GitLab sur le **https://gitlab.com/Emeline1995/lafleur**. J'ai rédigé la documentation sur la configuration et la création de l'Apk et son déploiement sur le PlayStore.

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Editeur de texte**

J'ai utilisé **Android Studio** qui est un environnement de développement pour développer des applications mobiles Android. Il intègre le langage **Java**, langage que nous avons décidé d'utiliser pour ce projet. De plus Android Studio met à disposition des émulateurs pour avoir un visuel de notre application.

- **Base de données**

J'ai utilisé comme système de gestion de base de données **SQFLite** (open source) qui incorpore une syntaxe **SQL**.

- **Outils collaboratifs et de gestion**

Ce projet a été réaliser seule, pour mettre en œuvre les bonnes pratiques à adopter.
J'ai utilisé :

- **GitLab** : qui a permis d'avoir une base de projet identique. Avec le système de branche, chacun a pu travail sur différentes fonctionnalités. De plus, le système de CI/CD (système d'intégration continue) réduit les marges d'erreur de l'application. Et nous permet de faire du versionning.

- **Documentations**

DART
Je me suis initiée à Dart avec : https://dart.dev/guides

FLUTTER
Pour la prise en main et son installation sous Android Studio : https://flutter.dev/docs

Pour mon **apprentissage personnel**, j'ai investi dans ce cours : https://www.udemy.com/flutter-dart-creez-des-applications-pour-ios-et-android/

3. Avec qui avez-vous travaillé ?

J'ai réalisé ce projet seule. C'est un sujet que j'ai abordé en BTS, que j'ai retravail pour répondre à cette activité. Et ainsi, m'apporter des connaissances personnelles sur les technologies Cross-Platform.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Dans le cadre de la formation : Technologie multi plateforme

Période d'exercice ▸ Du : 18/03/19 au : 29/03/19

5. Informations complémentaires *(facultatif)*

Cette technologie a été tout d'abord mise en pratique sur un projet réaliser en groupe. Qui s'intitule **PlayMusic** avec la collaboration de BATONNET Loic et LEGROS Richard.

Activité-type 3 Développer une application n-tiers

Exemple n°3 ▸ Développement application de gestion de prêt Livre et DVD – PLiDePA

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Cette application mobile cross plateforme a pour but de faire la gestion des emprunts de livres et de CD entre particuliers.

Pour mettre en place cette application, j'ai du tout d'abord rédiger les cahiers des charges.
À partir des spécifications fonctionnelles, telle que :

- L'utilisateur devra être authentifié pour accéder à l'application
- L'utilisateur pourra également se déconnecter de l'application
- L'utilisateur pourra ajouter le nom de la personne qui emprunte l'élément, quand il choisit de le prêter. Le nom apparaîtra également dans la page de liste correspondante et sera supprimé quand l'élément est rendu.
- ….

Pour répondre au critère de qualité logicielle (Interopérabilité, Validité, Fiabilité, Maintenabilité …), je me suis appuyée sur ce modèle d'architecture suivante :

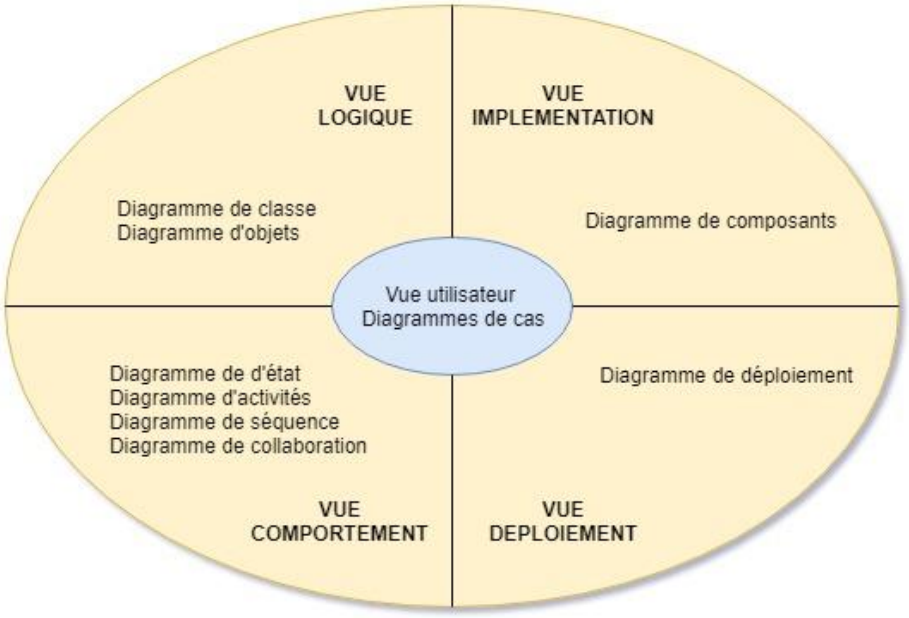


Figure 17 - Méthode de bonne pratique

Réalisation

☞ Système d'authentification

Pour mettre en place mon système d'authentification, j'ai installé dans mon projet avec NPM :

```
npm install firebase --save
```

J'ai jouté ensuite la configuration de Firebase dans le contructeur de AppComponent (en ajoutant import * as firebase from 'frirebase' ; en haut du fichier mettant à disposition la méthode initializeApp ()).

Cette application utilise l'authentification par adresse mail et mot de passe proposée par Firebase afin de sécuriser la base de données. Pour cela, j'ai activé dans la console de Firebase (**Figure 18**).

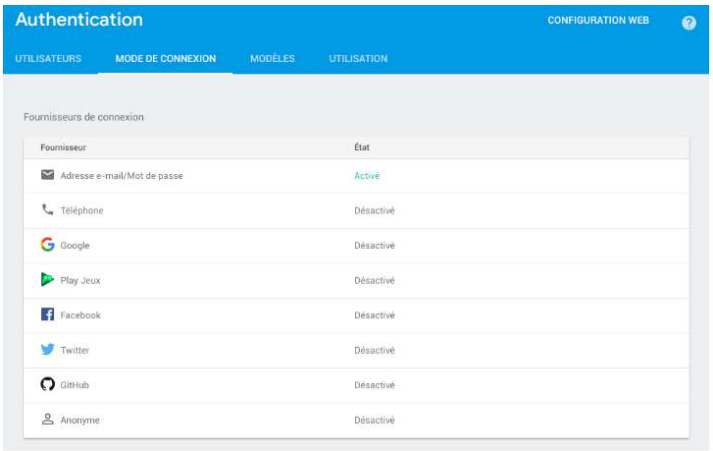


Figure 18 - Activation sur Firebase

J'ai créé après cela le service AuthService, qui comporte un Boolean lié à l'état d'authentification retourné par Firebase. J'ai en place ensuite, deux méthodes asynchrones l'un permettant l'inscription et l'autre pour la connexion d'un utilisateur existant. Et pour finir niveau authentification, la méthode de déconnexion de l'utilisateur.

```
signUpUser(email: string, password: string) {
  return new Promise((resolve, reject) => {
    firebase.auth().createUserWithEmailAndPassword(email, password).then(
      (user) => {
        resolve(user);
      },
      (error) => {
        reject(error);
      }
    );
  });
}

signInUser(email: string, password: string) {
  return new Promise((resolve, reject) => {
    firebase.auth().signInWithEmailAndPassword(email, password).then(
      (user) => {
        resolve(user);
      },
      (error) => {
        reject(error);
      }
    );
  });
}
```

Figure 19 - Fonctions inscription & connexion

☞ Enregistrement dans la base de données

Pour pouvoir réaliser cette partie, j'ai installé le package rxjs-compat qui évite de modifier les données directement mais modifier et observer les données de l'application.

```
npm install rxjs-compat  --save
```

J'ai créé pour cela un objet de type Subjet<>, je crée une méthode permettant d'émettre ce Subjet<>. Ensuite, j'ai mis en place l'enregistrement et la récupération des données. Pour cela j'utilise comme méthode database().ref('nom table ') ce qui crée une référence dans la base de données. Ensuite, pour l'enregistrement des données, la méthode set () permet d'écrire les données passées en argument.

☞ Native feature : appareil photo

Pour mettre en place cette fonctionnalité, j'ai installé les plugins Cordova et Ionic.

```
Ionic cordova plugin add cordova-plugin-camera
npm install—save @ionic-native/camera
```

Injection Camera dans **AppModule :**

```
providers: [
  StatusBar,
  SplashScreen,
  NatureViewService,
  Geolocation,
  Camera,
  {provide: ErrorHandler, useClass: IonicErrorHandler}
]
```

Déploiement

☞ Configuration de l'application

Pour configurer l'application, je me suis rendu dans le fichier config.xml. Ce fichier est généré automatiquement lors de la création du projet.
J'ai modifié l'id de l'application, ainsi que sa version. Ces éléments sont très importants lors de la signature de l'application pour le déploiement (deux applications ne peuvent pas avoir le même id, par exemple). C'est également ici que j'ai modifié le nom de l'application et sa description qui apparaîtra dans l'App Store ou le Play Store au moment de la mise en ligne de l'application. Je pouvais également modifier les informations en tant qu'auteur de l'application : mon adresse mail, mon site internet et mon nom.

Il y a beaucoup de possibilités de configuration dans config.xml. Pour en savoir plus afin de peaufiner la configuration, voici la documentation Cordova sur **cordova.apache.org.**

Et pour finir, j'ai build l'application

```
ioniccordovabuild—releaseios
ioniccordovabuild—releaseandroid
```

2. Précisez les moyens utilisés :

Les moyens utilisés pour ce projet :

- **Editeur de texte**

J'ai utilisé **ATOM** est un éditeur de texte libre pour macOS, GNU/Linux et Windows développé par GitHub.

- **Base de données**

J'ai utilisé comme système de gestion de base de données **Cloud Firestore** (outil intégré a **Firebase**[4]) qui incorpore le **langage NoSQL** basé sur le type documentaire.

- **Outils collaboratifs et de gestion**

Ce projet a été réaliser seule, pour mettre en œuvre les bonnes pratiques à adopter.
J'ai utilisé :

- **GitLab** : qui a permis d'avoir une base de projet identique. Avec le système de branche, chacun a pu travail sur différentes fonctionnalités. De plus, le système de CI/CD (système d'intégration continue) réduit les marges d'erreur de l'application. Et nous permet de faire du versionning.

- **Documentations**

ATOM
https://atom.io/docs

IONIC
Documentation officielle https://ionicframework.com/docs

FIREBASE
https://firebase.google.com/docs

[4] Est un ensemble de services d'hébergement pour n'impote quel type d'application. Il propose d'héberger en NoSQLet en temps réel des bases de données, de contenu, de l'authentification socialeet des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel.

3. Avec qui avez-vous travaillé ?

J'ai réalisé ce projet seule. C'est un sujet repris d'Openclassrom dans le but d'acquérir des connaissances personnelles sur les technologies Cross-Platform.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ EXPERNET

Chantier, atelier, service ▸ Dans le cadre de la formation : Technologie multi plateforme

Période d'exercice ▸ Du : 13/05/19 au : 17/05/19

5. Informations complémentaires *(facultatif)*

Titres, diplômes, CQP, attestations de formation

(Facultatif)

Intitulé	Autorité ou organisme	Date
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] .....Emeline KICHENIN..... ,

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l'auteur(e) des réalisations jointes.

Fait à .....LE PORT..... le .....24 /04 /2019.....
pour faire valoir ce que de droit.

Signature :

Documents illustrant la pratique professionnelle

(Facultatif)

Intitulé
ANNEXE 1 : DOCUMENTATION UTILISATEUR
ANNEXE 2 : DOCUMENTATION TECHNIQUE SUR LA CONFIGURATION ET GENERATION D'APK

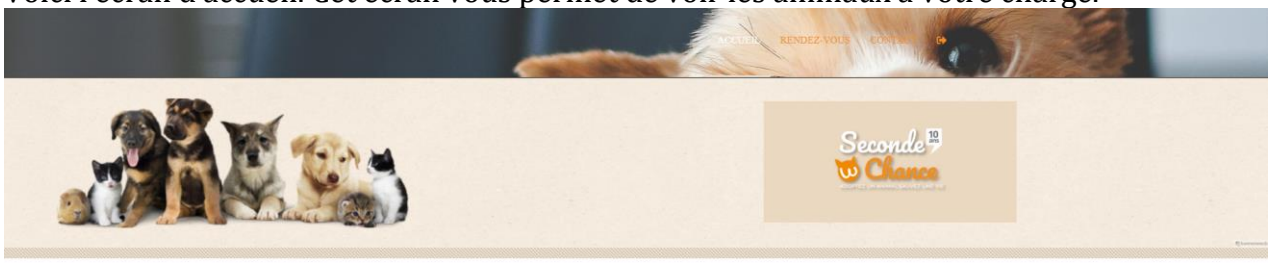
ANNEXES

(Si le RC le prévoit)

ANNEXE 1 : DOCUMENTATION UTILISATEUR

Authentifiez-vous, en renseignant votre login et mot de passe.

Voici l'écran d'accueil. Cet écran vous permet de voir les animaux à vôtre charge.



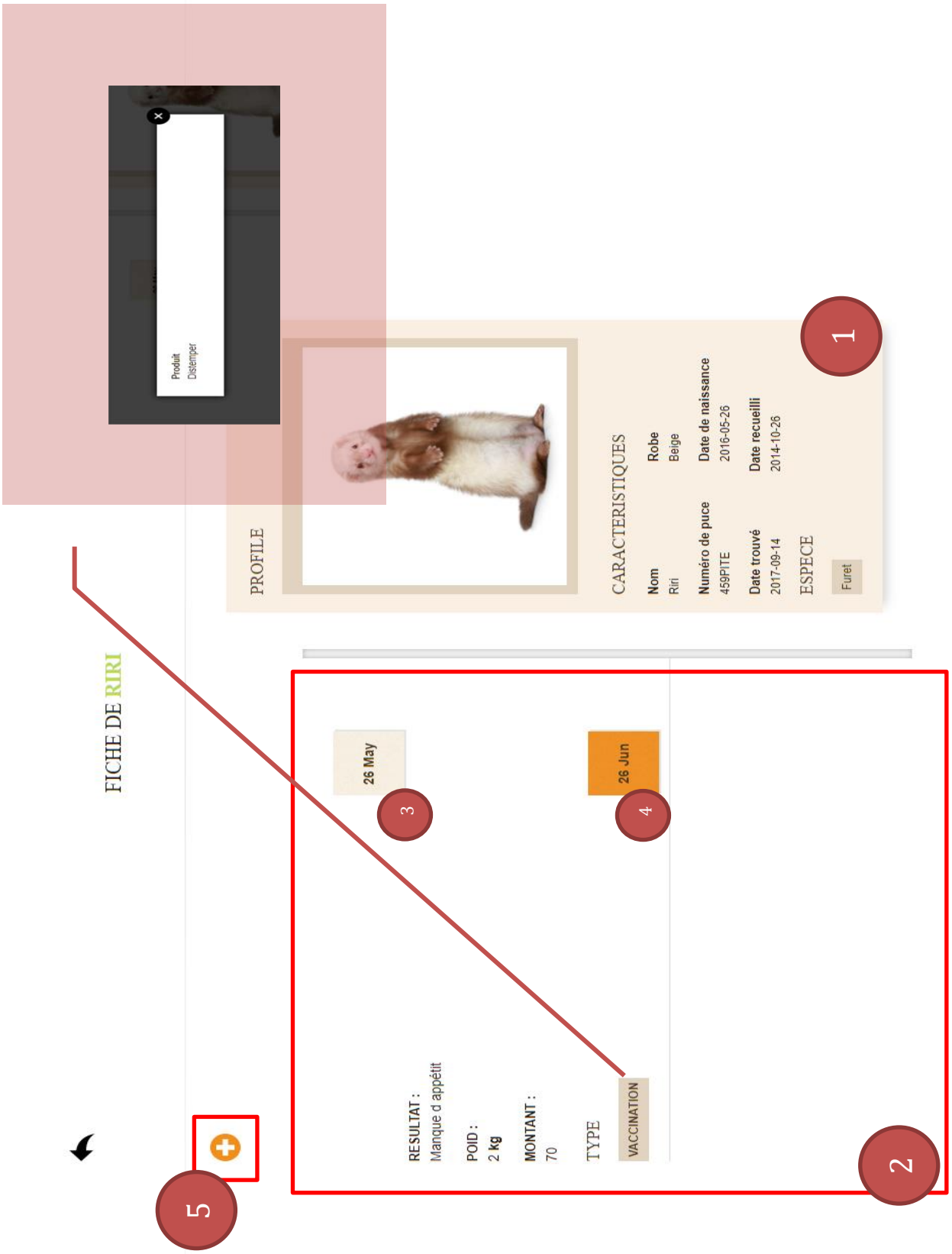
MES COMPAGNONS!

La bannière vous propose 4 actions, vous pouvez prendre un rendez-vous, contacté le centre, ou vous pouvez vous déconnecter.

Cliquez sur l'étiquette de l'animal que vous souhaitez renseigner un événement.

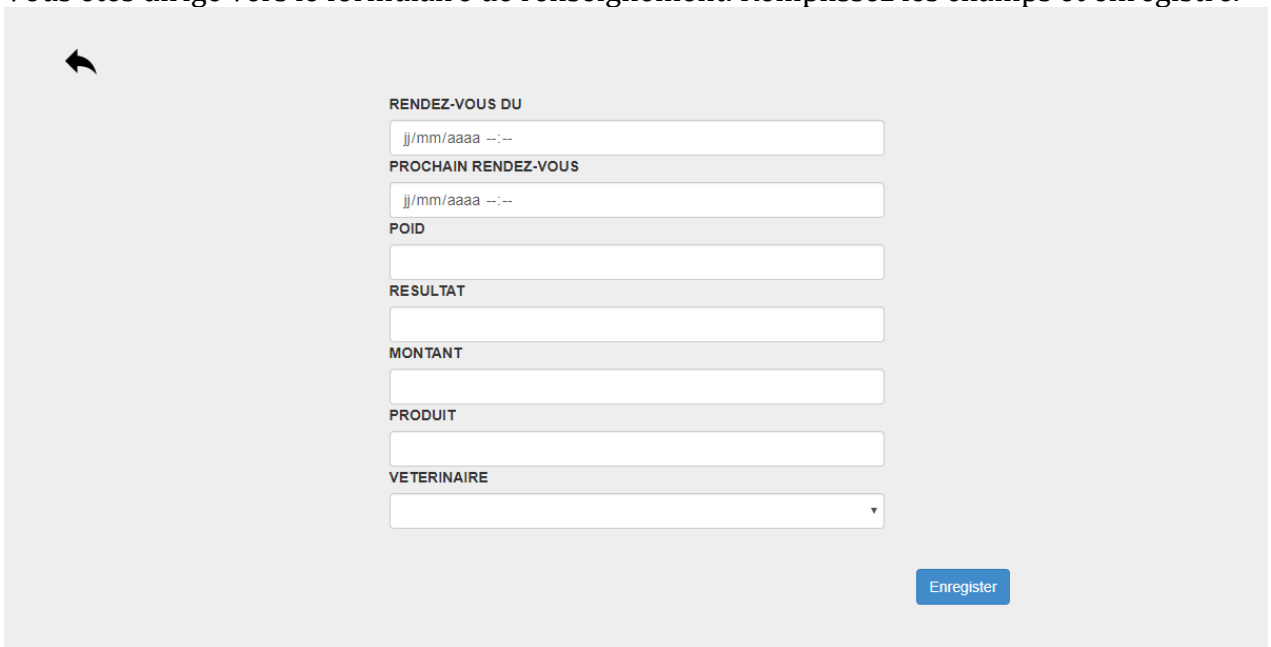
Vous pouvez voir sur cet écran :

- ☞ N°1 – Les détails concernant l'animal.
- ☞ N°2 : Les anciens renseignement que vous avez noté lors d'une vaccination chez le vétérinaire.
- ☞ En n°3, c'est la date auxquelles la vaccination a été faite, en n°4 c'est le prochain rendez-vous chez le vétérinaire.
- ☞ En n°5, le bouton « + », permet d'ajouter un événement.
- ☞ En cliquant sur le type d'évènement, cela déclenche une pop -up contenant les détails de l'évènement.


Produit
Distemper
PROFILE
CARACTERISTIQUES
Nom
Riri
Robe
Beige
Numéro de puce
459PITE
Date de naissance
2016-05-26
Date trouvé
2017-09-14
Date recueilli
2014-10-26
ESPECE
Furet
1
FICHE DE RIRI
26 May
3
26 Jun
4
RESULTAT :
Manque d appétit
POID :
2 kg
MONTANT :
70
TYPE
VACCINATION
2
5

Cliquez sur le bouton « + », choisissez le type d'événement que vous voulez renseigner.

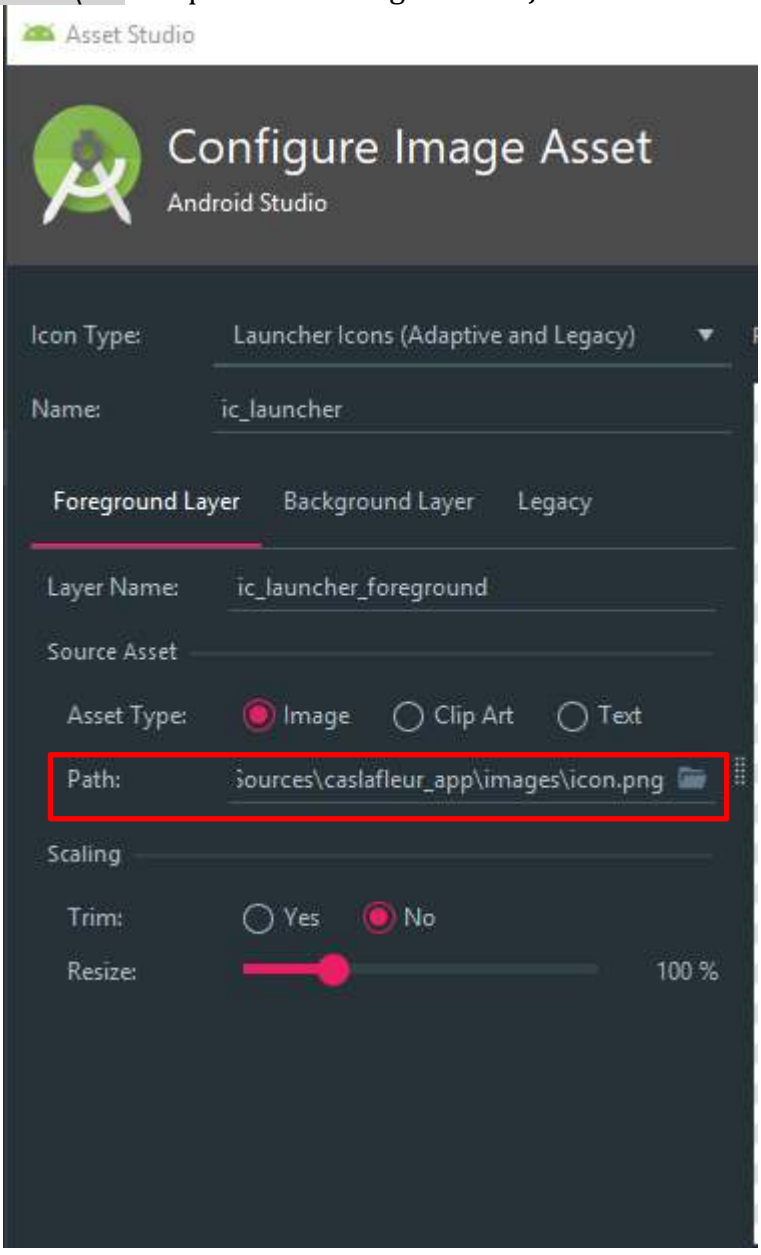
Vous êtes dirigé vers le formulaire de renseignement. Remplissez les champs et enregistré.



ANNEXE 2 : DOCUMENTATION TECHNIQUE SUR LA CONFIGURATION ET GENERATION D'APK

Ajout d'une icône sur le lanceur

L'application possède par défaut une icône. Pour changer cette icône, placez – vous android\app\src\main\res : clique droit > Image Assert. Joindre votre icône.



Signature de l'application

Pour publier sur PlayStore, vous devez attribuer une signature numérique à l'application. Si vous avez une clé existante, passez à l'étape suivante. Sinon, créez-en vous un en lançant la commande suivante dans une invite de commande (Ne pas oubliez d'installer un JDK au préalablement) :

```
keytool -genkey -v -keystore ~/key.jks -keyalg RSA -keysize 2048 -validity
10000 -alias key
```

Créez un fichier nommé key.properties,à la racine du dossier android. Renseignez le mot de passe que vous avez choisi pour la création de la clé à l'étape précédente :

```
storePassword=<password from previous step>
keyPassword=<password from previous step>
keyAlias=key
storeFile=<location of the key store file, such as /Users/<user name>/key.jks>
```

Modifier le fichier android/app/build.gradle

1. Remplacez l'élément suivant :

```
android {
```

Par les information de la clé:

```
def keystoreProperties = new Properties()
def keystorePropertiesFile = rootProject.file('key.properties')
if (keystorePropertiesFile.exists()) {
    keystoreProperties.load(new FileInputStream(keystorePropertiesFile))
}

android {
```

2. Remplacez l'élément suivant :

```
buildTypes {
    release {
        // TODO: Add your own signing config for the release build.
        // Signing with the debug keys for now,
        // so `flutter run --release` works.
        signingConfig signingConfigs.debug
    }
}
```

Par les informations de configuration de la signature :

```
signingConfigs {
    release {
        keyAlias keystoreProperties['keyAlias']
        keyPassword keystoreProperties['keyPassword']
        storeFile file(keystoreProperties['storeFile'])
        storePassword keystoreProperties['storePassword']
    }
}
buildTypes {
    release {
        signingConfig signingConfigs.release
    }
}
```

Construire l'APK en lançant la commande :

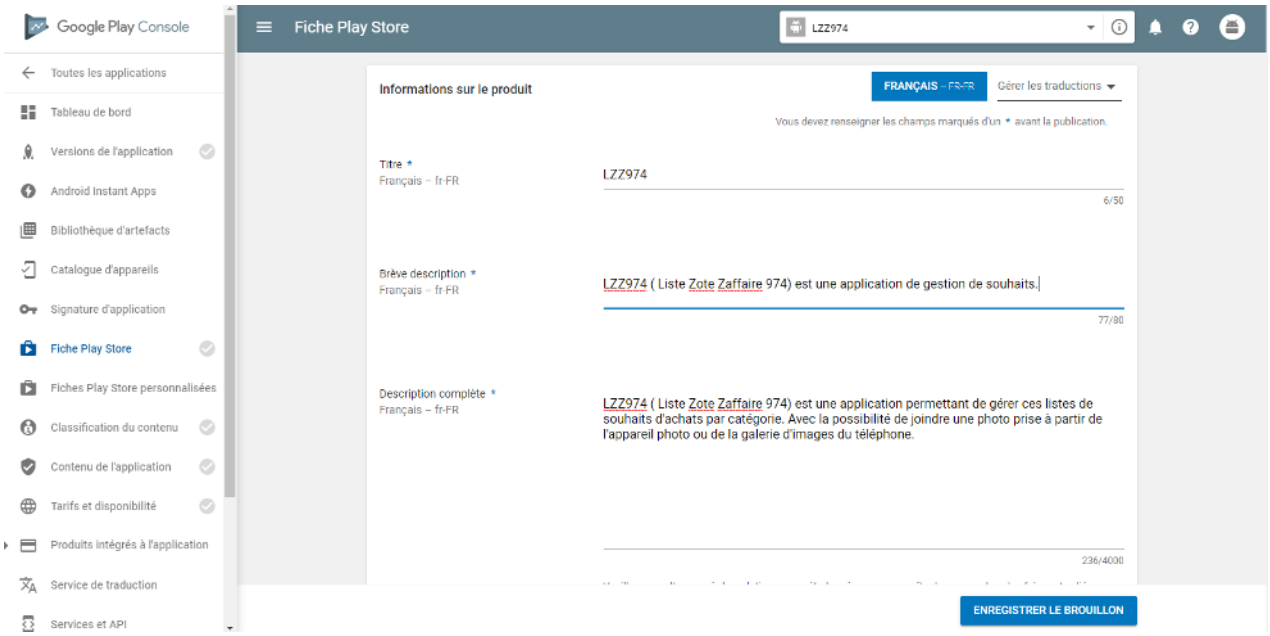
```
flutter build apk
```

```
C:\Users\asus\Documents\Projets\lafleur>flutter build apk
You are building a fat APK that includes binaries for android-arm, android-arm64.
If you are deploying the app to the Play Store, it's recommended to use app bundles or split the APK to reduce the APK size.
    To generate an app bundle, run:
        flutter build appbundle --target-platform android-arm,android-arm64
        Learn more on: https://developer.android.com/guide/app-bundle
    To split the APKs per ABI, run:
        flutter build apk --target-platform android-arm,android-arm64 --split-per-abi
        Learn more on:  https://developer.android.com/studio/build/configure-apk-splits#configure-abi-split
Initializing gradle...                                    2,8s
Resolving dependencies...                                 14,7s
Running Gradle task 'assembleRelease'...
\asus\.gradle\caches\transforms-1\files-1.1\android.jar\1183ee6359e739aedf395bf7ff7ab2f2\android.jar with input C:\Users\asus\
Calling mockable JAR artifact transform to create file: C:\Users\asus\.gradle\caches\transforms-1\files-1.1\android.jar\ad6e58
platforms\android-27\android.jar
Running Gradle task 'assembleRelease'...
Running Gradle task 'assembleRelease'... Done                     299,7s (!)
Built build\app\outputs\apk\release\app-release.apk (12.1MB).
```

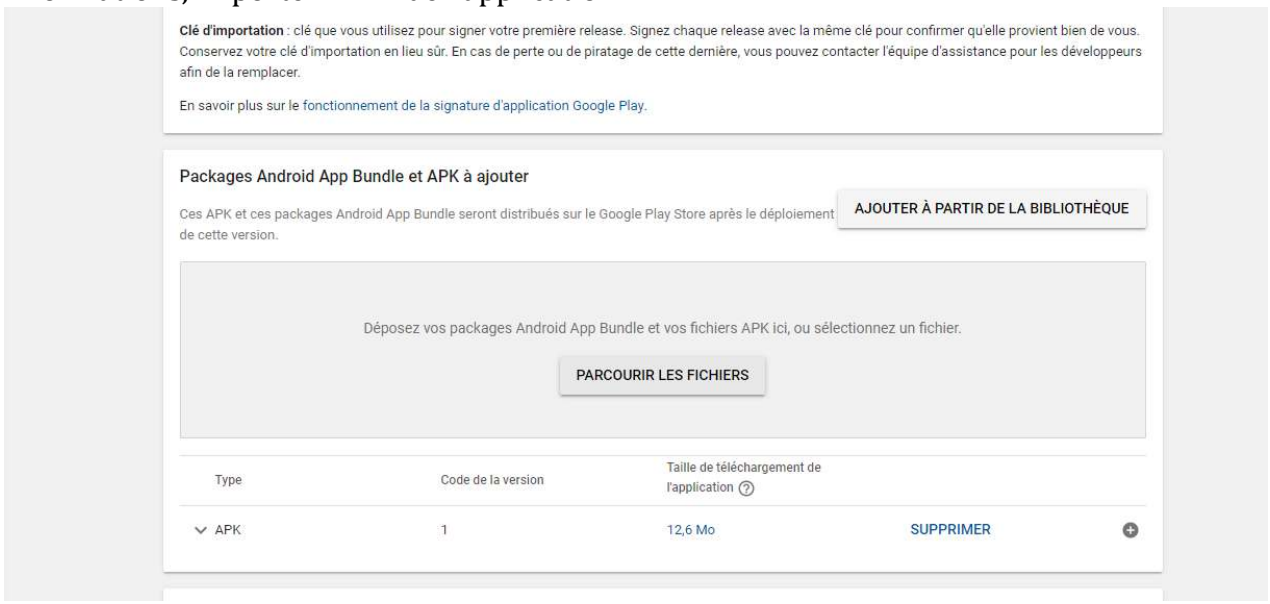
Dans la console de PlayStore

Créer un nouveau projet, remplissez les informations des rubriques suivantes :

- Fiche Play Store,
- Classification du contenu,
- Contenu de l'application,
- Tarifs et disponibilité



Remplissez en dernier, la rubrique « Versions de l'application ». Renseigner les informations, importer l'APK de l'application.



Lorsque les rubriques sont validées, lancer le déploiement de l'application par le biais du bouton déployer.