

# Documentation de l'application LoveMyPet

# Table des matières

Partie 1 : Synthèse .....	1
titre du projet .....	1
Abréviation .....	1
Logo .....	1
Membres de l'équipe .....	1
Présentation .....	1
Partie 2 : Aspects techniques .....	5
type d'application .....	5
Schéma architectural de l'application .....	5
Documentation Complète de l'Application .....	10
Partie 1 : Modélisation .....	11
Petite feature pour l'affichage et la modification des informations des Animaux .....	13
Fonctionnalité de Suivi des Vaccins - Carnet de Vaccination en Ligne .....	13
Fonctionnalité de recherche des Vétérinaires et Jardins à Proximité .....	14
Utilisation de la Carte depuis la Page "Mes Services" .....	15
Accès à la Page "Mes Services" .....	15
Option "Trouver un Service" .....	15
Choix entre Vétérinaires et Jardins .....	15
Requête vers l'API Overpass et Nominatim .....	17
Trouver un Vétérinaire : .....	17
Affichage des Résultats sur la Carte .....	18
Recherche par ville "Geneve": .....	18
Explication de l'API Overpass et Nominatim .....	19
Diagramme de class .....	19
Paramètres de Requête .....	21
Exemple de Réponse .....	22
Explication de l'API Overpass et Nominatim .....	22
Fonctionnalité Interaction avec les Avis (Petite) .....	23
Diagramme de séquence .....	23
Fonctionnalité de Planification des Heures de Repas et Envoi de Mails et confirmation du mail .....	25
Planification des Heures de Repas et Envoi de Mails .....	25
Envoi de Mails de Rappel .....	25
Confirmation de l'E-mail .....	26
API de la fonctionnalité .....	27
Fonctionnalité de donation d'objets .....	31
API de la fonctionnalité de donation d'objets .....	33
Diagramme de séquence .....	36
Fonctionnalité de Création et Inscription aux Événements sur LoveMyPet .....	37

Présentation .....	38
Utilisation depuis la Page "Mes Services".....	39
Ajout d'un Événement .....	39
Affichage des Événements Non Expirés.....	40
Inscription à un Événement .....	40
Affichage des Événements Crées par un Utilisateur.....	40
Suppression d'un Événement Crée .....	41
Diagramme de Classe .....	42
API .....	43
Ajout d'Événement .....	43
Affichage des Événements Non Expirés.....	43
Détails d'un Événement Non Expiré .....	43
Liste des Événements Crées par un Utilisateur .....	44
Inscription à un Événement .....	44
Diagramme de Séquence - Ajout d'Événement.....	45
Diagramme de Séquence - Affichage des Événements Non Expirés .....	46
Diagramme de Séquence - Inscription à un Événement .....	47

# Partie 1 : Synthèse

## titre du projet

Le titre qu'on a choisi pour notre projet est **LoveMyPet**.

## Abréviation

L'abréviation qu'on a choisi pour notre projet est **LMP**.

## Logo

Pour le logo du projet, on a crée le logo suivant :



LOVEMYPET

## Membres de l'équipe

Le projet LoveMyPet a été réalisé par les membres suivants :

Imane Errahmani

Adenle Sadikou

Malek Messaoudi

## Présentation

### Contexte

Les animaux de compagnie apportent la joie et le soutien émotionnel, cependant, leur adoption peut être un processus très complexe et parfois décourageant, tant pour les futurs adoptants que

pour les animaux en attente de foyer. Faciliter l'adoption d'animaux de compagnie n'est pas seulement une question de confort, c'est une opportunité pour sauver des vies et créer des connexions durables entre les animaux et leurs propriétaires.

## Problématique

Le processus d'adoption des animaux de compagnie présente actuellement des défis significatifs tant pour les donneurs que pour les adoptants. Les principales problématiques identifiées sont les suivantes :

**Complexité du Processus:** Les plateformes existantes se concentrent principalement sur la mise en relation avec des refuges ou des associations, rendant le processus d'adoption complexe et bureaucratique.

**Suivi du Bien-Être des Animaux:** Après l'adoption, il existe un manque de mécanismes efficaces pour assurer le suivi du bien-être des animaux, ce qui peut entraîner une négligence involontaire.

**Manque de Flexibilité:** Les plateformes actuelles ne permettent pas aux donneurs de spécifier la durée de disponibilité de l'animal, limitant ainsi les options d'adoption temporaire.

## Gain attendu

En abordant ces problématiques, notre projet, LoveMyPet, vise à apporter les améliorations suivantes :

**Simplification du Processus d'Adoption:** Offrir une plateforme conviviale où les donneurs peuvent directement mettre en contact des personnes désireuses d'adopter, simplifiant ainsi le processus.

**Suivi Continu du Bien-Être:** Introduire un système de suivi du bien-être des animaux, incluant des rappels pour les vaccinations et d'autres aspects cruciaux, pour garantir une vie épanouissante après l'adoption.

**Flexibilité dans l'Adoption:** Permettre aux donneurs de spécifier la durée de disponibilité de l'animal, offrant ainsi des options d'adoption temporaire et permanente, selon les préférences des utilisateurs.

## Motivation de l'équipe par rapport au sujet

La motivation principale de notre projet est la volonté de simplifier le processus de l'adoption des animaux de compagnie et d'améliorer leur bien-être. On vise à créer une plateforme dont l'objectif est de connecter les adoptants potentiels avec les gens qui souhaitent donner leur animal de compagnie soit pour une durée définie (vacances) ou pour toujours.

Notre projet vise aussi à éduquer les futurs propriétaires sur les meilleures pratiques de soins afin de créer une communauté engagée et responsable pour contribuer à une adoption plus répandue et à des vies animales plus épanouies.

## Concurrence

Afin de faire une étude de la concurrence, on s'est posé les questions suivantes :

Qui sont nos concurrents ? Où sont-ils ? Que proposent-ils ? Quelles sont leurs forces et leurs faiblesses ?

Après une recherche sur internet, on a vu qu'en France, il existe beaucoup de sites web permettant l'adoption des animaux que ce soit des sites des fondations (Fondation 30 Millions d'amis, Fondation Brigitte Bardot...) ou des plateformes d'adoption comme Seconde Chance, La-Spa.fr, PAAW...

On a constaté que sur ces sites web, le service proposé est de mettre en contact un futur adoptant avec une association de la protection des animaux ou avec un refuge, cela est totalement différent du service proposé par notre projet qui est de mettre en contact deux personnes, une qui souhaite donner son animal de compagnie soit de façon définitive ou pour une durée précise (vacance) et l'autre qui représente un futur propriétaire de cet animal.

En revanche ce service peut être existant dans des sites comme LeBonCoin, sauf que dans ce cas, ce site n'est pas dédié seulement aux animaux et leur adoption ce qui n'est pas pratique pour les utilisateurs souhaitant profiter d'un processus d'adoption simple et efficace.

Ce qui diffère notre projet des concurrents cités, c'est le fait que c'est un site qui permet un suivi du bien être des animaux, cela inclut : Trouver un propriétaire pour l'animal, Assurer que le propriétaire prend soin de l'animal en lui envoyant des rappels pour nourrir l'animal, le laver, le vacciner...

Donc au final, notre projet est une combinaison de fonctionnalités permettant une meilleure expérience utilisateur.

## Personas

### Donneur d'Animal - Sophie:



Contexte: Sophie, 35 ans, a un chien nommé Max qu'elle aime profondément. Cependant, en raison de changements de vie, elle doit trouver un nouveau foyer pour Max.

Fonctionnalités Clés:

Enregistrement facile des informations de Max sur la plateforme.

Possibilité de donner Max pour une durée à spécifier (temporaire ou permanente).

#### Futur Adoptant - Antoine:



Contexte: Antoine, 28 ans, cherche un compagnon animal. Il aimerait adopter un chien pour lui tenir compagnie.

#### Fonctionnalités Clés:

Recherche simplifiée d'animaux disponibles à l'adoption.

Faire une candidature pour exprimer son intérêt à adopter.

Accès à des informations complètes sur les vaccinations et le suivi du bien-être de l'animal.

## Prévisions Marketing

#### Réseaux Sociaux:

Stratégie: Campagnes engageantes sur des plateformes populaires telles que Facebook, Instagram et Twitter. Contenu: Témoignages d'adoptions réussies, conseils de soins, et mises en avant des fonctionnalités uniques de LoveMyPet. Impact Attendu: Augmentation de la notoriété de la plateforme, engagement de la communauté, partages sociaux.

#### Site Web:

Stratégie: Développement d'un site web attrayant et convivial. Contenu: Histoires inspirantes, guides de soins, témoignages d'utilisateurs, et accès facile aux fonctionnalités de la plateforme. Impact Attendu: Plateforme centrale pour les informations sur LoveMyPet, conversion des visiteurs en utilisateurs actifs.

#### Campagnes d'Influenceurs:

Stratégie: Partenariat avec des influenceurs dans le domaine des animaux et de l'adoption responsable. Contenu: Contenu authentique mettant en avant l'utilité de LoveMyPet. Impact Attendu: Atteinte d'un public plus large, renforcement de la confiance grâce à des recommandations d'influenceurs.

## Partie 2 : Aspects techniques

### type d'application

LoveMyPet est une application **web**

### Schéma architectural de l'application

Voici notre schéma architectural



et voici toutes les API utilisées dans notre projet :

### AdoptionController

Point de départ de l'API : [/adoption](#)

- **GET /adoptions** : Obtenir la liste de toutes les URL d'adoption.

- **GET /{idAdoption}** : Obtenir les détails d'une adoption spécifique par ID.

## **AnimalController**

Point de départ de l'API : **/animal**

- **POST /add** : Ajouter un nouvel animal avec un fichier image.
- **GET /person/{idPerson}** : Obtenir la liste des références d'animaux par ID de personne.
- **GET /{id}** : Obtenir les détails d'un animal spécifique par ID.
- **GET /{animalId}/candidatures** : Obtenir la liste des candidatures pour un animal spécifique.
- **POST /addadoption** : Ajouter une nouvelle adoption.

## **CandidatureController**

Point de départ de l'API : **/animal/{animalId}/candidature**

- **GET /{candidatureId}** : Obtenir les détails d'une candidature spécifique pour un animal.

## **MesCandidatureController**

Point de départ de l'API : **/mescandidature**

- **GET /person/{idPerson}** : Obtenir la liste des références de candidatures par ID de personne.
- **GET /{idCandidature}** : Obtenir les détails d'une candidature spécifique par ID.

## **PersonController**

Point de départ de l'API : **/person**

- **GET /{id}** : Obtenir les détails d'une personne spécifique par ID.
- **POST /add** : Ajouter une nouvelle personne avec un fichier image.
- **POST /login** : Authentifier une personne à l'aide de l'e-mail et du mot de passe.
- **GET /profile** : Obtenir les informations du profil utilisateur.
- **POST /addcandidature** : Ajouter une nouvelle candidature.

## **VaccinationController**

Point de départ de l'API : **/vaccination**

- **GET /animal/{idAnimal}** : Obtenir les références de vaccination par ID d'animal.
- **GET /{idVaccination}** : Obtenir les détails d'une vaccination spécifique par ID.
- **POST /add** : Ajouter une nouvelle vaccination.

## **VaccinController**

Point de départ de l'API : **/vaccin**

- **GET /all** : Obtenir toutes les vaccinations.

## HistoriqueAdoptionController

Point de départ de l'API : [/historiqueadoption](#)

- **POST /ajouteradoption** : ajouter une adoption temporaire dans la table.

## HistoriqueWeightController

Point de départ de l'API : [/historiqueWeight](#)

- **GET /animal/{idAnimal}** : Récupérer la liste des liens d'historique de poids pour un animal donné. **\*GET /{id}** : Récupérer les détails d'un enregistrement d'historique de poids par son identifiant. **\*GET /data/{idAnimal}** : Récupérer les données d'historique de poids pour un animal donné.

## InfoAnimalController

Point de départ de l'API : [/infoanimal](#)

- **GET /{idAnimal}** : Récupérer les informations concernant un animal à partir de son id **\*POST /updateName/{idAnimal}** : modifier le nom de l'animal dont l'identifiant est idAnimal. **\*POST /updateWeight/{idAnimal}** : modifier le poids de l'animal dont l'identifiant est idAnimal. **\*POST /updateImage/{idAnimal}** : modifier l'image de l'animal dont l'identifiant est idAnimal.

## MesCandidatureController

Point de départ de l'API : [/mescandidature](#)

- **GET /person/{idPerson}** : Récupérer les liens des candidatures de la personne. **\*GET /{idCandidature}** : Récupérer la candidature.

## plateforme technologique

### Langages utilisés

**Backend (Java)**: Utilisation de Java pour la logique métier, la gestion de la base de données, et la création de l'API REST.

**Frontend (JavaScript, HTML, JSON)**: Utilisation de JavaScript pour la logique côté client, HTML pour la structure des pages, et JSON pour le format des données échangées.

### API REST

Utilisation d'une architecture RESTful pour la communication entre le frontend et le backend.

### Frameworks de Test

**Jacoco**: Utilisation de Jacoco pour la mesure de la couverture de code, permettant d'évaluer l'étendue des tests effectués sur le code source Java.

## **Gestion de Versions**

**Git:** Utilisation du système de gestion de versions Git pour le suivi des modifications, la collaboration entre les membres de l'équipe, et la gestion des branches de développement.

## **Build**

**Maven:** Utilisation de Maven pour la gestion des dépendances, la compilation du code source, et la création d'artefacts binaires.

## **Intégration Continue (CI)**

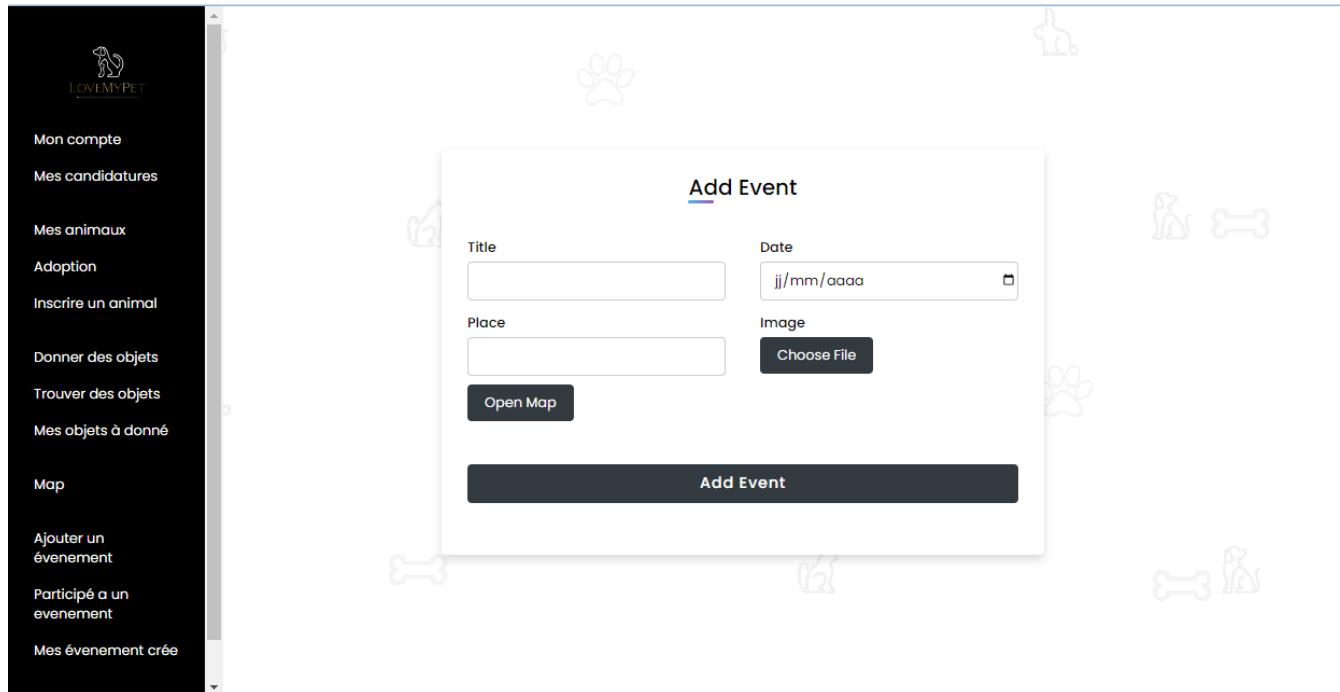
**GitAction:** Mise en place d'un système d'intégration continue pour automatiser les tests, la compilation, et la vérification de la qualité du code à chaque modification dans le référentiel Git.

# **Documentation Complète de l'Application**

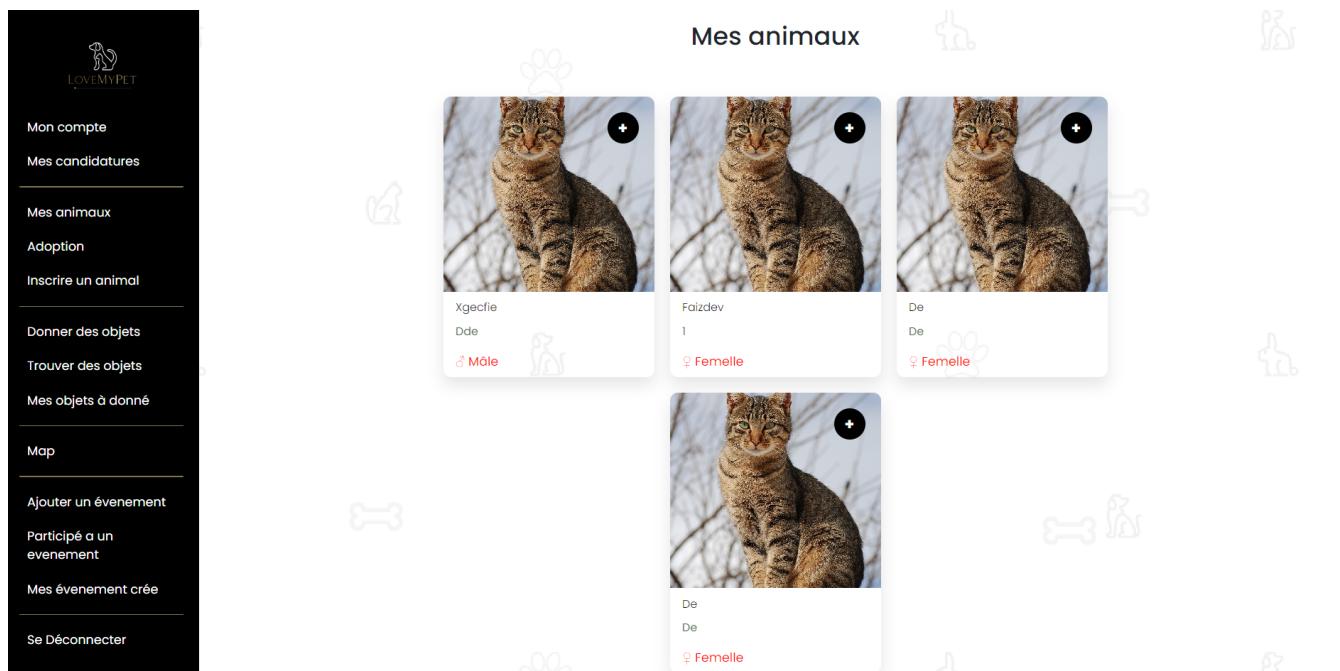
# Partie 1 : Modélisation

Après avoir ajouté des compagnons dans la section "Mes Animaux", chaque animal est associé au bouton unique "Donner". Cela déclenche un pop-up interactif pour faciliter l'adoption.

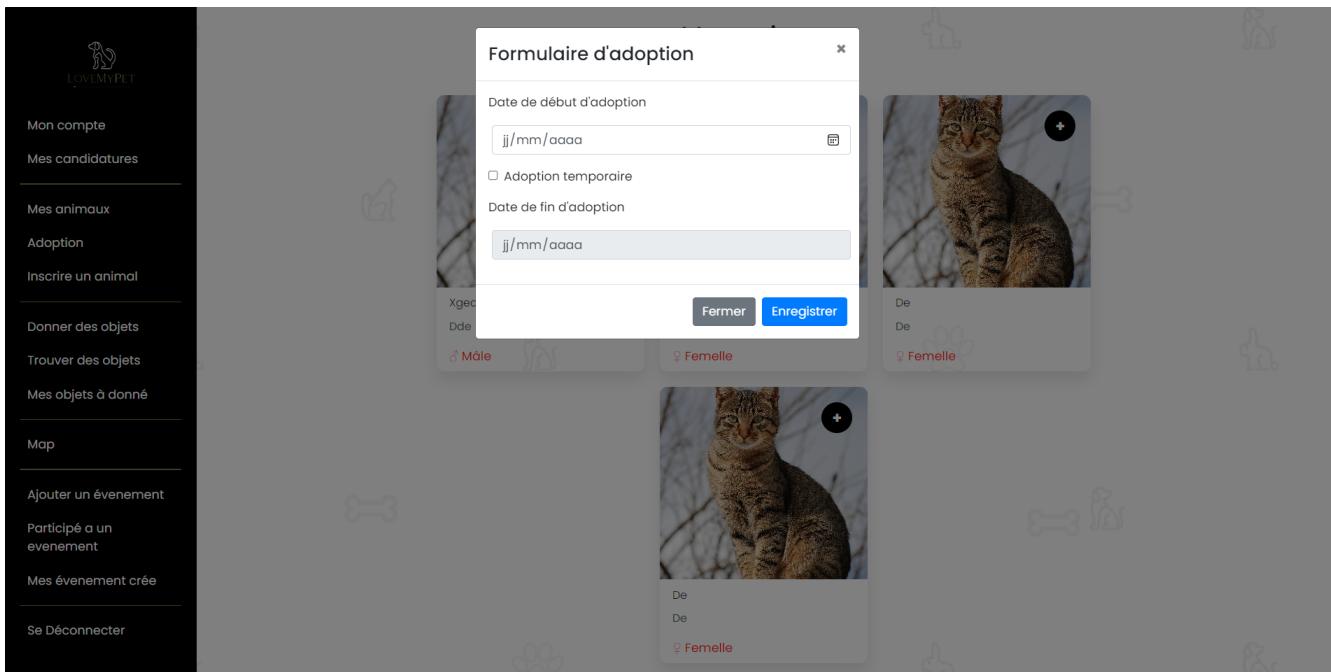
## étape 1



## étape 2



## étape 3



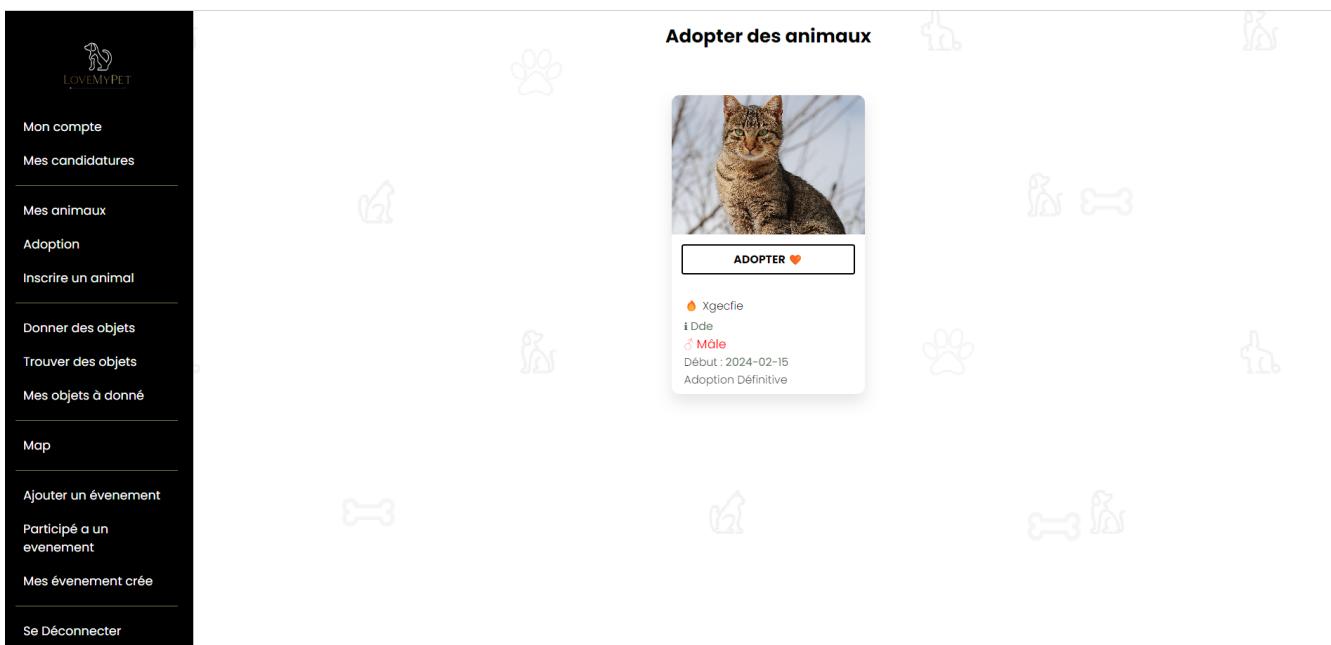
## Configuration des Dates pour l'Adoption

Dans le pop-up, la date de début est obligatoire, indiquant quand l'animal sera disponible pour adoption. La date de fin est facultative, laissant aux donateurs le choix de l'adoption permanente ou temporaire, idéal pour les périodes comme les vacances.

## Affichage dans la Section "Adoption"

Une fois configuré, l'animal est répertorié dans la section dédiée à l'adoption, avec un bouton "Candidater" pour exprimer l'intérêt.

### étape 4



Cette approche flexible simplifie l'adoption, laissant aux propriétaires le choix de la durée d'adoption.

## Gestion des Candidatures

Chaque animal ajouté à la section "Mes Animaux" est équipé d'un bouton "Candidatures". Ce bouton permet au propriétaire de consulter les détails des personnes ayant postulé pour garder son animal.

### étape 5

Le propriétaire peut ensuite examiner chaque candidature individuellement et décider d'accepter ou de refuser.

### étape 6

## Petite feature pour l'affichage et la modification des informations des Animaux

Chaque animal dans la section "Mes Animaux" est doté d'un bouton en haut à droite avec le signe plus (+). En cliquant sur ce bouton, les informations détaillées de l'animal sont affichées, offrant la possibilité de modifier le nom, le poids et la photo de l'animal.

## Fonctionnalité de Suivi des Vaccins - Carnet de Vaccination en Ligne

La fonction de suivi des vaccins offre une gestion centralisée des vaccinations des animaux. Accessible depuis "Mes Animaux", elle permet aux propriétaires de maintenir un carnet de vaccination en ligne.

### Accès Simple

étape 1 image::suivi\_vaccins.png[étape 1]

étape 2 image::mes\_animaux.png[étape 2]

### Vue d'Ensemble des Vaccinations

L'interface propose une vue d'ensemble détaillée des vaccinations, incluant le nom et la date de chaque vaccin.

### Ajout de Vaccinations

Les propriétaires peuvent ajouter de nouvelles vaccinations à tout moment, garantissant un historique complet et à jour.

## **Fonctionnalité de recherche des Vétérinaires et Jardins à Proximité**

L'application offre une fonctionnalité avancée permettant aux utilisateurs de localiser rapidement des vétérinaires ou des jardins à proximité de leur emplacement actuel. Cette fonctionnalité est accessible depuis la page "Mes Services".

# Utilisation de la Carte depuis la Page "Mes Services"

## Accès à la Page "Mes Services"

Connectez-vous à votre compte sur le site et accédez à la page "Mes Services".

## Option "Trouver un Service"

Recherchez l'option "EXPLORATION FACILE" sur la page "Mes Services" et cliquez dessus.

Bonjour null null !

## Nos Services

Découvrez nos services dédiés à rendre la vie avec votre compagnon encore plus spéciale.

**ADOPTER**  
Offrez à un compagnon à quatre pattes une nouvelle maison pleine d'amour et d'aventures en adoptant l'un de nos adorables amis poilus avec LoveMyPet.  
**Adopter maintenant**

**Inscrire mon animal**  
Inscrivez votre Animal et restez connecté à chaque étape de la vie de votre compagnon avec nos notifications dédiées. Soyez toujours au courant de ses besoins essentiels.  
**Scrire mon animal**

**SUIVI COMPLET DE L'ANIMAL**  
Assurez un suivi complet de la santé de votre compagnon, de son poids à ses vaccinations, pour une vie épanouissante.  
**Suivi complet**

**EXPLORATION FACILE**  
Notre carte interactive vous guide facilement vers les vétérinaires et parcs locaux, transformant chaque sortie en une aventure agréable pour vous et votre compagnon.  
**Explorer**

## Choix entre Vétérinaires et Jardins

Une fois sur la page "EXPLORATION FACILE", les utilisateurs ont deux choix : - "Trouver un Vétérinaire" - "Rechercher un Jardin pour se Balader" - "Recherche par Ville et Périmètre"

Les utilisateurs peuvent sélectionner l'option qui correspond à leur recherche.

**LOVEMYPET**

Mon compte

Mes candidatures

Mes animaux

Adoption

Inscrire un animal

Donner des objets

Trouver des objets

Mes objets à donné

Map

Ajouter un événement

Participé a un evenement

Mes événement crée

Se Déconnecter

Trouvez votre vétérinaire et des parcs avec LoveMyPet

Découvrez les vétérinaires et les parcs les plus proches de chez vous grâce à notre application LoveMyPet.

Trouver un vétérinaire      Trouver un parc

# Requête vers l'API Overpass et Nominatim

## Trouver un Vétérinaire :

Lorsque l'utilisateur choisit "Trouver un Vétérinaire", l'application utilise l'API Overpass pour rechercher les vétérinaires à proximité de la position actuelle de l'utilisateur. La requête à l'API Overpass est générée dynamiquement pour récupérer les nœuds correspondant à la catégorie "amenity=veterinary" dans un rayon défini autour de la position de l'utilisateur.

En outre, l'application utilise l'API Nominatim pour convertir le nom de la ville saisi par l'utilisateur en coordonnées géographiques.

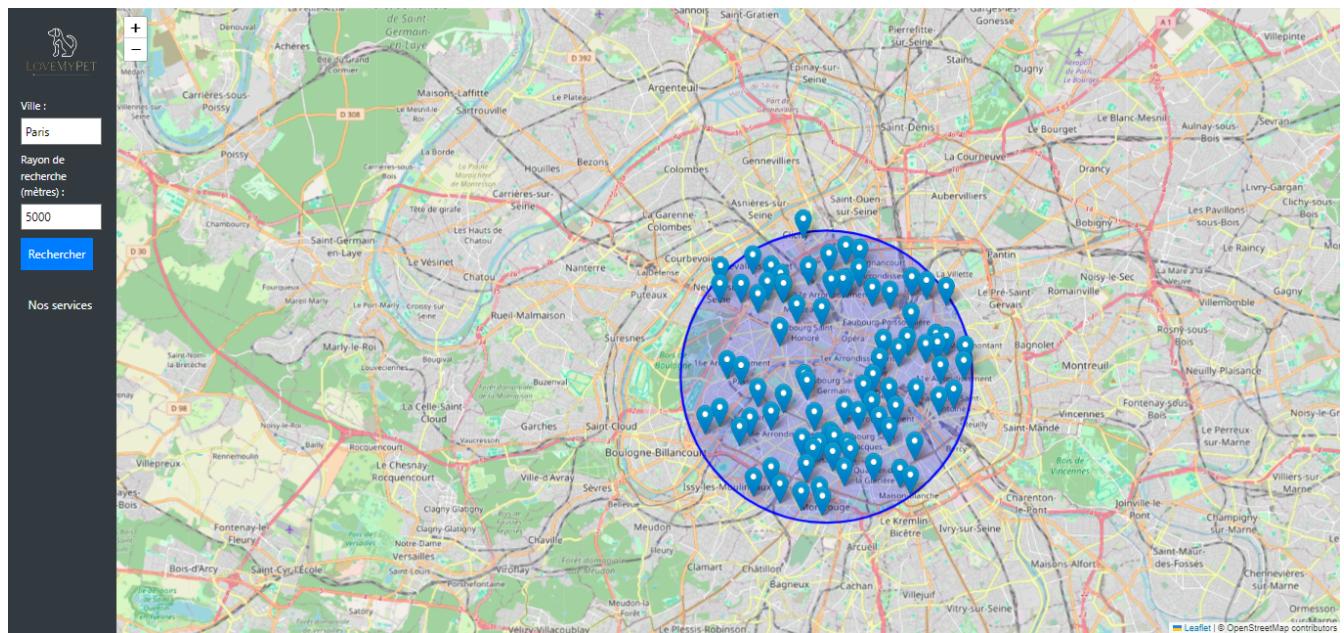
- === Rechercher un Jardin pour se Balader :

Si l'utilisateur opte pour "Rechercher un Jardin pour se Balader", l'application effectue une requête pour trouver les nœuds correspondant à la catégorie "leisure=garden" autour de la position actuelle de l'utilisateur. Cette requête est également générée dynamiquement pour récupérer les informations nécessaires.

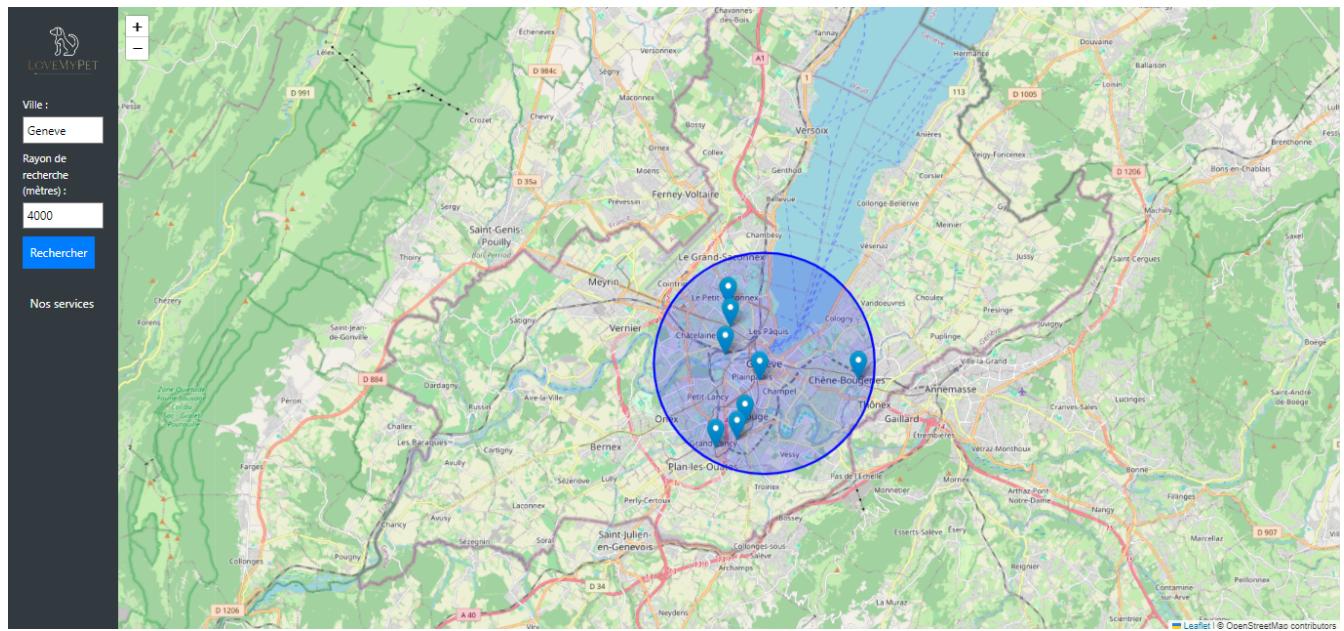
# Affichage des Résultats sur la Carte

- Trouver un Vétérinaire :

Les résultats de la requête pour les vétérinaires sont affichés sur la carte sous forme de marqueurs. Chaque marqueur représente l'emplacement d'un vétérinaire trouvé.

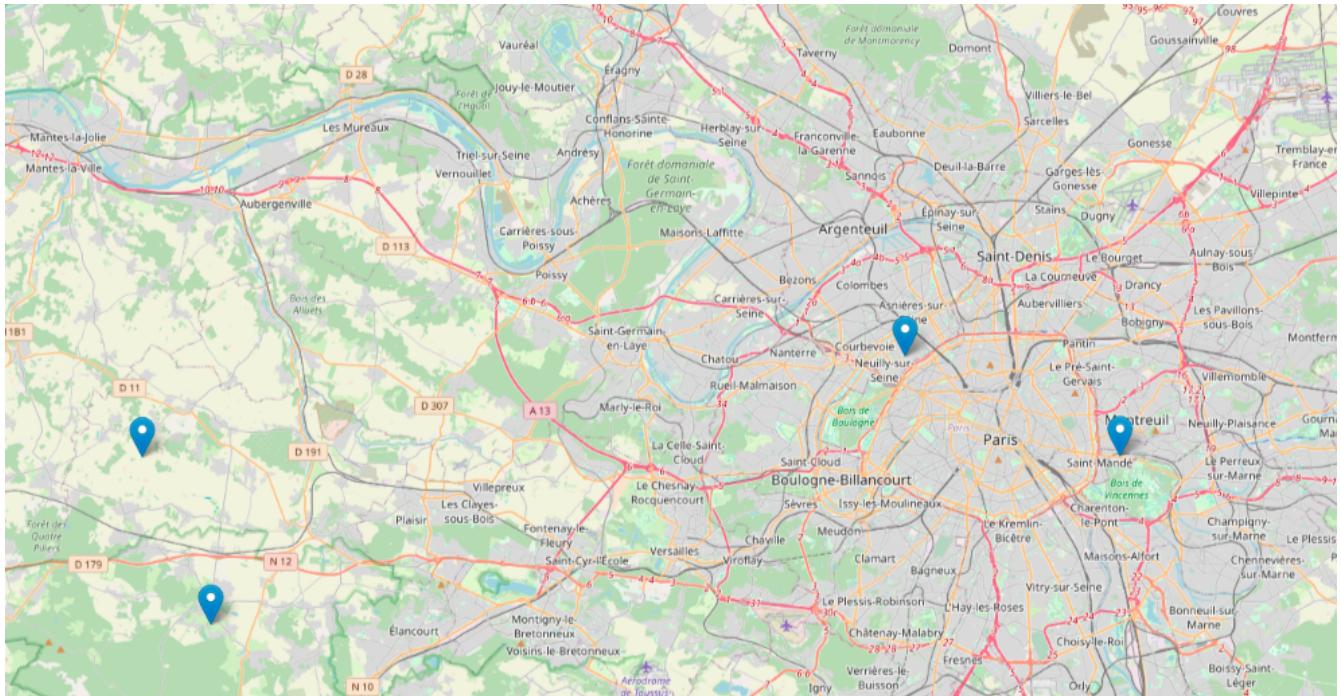


## Recherche par ville "Geneve":



- Rechercher un Jardin pour se Balader :

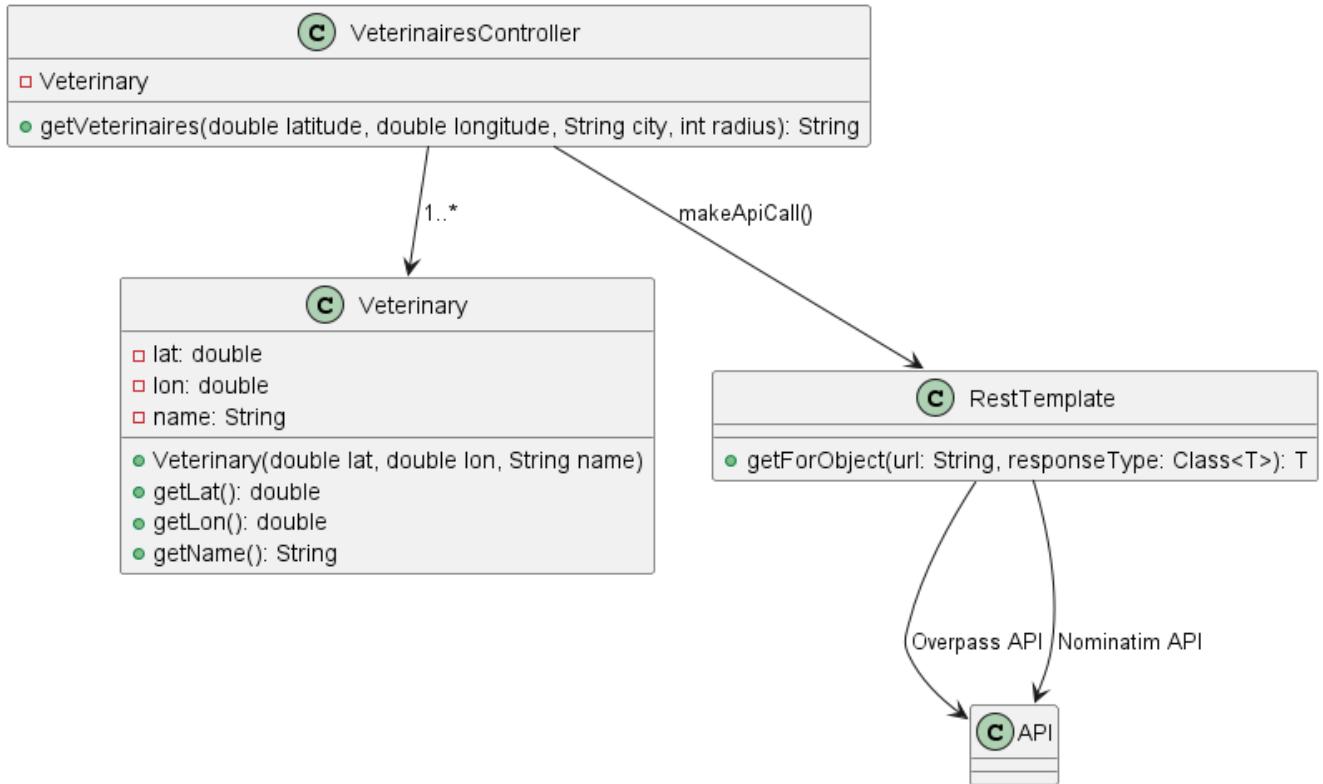
Pour la recherche de jardins, les résultats correspondants aux nœuds "leisure=garden" sont affichés sur la carte.



## Explication de l'API Overpass et Nominatim

L'API Overpass est un service d'interrogation et d'analyse de données OpenStreetMap. Elle permet de récupérer des données géographiques en utilisant un langage de requête spécifique. Dans le contexte de cette application, elle est utilisée pour obtenir des informations sur les vétérinaires et les jardins à proximité en fonction de la position de l'utilisateur. Les requêtes sont construites dynamiquement pour cibler les catégories spécifiques (amenity=veterinary, leisure=garden) et les résultats sont intégrés à la carte de l'application.

## Diagramme de class



==API Contrôleur des Vétérinaires

Ce contrôleur utilise l'API Overpass pour récupérer la liste des vétérinaires dans une ville donnée.

# Paramètres de Requête

Ce contrôleur accepte les paramètres de requête suivants :

- **city** (String): Nom de la ville.
- **latitude** (double): Latitude du point central.
- **longitude** (double): Longitude du point central.
- **radius** (double, facultatif, valeur par défaut: 3000): Rayon de recherche en mètres.

Exemple de requête:

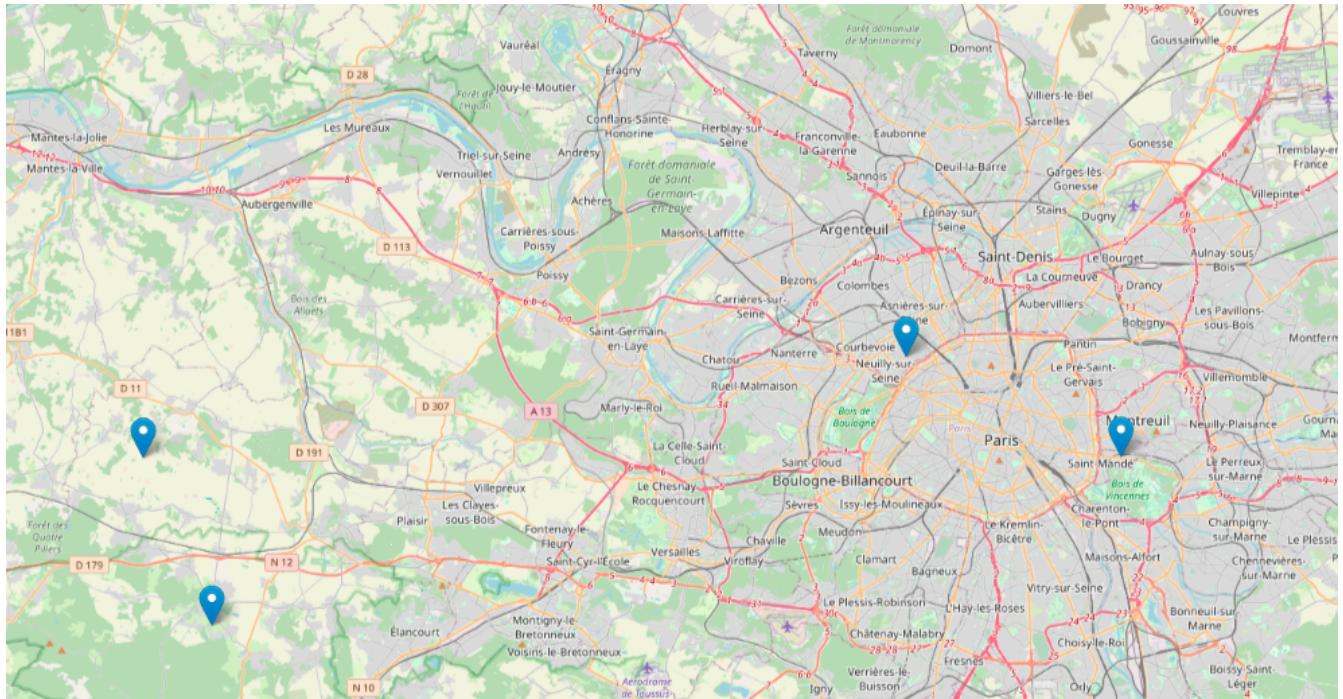
[Exemple]

```
/api/veterinaires?city=Paris&latitude=48.8566&longitude=2.3522&radius=5000
```

# Exemple de Réponse

La réponse du contrôleur est une liste d'objets **Veterinary** avec les coordonnées géographiques et le nom du vétérinaire.

Exemple de réponse:



## Explication de l'API Overpass et Nominatim

L'API Overpass est un service d'interrogation et d'analyse de données OpenStreetMap. Elle permet de récupérer des données géographiques en utilisant un langage de requête spécifique. Dans le contexte de cette application, elle est utilisée pour obtenir des informations sur les vétérinaires et les jardins à proximité en fonction de la position de l'utilisateur. Les requêtes sont construites dynamiquement pour cibler les catégories spécifiques (`amenity=veterinary`, `leisure=garden`) et les résultats sont intégrés à la carte de l'application.

# Fonctionnalité Interaction avec les Avis (Petite)

## Maquette du front

Les utilisateurs ont la possibilité d'interagir avec les avis en les consultant, en les likant, ou en les dislikant. Cette fonctionnalité permet d'exprimer des réactions vis-à-vis des conseils postés par d'autres utilisateurs.

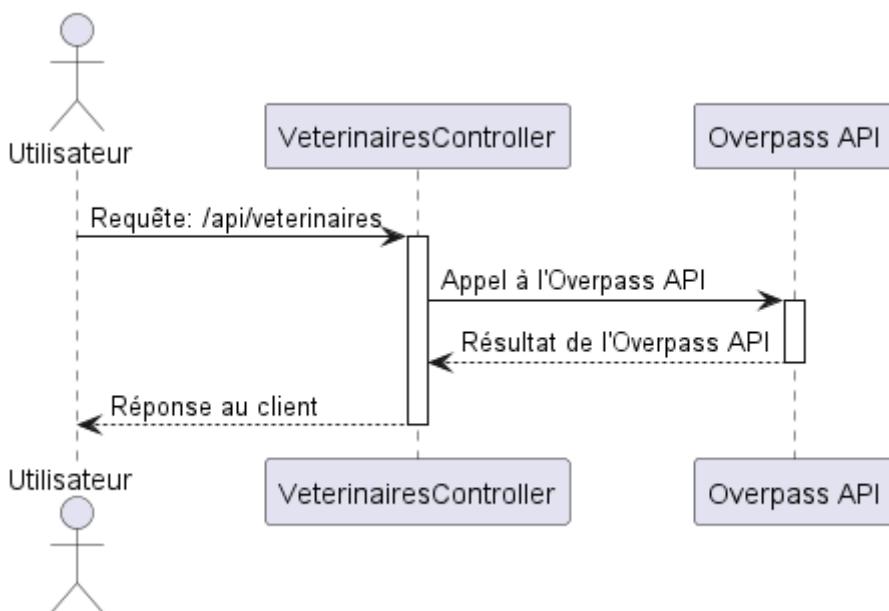
## Maquette du front

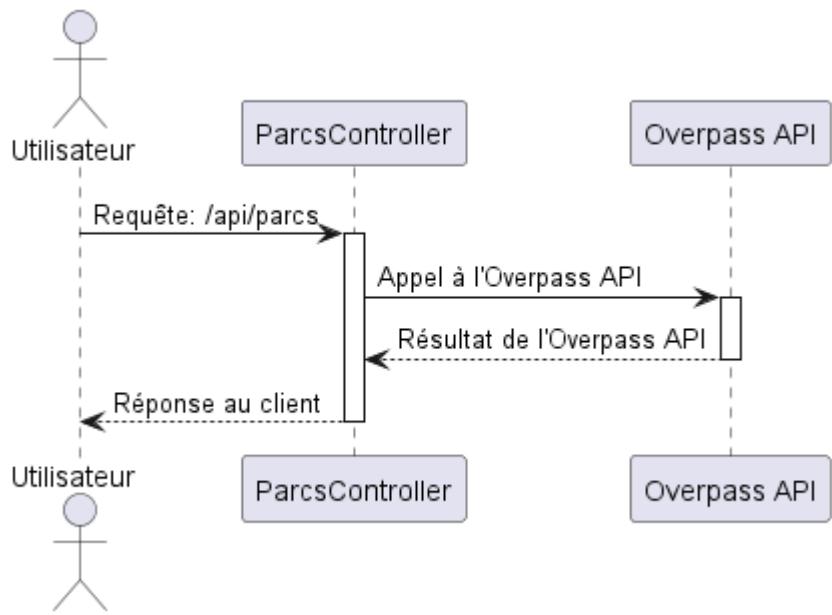
[advice] | *images\_planification\_food/7.png*

### Consulter les Avis

```
[  
 {  
   "lat": 45.8534,  
   "lon": 3.3488,  
   "name": "Vétérinaire A"  
 },  
 {  
   "lat": 48.8591,  
   "lon": 2.3637,  
   "name": "Vétérinaire B"  
 },  
 // ...  
 ]
```

## Diagramme de séquence





# Fonctionnalité de Planification des Heures de Repas et Envoi de Mails et confirmation du mail

## Planification des Heures de Repas et Envoi de Mails

Cette fonctionnalité permet à un utilisateur de programmer les heures auxquelles il souhaite nourrir son animal

### Maquette du front

Etape 1: Rentrer une heure puis cliquer sur le boutton "Creer une alerte" Puis l'alerte sera ajouter au tableau

Aussi on peut modifier les alertes créer en cliquant sur "Edith" , et supprimer ses alerte en "cliquant" sur Delete

La maquette du front de l'application LoveMyPet est divisée en deux sections principales.

**Section de gauche (Barre latérale):**

- Logo LOVEmypet avec un dessin d'un chien.
- Menu vertical :
  - Mon compte
  - Mes candidatures
  - Separateur
  - Mes animaux
  - Adoption
  - Inscrire un animal
  - Separateur
  - Se Déconnecter
  - Separateur
  - Donner des objets
  - Trouver des objets
  - Mes objets à donné

**Section de droite (Tableau des alertes):**

- Titre : Mes alertes**
- Message : Vous allez recevoir des Mails ! 📧**
- Bouton : CRÉER UNE ALERTE ⚡**
- Tableau de gestion des alertes :**

Feeding Time	Actions	
⌚ 06:30	SUPPRIMER	MODIFIER
⌚ 17:30	SUPPRIMER	MODIFIER

## Envoi de Mails de Rappel

Ensuite l'utilisateur n'a plus rien à faire , il recevra un email pour chaque heure rentrer l'informant de l'heure de repas prévue pour son animal.

Bonjour sadikou,

C'est l'heure de nourrir Lola

Cliquez sur le bouton ci-dessous pour confirmer que vous avez nourri votre animal :

[Confirmer](#)

N'oubliez pas de donner à votre animal son repas quotidien.

Merci de prendre soin de votre animal!

Cordialement

LoveMyPet

savagedeveloppement@gmail.com  
À moi

mer. 20 déc. 2023 11:38

Répondre Transférer

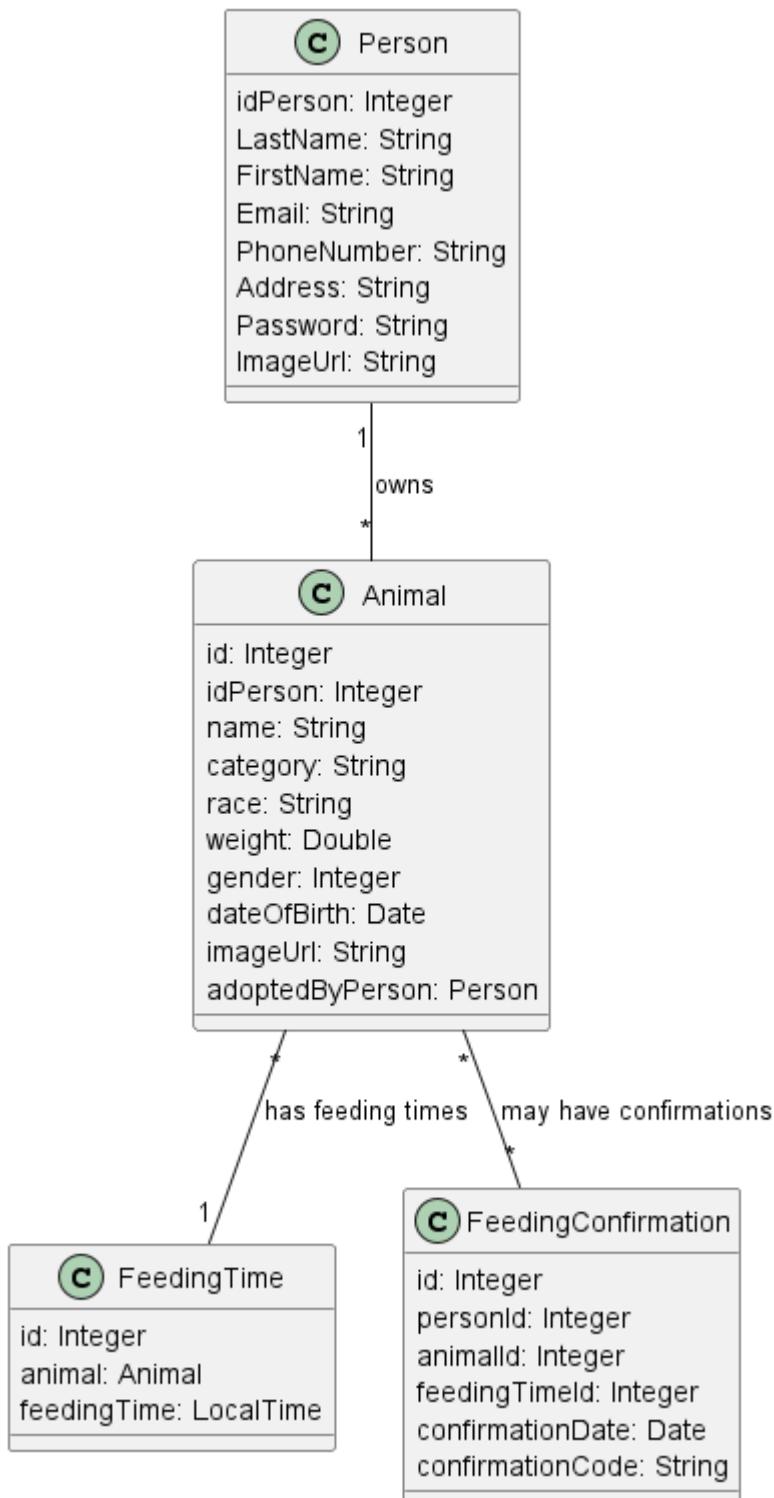
## Confirmation de l'E-mail

Pour confirmer la réception de l'e-mail et l'alimentation de l'animal, l'utilisateur peut cliquer sur le bouton de confirmation présent dans l'e-mail. Lorsqu'il le fait, un message de réussite est affiché, et une nouvelle ligne est insérée dans la base de données pour enregistrer la confirmation.

Le bouton de confirmation dans l'e-mail doit rediriger l'utilisateur vers une page ou un endpoint dédié, par exemple :

```
<a href="/feeding-confirmation/confirm?personId=1&animalId=1&feedingTimeId=39&confirmationCode=ad97faf5">Confirmer</a>
```

## Diagramme de classes global (partie métier)



## API de la fonctionnalité

### 1) Ajout d'un Horaire d'Alimentation

Endpoint : `POST /api/feeding-times/add`

Voici le body de la requête :

```
{  
  "animal": {  
    "id": 1  
  },  
  "feedingTime": "20:45"  
}
```

et le output :

Message : Horaires d'alimentation ajouté avec succès.

## 2) Liste des Horaires d'Alimentation pour un Animal

Endpoint : `GET /api/feeding-times/{idAnimal}`

et le output :

```
["time/1", "time/2", "time/3"]
```

## 3) Détail d'un Horaire d'Alimentation

Endpoint : `GET /api/feeding-times/time/{id}`

et le output :

```
{  
  "id": 1,  
  "animal": {  
    "id": 1,  
    "idPerson": 1,  
    "name": "xgecfie",  
    "category": "cat",  
    "race": "dde",  
    "weight": 20.0,  
    "gender": 1,  
    "dateOfBirth": "2024-01-25",  
    "imageUrl": "selenium_true_vrai.jpg",  
    "adoptedByPerson": {  
      "idPerson": 1,  
      "password": "f",  
      "address": null,  
      "email": "f@gmail.com",  
      "imageUrl": null,  
      "lastName": "ichola",  
      "firstName": "sadikou",  
      "phoneNumber": null  
    }  
  },  
}
```

```
"feedingTime": "15:42"  
}
```

## 4) Suppression d'un Horaire d'Alimentation

Endpoint : `DELETE /api/feeding-times/delete/{id}`

Et output :

L'heure a été supprimée

## 4) Mise à Jour d'un Horaire d'Alimentation

Endpoint : `PUT /api/feeding-times/update/{id}`

```
{  
  "animal": {  
    "id": 1  
  },  
  "feedingTime": "15:45"  
}
```

Output : Mise à jour réussie.

## 5) Récupération des Horaire d'Alimentation Actuels pour les E-mails

Endpoint : `GET /api/feeding-times/email-animal-current-feeding-times`

```
[  
  [  
    "f@gmail.com",  
    "Fanuel",  
    1,  
    1,  
    "loulou",  
    "image_animal.jpg",  
    4  
  ]  
]
```

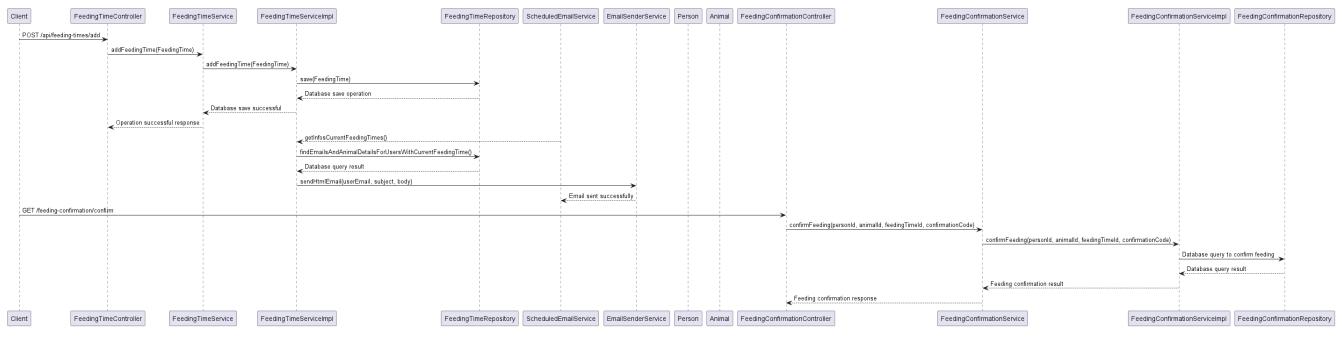
## 6) Confirmer l'email

Endpoint : `GET /feeding-confirmation/confirm?personId=1&animalId=2&feedingTimeId=3&confirmationCode=ABC123`

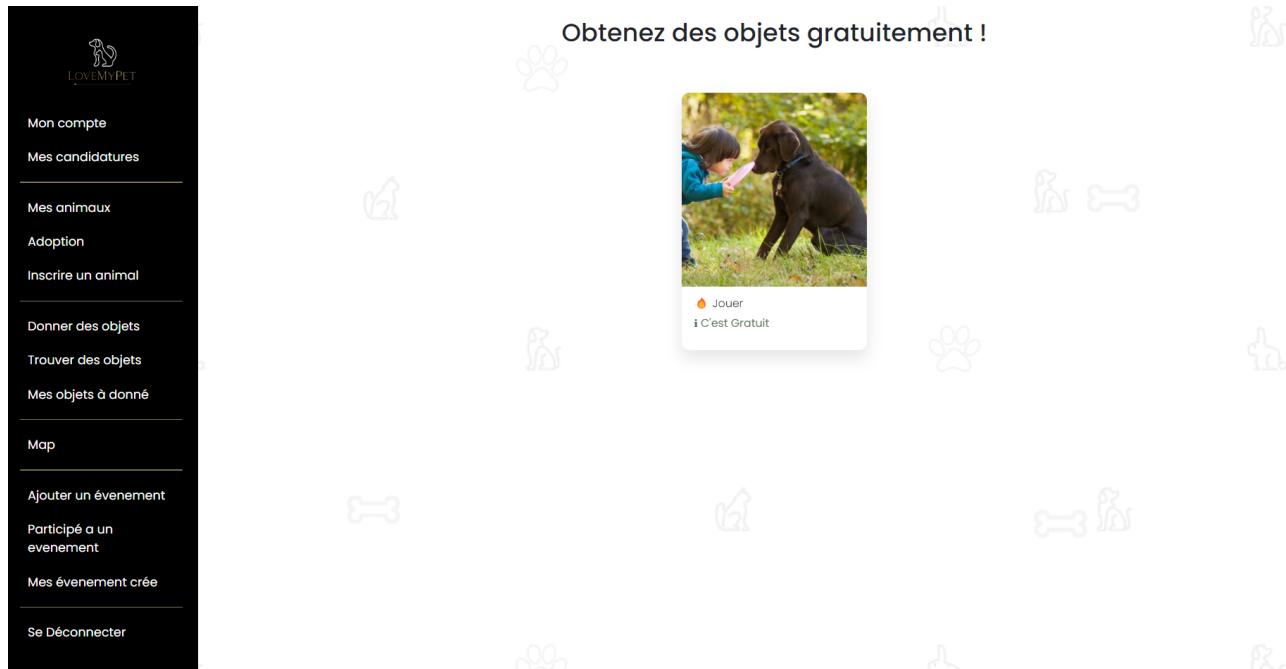
Output :

Confirmation réussie Failed to confirm feeding.

# Diagramme de séquence



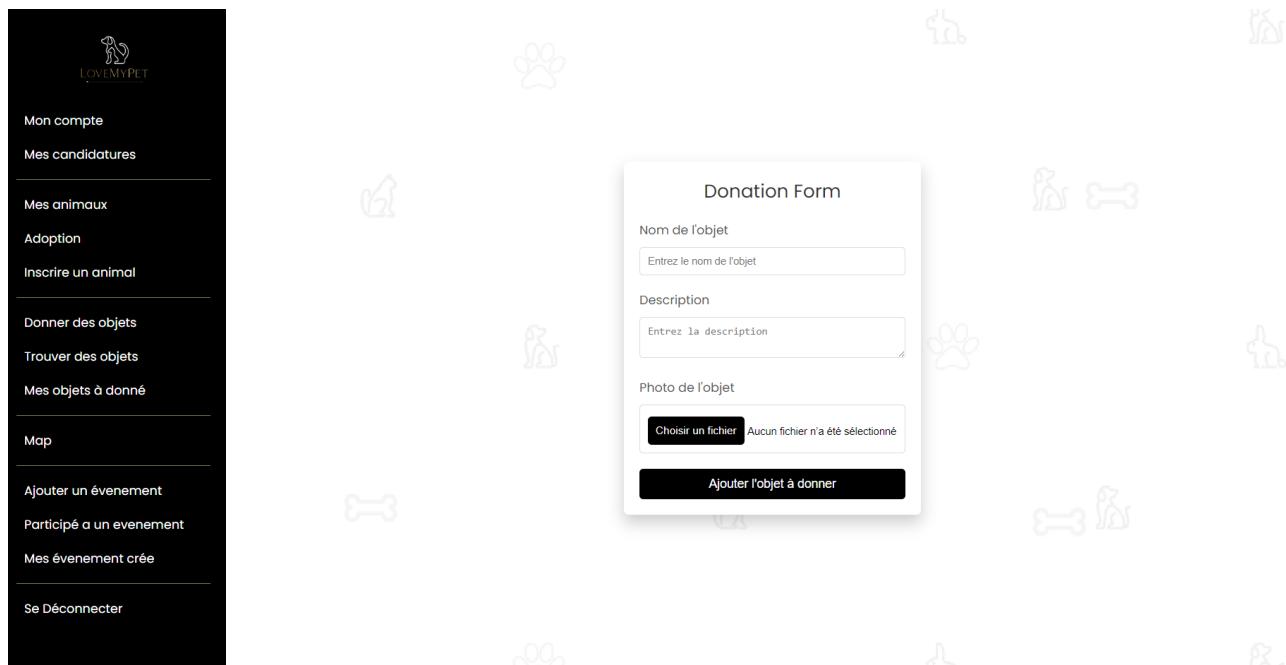
# Fonctionnalité de donnation d'objets



## Description de la fonctionnalité

La fonctionnalité de donation d'objets permet aux utilisateurs de proposer des objets qu'ils souhaitent donner à d'autres personnes. Voici comment fonctionne cette fonctionnalité :

### Ajout d'objets à donner



Les utilisateurs peuvent ajouter des informations sur les objets qu'ils souhaitent donner, telles que le nom de l'objet, une description et éventuellement une photo. En utilisant une interface

conviviale, ils remplissent un formulaire avec les détails de l'objet, y compris le nom, la description et la photo facultative. Une fois le formulaire soumis, les informations sur l'objet sont enregistrées dans la base de données.

## **Consultation des objets disponibles**

Les autres utilisateurs peuvent consulter la liste des objets disponibles à donner. Ils peuvent parcourir les objets ajoutés par d'autres utilisateurs et voir leurs détails, tels que le nom, la description et la photo. Cette fonctionnalité leur permet de trouver des objets qui pourraient les intéresser.

## **Contacter le posteur pour récupérer l'objet**

Si un utilisateur est intéressé par un objet à donner, il peut contacter le posteur de l'objet pour organiser la récupération. Cela peut se faire par le biais de coordonnées fournies par l'utilisateur qui donne l'objet, telles qu'une adresse e-mail ou un numéro de téléphone. Les deux parties peuvent ensuite convenir d'un moment et d'un lieu pour que l'utilisateur récupère l'objet donné.

## **Objectifs de la fonctionnalité**

- Faciliter le processus de donation d'objets en permettant aux utilisateurs de proposer des objets à donner.
- Fournir aux utilisateurs une plateforme où ils peuvent trouver des objets disponibles à donner qui correspondent à leurs besoins.
- Encourager le partage et la réutilisation des objets pour réduire le gaspillage et favoriser le développement durable.
- Créer une communauté où les utilisateurs peuvent se soutenir mutuellement en donnant et en recevant des objets de manière désintéressée

# API de la fonctionnalité de donation d'objets

L'API de la fonctionnalité de donation d'objets permet aux utilisateurs d'effectuer différentes opérations liées à la gestion des objets à donner.

## Ajout d'objets à donner

Endpoint : **POST /api/items-to-donate/add**

Ce point de terminaison permet aux utilisateurs d'ajouter des informations sur les objets qu'ils souhaitent donner. Les informations nécessaires comprennent le nom de l'objet, une description et éventuellement une photo. Les paramètres de la requête sont les suivants :

- **itemName** : Le nom de l'objet à donner.
- **description** : La description de l'objet à donner.
- **photo** : La photo de l'objet à donner (facultatif).
- **idPerson** : L'identifiant de la personne qui donne l'objet.

Exemple de corps de requête JSON :

```
{  
  "itemName": "Chaise",  
  "description": "Chaise en bois",  
  "photo": "photo_chaise.jpg",  
  "idPerson": 123  
}
```

Réponse : En cas de succès, une réponse avec le message "Objet à donner ajouté avec succès" est renvoyée avec le code d'état HTTP 200 (OK). En cas d'erreur, une réponse avec le message "Erreur lors de l'ajout de l'objet à donner" est renvoyée avec le code d'état HTTP 500 (Internal Server Error).

## Consultation des objets disponibles

Endpoint : **GET /api/items-to-donate/**

Ce point de terminaison permet aux utilisateurs de consulter la liste des objets disponibles à donner. Il renvoie une liste d'URLs vers les détails de chaque objet disponible.

Exemple de réponse JSON :

```
[  
  "/api/items-to-donate/item/1",  
  ...]
```

```
    "/api/items-to-donate/item/2",
    "/api/items-to-donate/item/3"
]
```

## Détails d'un objet à donner

Endpoint : `GET /api/items-to-donate/item/{id}`

Ce point de terminaison permet aux utilisateurs de récupérer les détails d'un objet à donner spécifique en fournissant son identifiant (`id`).

Exemple de réponse JSON pour un objet avec l'identifiant 1 :

```
{
  "id": 1,
  "itemName": "Chaise",
  "description": "Chaise en bois",
  "imageUrl": "photo_chaise.jpg",
  "donatingPerson": {
    "idPerson": 123,
    "firstName": "John",
    "lastName": "Doe"
  }
}
```

## Récupération des objets d'une personne

Endpoint : `GET /api/items-to-donate/person/{personId}`

Ce point de terminaison permet aux utilisateurs de récupérer les objets qu'une personne spécifique a l'intention de donner. Il prend en paramètre l'identifiant de la personne (`personId`) et renvoie une liste d'URLs vers les détails de chaque objet.

Exemple de réponse JSON pour les objets d'une personne avec l'identifiant 123 :

```
[
  "/api/items-to-donate/item/1",
  "/api/items-to-donate/item/2"
]
```

## Suppression d'un objet à donner

Endpoint : `DELETE /api/items-to-donate/item/{id}`

Ce point de terminaison permet aux utilisateurs de supprimer un objet à donner spécifique en fournissant son identifiant (`id`). En cas de succès, une réponse avec le message "Objet à donner supprimé avec succès" est renvoyée avec le code d'état HTTP 200 (OK).

Exemple de réponse JSON pour la suppression d'un objet avec l'identifiant 1 :

```
{  
    "message": "Objet à donner supprimé avec succès"  
}
```

# Diagramme de séquence



# **Fonctionnalité de Crédit et Inscription aux Événements sur LoveMyPet**

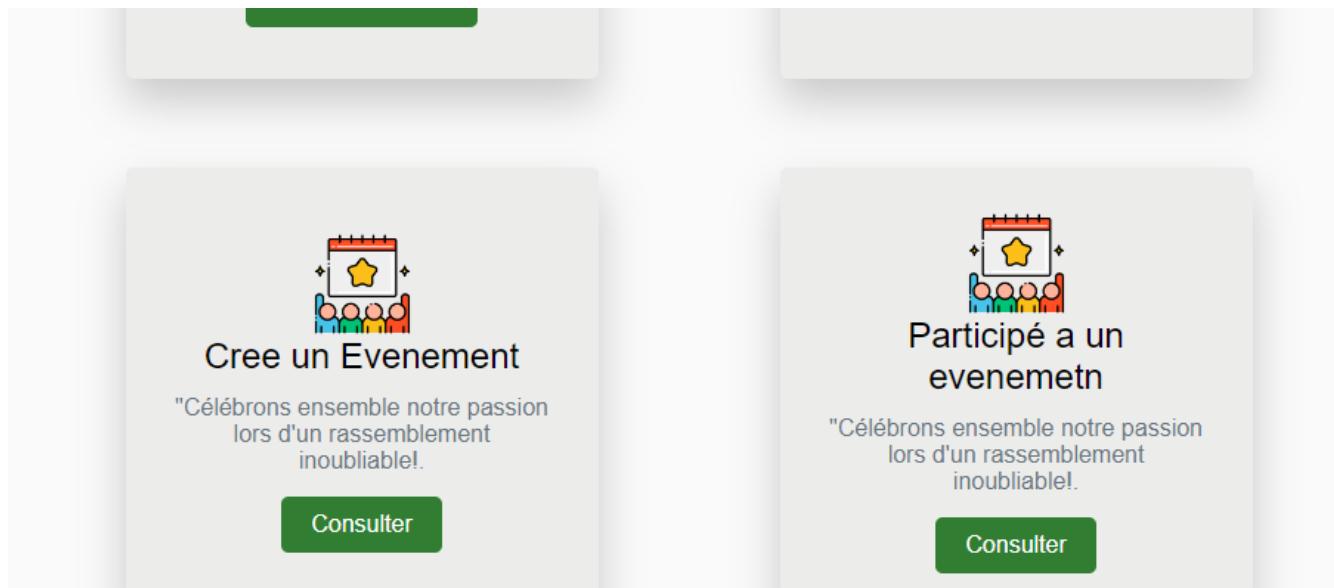
# Présentation

Cette fonctionnalité sur l'application LoveMyPet permet aux utilisateurs de créer et partager des événements, tels que des sorties en groupe, et de s'inscrire à des événements créés par d'autres utilisateurs.

# Utilisation depuis la Page "Mes Services"

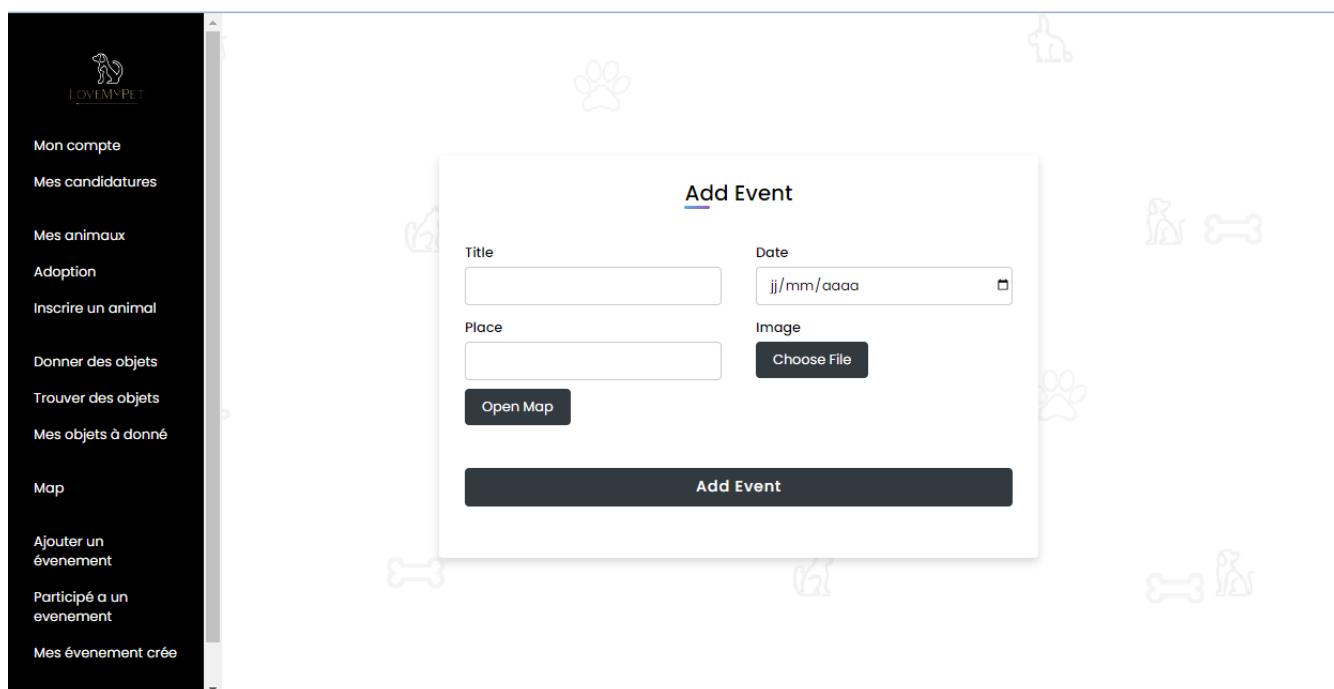
Sur la page "Mes Services", deux boutons sont disponibles :

- Un bouton permettant à l'utilisateur de visualiser tous les événements créés par d'autres utilisateurs qui ne sont pas encore expirés.
- Un bouton permettant à l'utilisateur de créer un nouvel événement et de le partager avec la communauté. De plus, l'utilisateur peut visualiser tous les événements qu'il a créés.



## Ajout d'un Événement

Lorsque l'utilisateur clique sur le bouton "Ajouter un Événement", il est redirigé vers une page où il peut remplir un formulaire avec des informations telles que le titre, la date, l'adresse de l'événement, et même ajouter une image pour l'événement.



# Affichage des Événements Non Expirés

En cliquant sur le bouton "Afficher les Événements Non Expirés", l'utilisateur est dirigé vers une page présentant tous les événements non expirés. Il peut facilement s'inscrire à un événement en cliquant sur le bouton d'inscription associé à l'événement choisi.

The screenshot shows a sidebar on the left with a dark background and white text, listing navigation options: Mon compte, Mes candidatures, Mes animaux, Adoption, Inscrire un animal, Donner des objets, Trouver des objets, Mes objets à donné, Map, Ajouter un événement, Participé a un evenement, Mes événement crée, and Se Déconnecter. The main content area has a light gray background with a title 'LISTE DES ÉVÉNEMENTS À VENIR' at the top. It displays three event cards with mountain landscapes. The first card is for a 'Sortie' on 2024-12-06 in Paris, with a 'Voir sur Google Maps' link and an 'Inscription' button. The second card is for a 'Rencontre' on 2024-03-25 in Paris, with a 'Voir sur Google Maps' link and an 'Inscription' button. The third card is for a 'Rencontre' on 2024-03-25 in Geneva, with a 'Voir sur Google Maps' link and an 'Inscription' button. There are also two smaller event cards below the main ones.

## Inscription à un Événement

L'inscription à un événement est simple. L'utilisateur clique sur le bouton "Inscription" associé à l'événement de son choix. Une fois inscrit, une entrée est ajoutée à la table des inscriptions avec l'ID de l'événement et l'ID de l'utilisateur.

## Affichage des Événements Crées par un Utilisateur

Si un utilisateur souhaite voir les événements qu'il a créés, il lui suffit de se rendre dans le menu dédié. Il sera alors redirigé vers une page listant tous les événements qu'il a créés.

The screenshot shows the same sidebar as the previous page. The main content area has a light gray background with a title 'LES ÉVÉNEMENTS QUE VOUS AVEZ CRÉÉS' at the top. It displays three event cards with mountain landscapes. The first card is for a 'Sortie' on 2024-12-06 in Paris, with a 'Voir sur Google Maps' link and a 'Supprimer' button. The second card is for a 'Rencontre' on 2024-03-25 in Geneva, with a 'Voir sur Google Maps' link and a 'Supprimer' button. The third card is for a 'Rencontre' on 2024-03-25 in Paris, with a 'Voir sur Google Maps' link and a 'Supprimer' button. There are also two smaller event cards below the main ones.

# Suppression d'un Événement Crée

L'utilisateur peut supprimer un événement qu'il a créé en accédant à la page "Mes Événements". Sur cette page, il voit la liste de tous les événements qu'il a créés et peut sélectionner l'événement à supprimer en cliquant sur le bouton "Supprimer".

The screenshot shows a user interface for managing events. On the left is a sidebar with a logo and various menu items. The main area displays three event cards. A modal dialog box is overlaid on the screen, indicating a successful operation.

**Mon compte**  
**Mes candidatures**  
**Mes animaux**  
**Adoption**  
**Inscrire un animal**  
**Donner des objets**  
**Trouver des objets**  
**Mes objets à donné**  
**Map**  
**Ajouter un événement**  
**Participé à un evenement**  
**Mes événement crée**  
**Se Déconnecter**

**localhost:8086 indique**  
Événement supprimé avec succès!

**Sortie**  
Date: 2024-12-06  
Adresse: Paris  
[Voir sur Google Maps](#)

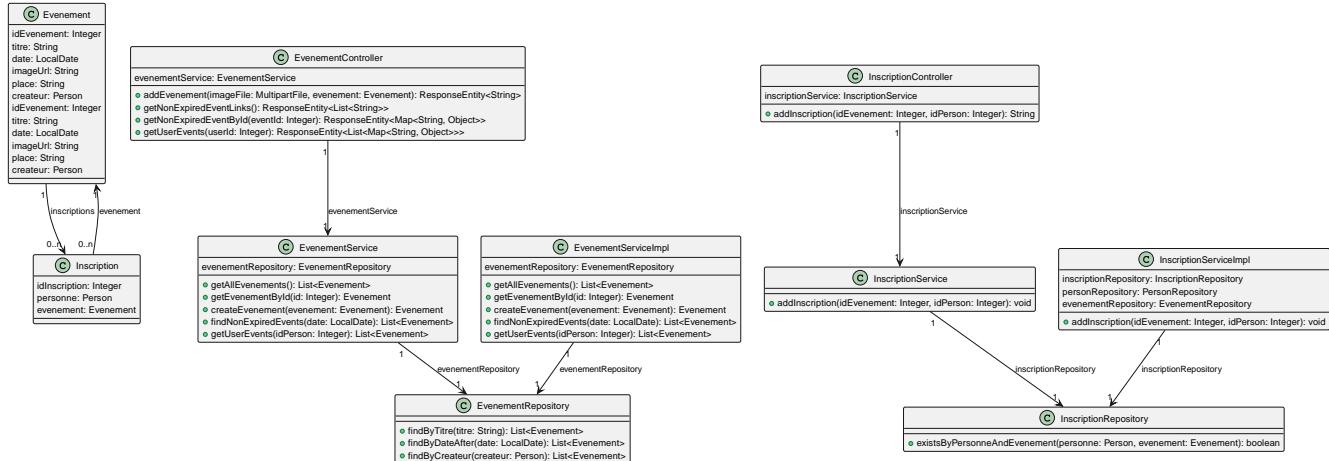
**Rencontre**  
Date: 2024-03-25  
Adresse: Genève  
[Voir sur Google Maps](#)

**Rencontre**  
Date: 2024-03-25  
Adresse: Paris  
[Voir sur Google Maps](#)

**Supprimer**

**OK**

# Diagramme de Classe



# API

## Ajout d'Événement

*Inscription d'un nouvel événement dans l'application.*

- Type: **POST**
- Endpoint: [/api/evenements/add](#)

```
{  
    "imageFile": "Contenu de l'image en format MultipartFile",  
    "evenement": {  
        "titre": "Sortie au Parc",  
        "date": "2024-02-01",  
        "place": "Parc XYZ",  
        "createur": {  
            "idPerson": 123  
        }  
    }  
}
```

- Succès (Status Code 200 OK): "Nouvel événement ajouté"

## Affichage des Événements Non Expirés

*Obtention de la liste des liens vers les événements non expirés.*

- Type: **GET**
- Endpoint: [/api/evenements/non-expired](#)
- Succès (Status Code 200 OK): Liste des liens vers les événements non expirés

## Détails d'un Événement Non Expiré

*Obtention des détails d'un événement non expiré.*

- Type: **GET**
- Endpoint: [/api/evenements/non-expired/1](#)
- Succès (Status Code 200 OK):

```
[  
    "/api/evenements/non-expired/4",  
    "/api/evenements/non-expired/5",  
    "/api/evenements/non-expired/7",  
    "/api/evenements/non-expired/8",  
    "/api/evenements/non-expired/14"
```

- Échec (Status Code 404 Not Found): Événement non trouvé

## Liste des Événements Crées par un Utilisateur

*Obtention de la liste des événements créés par un utilisateur.*

- Type: **GET**
- Endpoint: **/api/evenements/user-events?userId=123**
- Succès (Status Code 200 OK):

```
[
  {
    "idEvenement": 1,
    "titre": "Sortie au Parc",
    "date": "2024-02-01",
    "place": "Parc XYZ",
    "imageName": "sortie_parc.jpg",
    "details": "/api/evenements/non-expired/1"
  },
  {
    "idEvenement": 2,
    "titre": "Promenade en Ville",
    "date": "2024-02-15",
    "place": "Centre-ville",
    "imageName": "promenade_ville.jpg",
    "details": "/api/evenements/non-expired/2"
  }
]
```

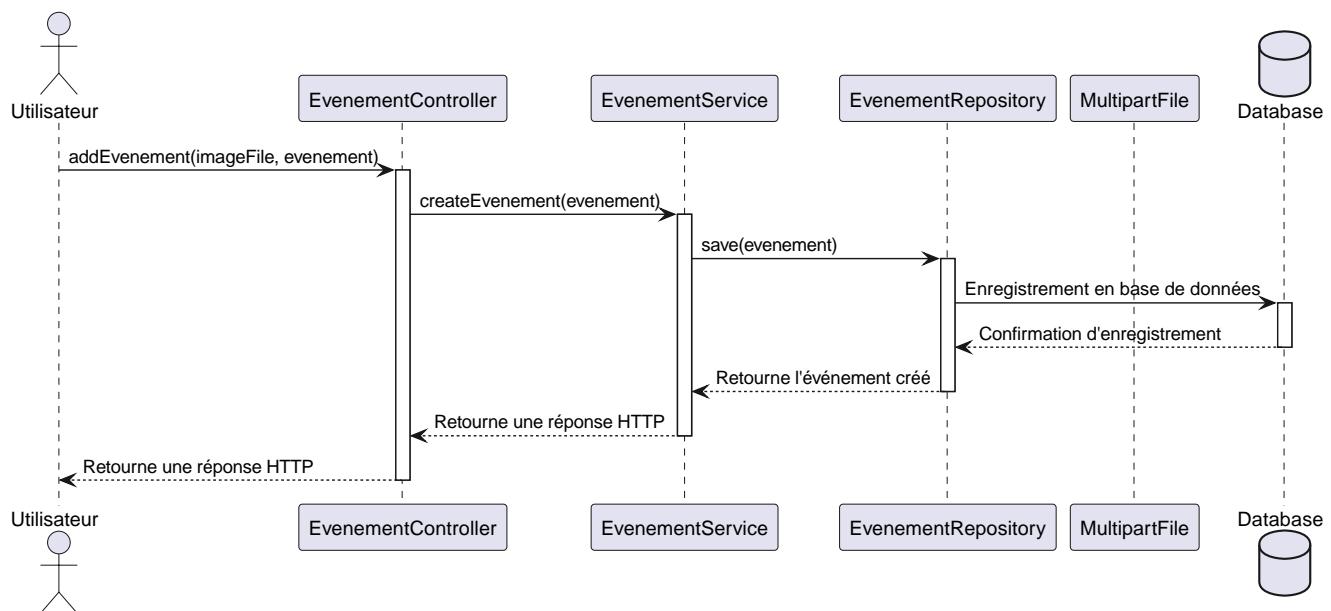
- Échec (Status Code 400 Bad Request): Paramètre manquant

## Inscription à un Événement

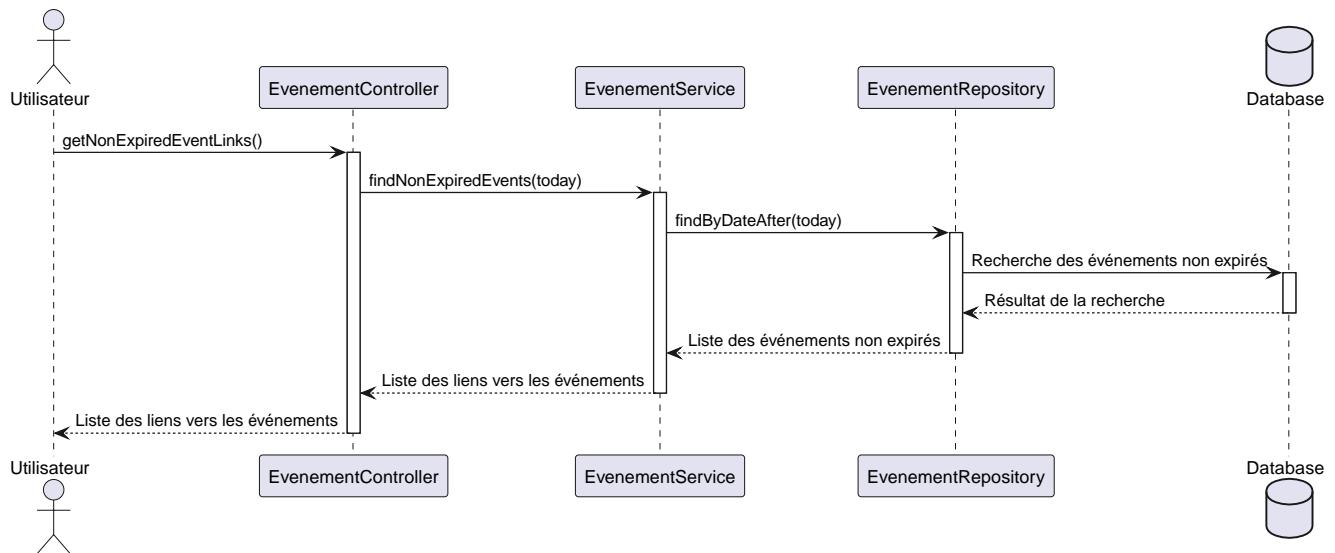
*Inscription à un événement spécifique.*

- Type: **POST**
- Endpoint: **/api/inscription/add?idEvenement=1&idPerson=456**
- Succès (Status Code 200 OK): "Inscription ajoutée avec succès!"
- Échec (Status Code 404 Not Found): "Personne ou événement non trouvé!"
- Échec (Status Code 400 Bad Request): "Personne est déjà inscrite à cet événement!"

# Diagramme de Séquence - Ajout d'Événement



# Diagramme de Séquence - Affichage des Événements Non Expirés



# Diagramme de Séquence - Inscription à un Événement

