

**INF3710 –Fichiers et Bases de données**

**Hiver 2019**

**TP No. 5**

**Groupe 2**

**1852132 – Kenny Lui**

**1893058 – Gabriel Houle-Violette**

**Soumis à : Julien Bergeron**

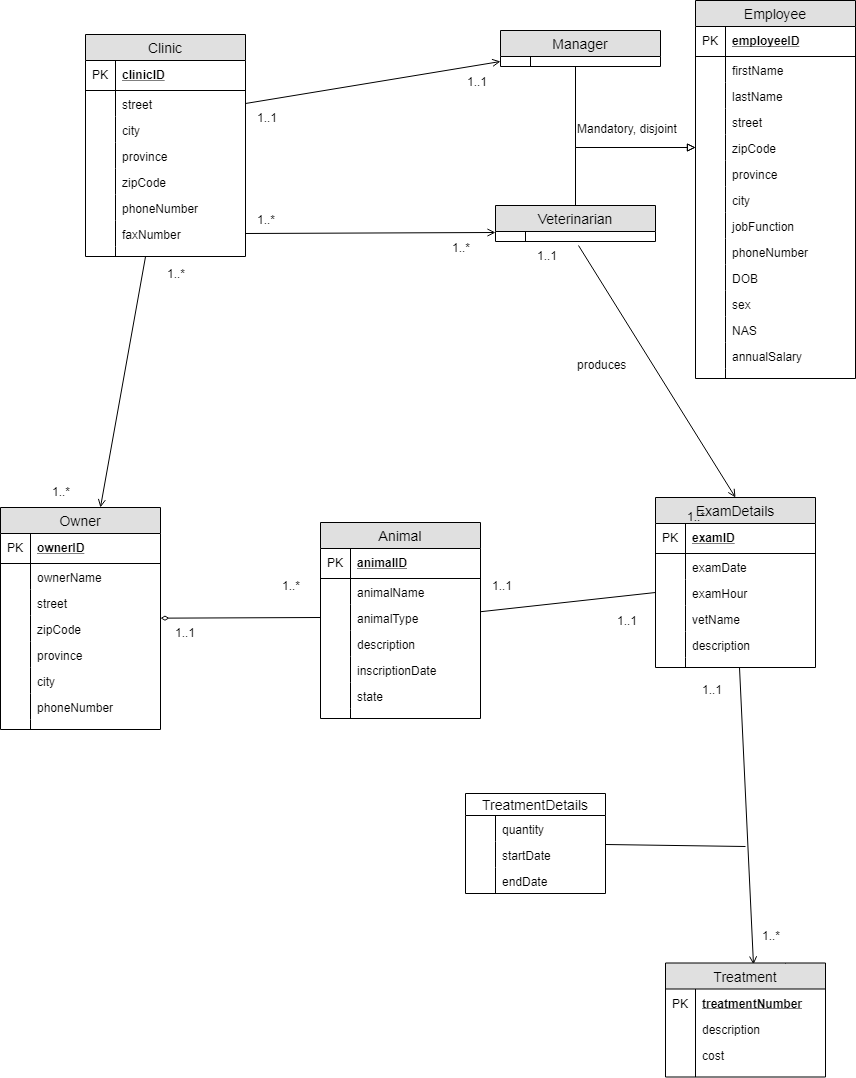
**14 avril 2019**

**Description du projet**

Dans le cadre de ce projet, il nous a été demandé de conceptualiser une base de donnée pour l’organisation VetoSansFrontieres afin qu’elle puisse contenir les diverses informations dans les cliniques bénéficiant de leurs services. Le manque de communication a représenter un problème dans la gestion des informations de l’organisation. Dans le cadre du projet, il nous a alors été nécessaire de produire un système de base de données centralisées afin d’améliorer la situation.

Les informations fournis pour la conception du système de base de données sont représentés par un texte présentant les requis de l’organisation. Il nous a alors été nécessaire de produire un modèle conceptuel répondant adéquatement aux requis. Ensuite, il nous a été nécessaire d’utiliser l’ensemble des connaissances en conception de base de données afin de produire une base de données à l’aide

**Modèle conceptuel**



**Modèle relationnel**

Clinic (clinicID, street, city, province, zipCode, phoneNumber, faxNumber)

Primary Key clinicID

Employee (employeeID, firstName, LastName, street, zipCode, city, province, jobFunction, phoneNumber, DOB, sex, NAS, annualSalary, clinicID)

Primary Key employeeID

Alternate Key NAS

Foreign Key clinicID references Clinic (clinicID)

Owner (ownerID, ownerName, street, zipCode, province, city, phoneNumber, clinicID)

Primary Key ownerID

Foreign Key clinicID references Clinic (clinicID)

Animal (animalID, animalName, animalType, description, inscriptionDate, animalState, ownerID)

Primary Key animalID

Foreign Key ownerID references Owner (ownerID)

ExamDetails (examID, examDate, examHour, description, animalID, vetID)

Primary Key examID

Foreign Key animalID references Animal (animalID)

Foreign Key vetID references Employee (employeeID)

Treatment (treatmentNumber, description, treatmentCost)

Primary Key treatmentNumber

TreatmentDetails (treatmentNumber, examID, quantity, startDate, endDate)

Primary Key treatmentNumber, examID

Foreign Key treatmentNumber references Treatment (treatmentNumber)

Foreign Key examID references ExamDetails (examID)

**Dépendances fonctionnels**

FD1:

clinicID --> street, city, province, zipCode, phoneNumber, faxNumber

FD2:

employeeID --> firstName, LastName, street, zipCode, city, province, jobFunction, phoneNumber, DOB, sex, NAS, annualSalary, clinicID

FD3:

NAS --> employeeID, firstName, LastName, street, zipCode, city, province, jobFunction, phoneNumber, DOB, sex, annualSalary, clinicID

FD4:

ownerID --> ownerName, street, zipCode, province, city, phoneNumber, clinicID

FD5:

animalID --> animalName, animalType, description, inscriptionDate, animalState, ownerID

FD6:

examID --> examDate, examHour, description, animalID, vetID

FD7:

treatmentNumber --> description, treatmentCost

FD8:

treatmentNumber, examID --> quantity, startDate, endDate

Explications

Notre base de données est de forme UNF puisque tous les éléments possèdent une clé

primaire unique (et composée dans le cas de FD8). Nous avons décidé de ne pas garder

le NAS comme clé primaire pour Employee (FD3).

Notre base de données est aussi en 1NF parce que tous les attributs sont atomiques.

C'est-à-dire que pour chaque attribut, une seule valeur est possible à la fois

par entité.

Notre BD est en 2NF, car chaque attribut qui n'est pas une clé primaire dépend

fonctionnellement complètement de la clé primaire.

À partir des clés primaires des relations 2NF, nous n'avons trouvé aucune dépendance

transitive. Par exemple, nous avions le nom du vétérinnaire dans FD6, mais nous

l'avons retiré puisqu'on avait déjà accès au vetID qui lui-même donne accès

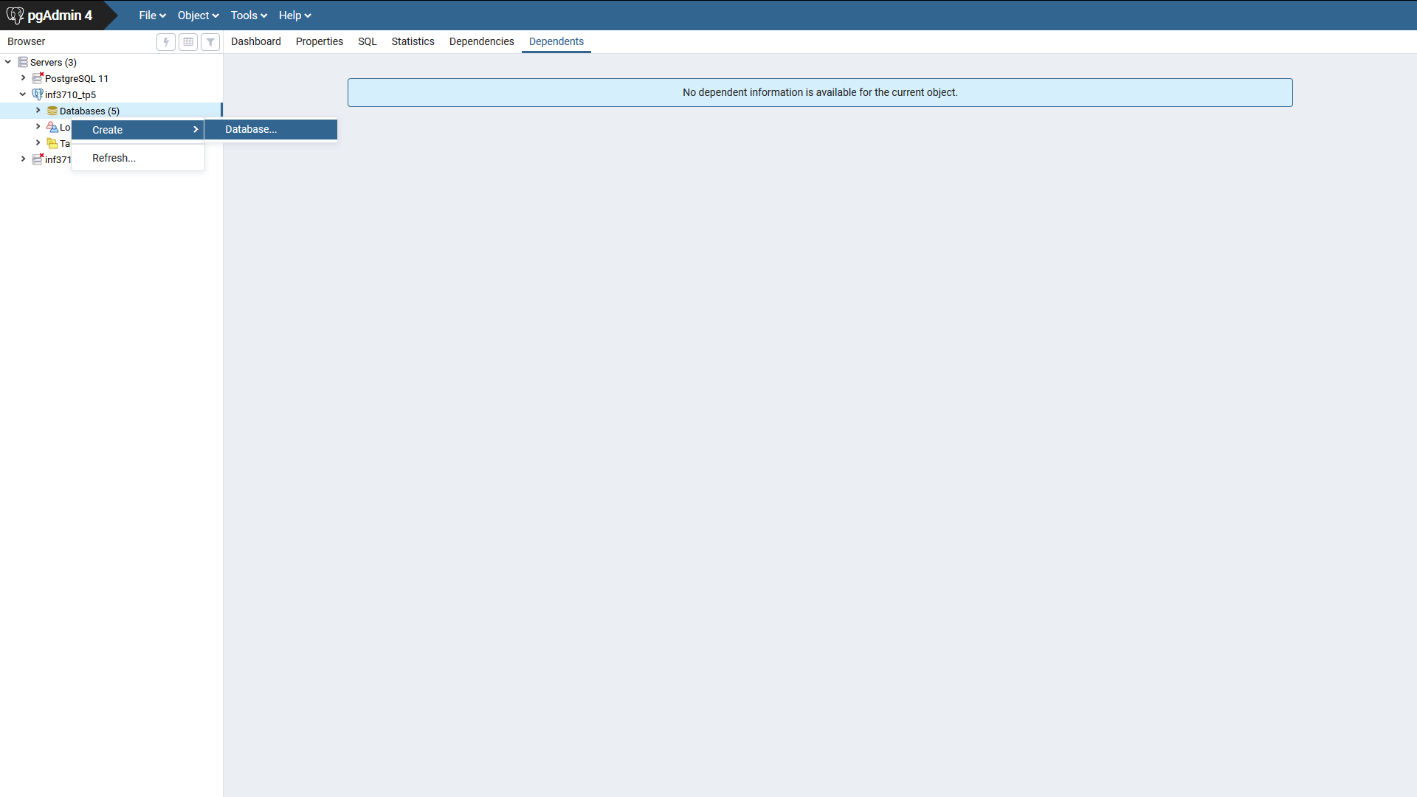
au nom. Nous avions donc déjà retiré la seule dépendance transitive avant de définir

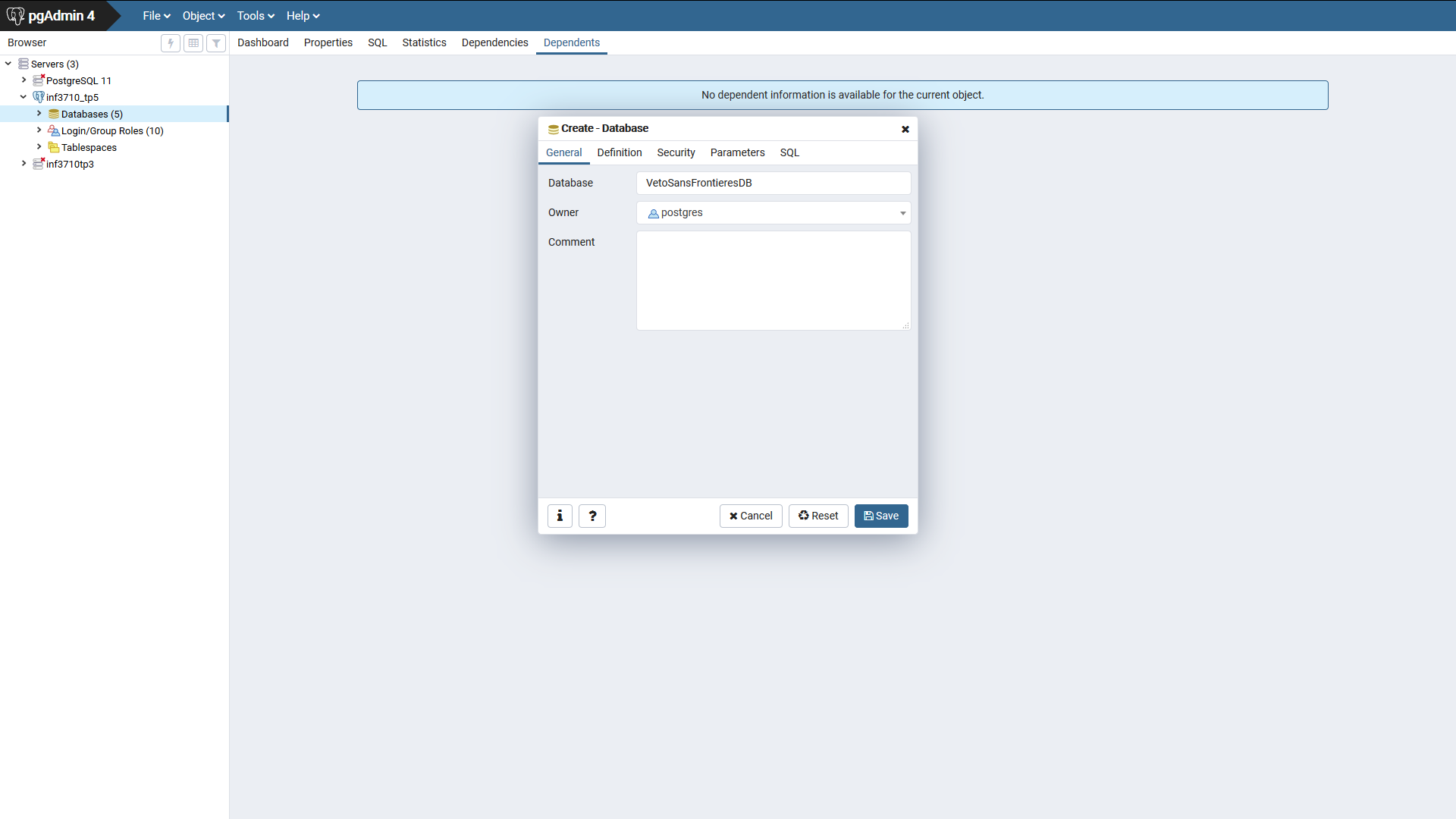
les dépendances fonctionnelles.

**Présentation de l’application Web**

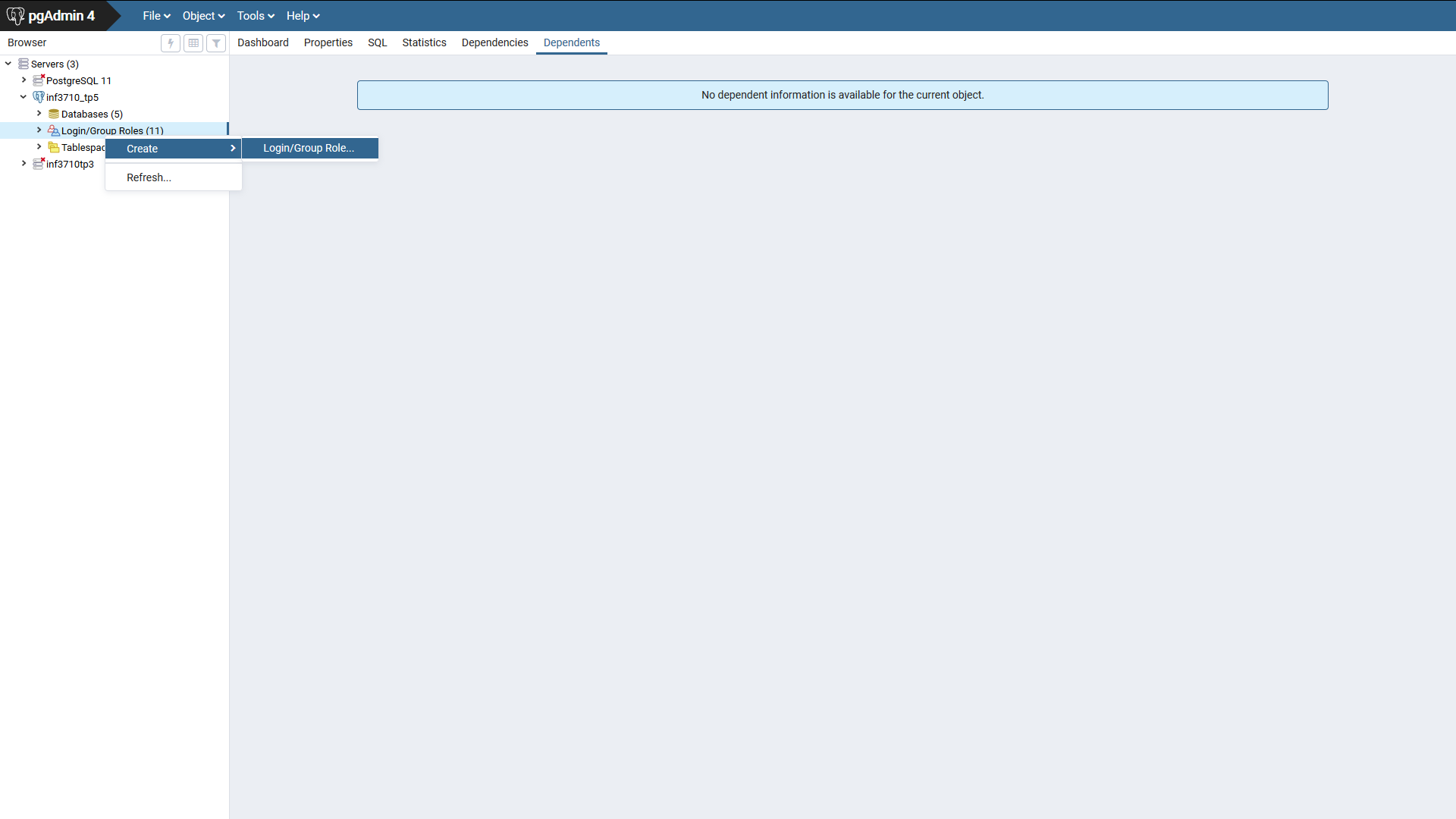
**Guide d’installation**

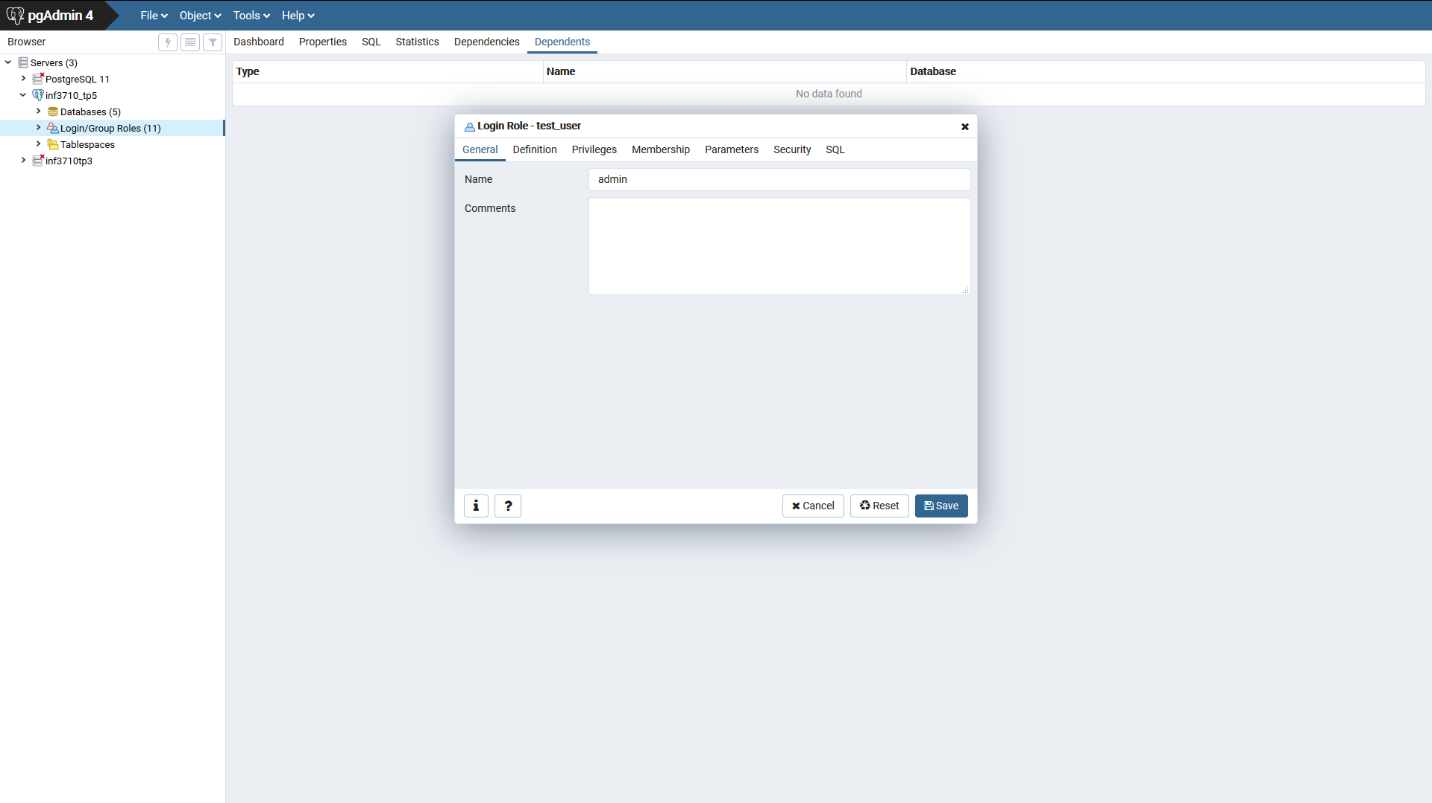
1. À l’aide de pgAdmin, veuillez créer un serveur nommé ‘’ VetoSansFrontieresDB’’.

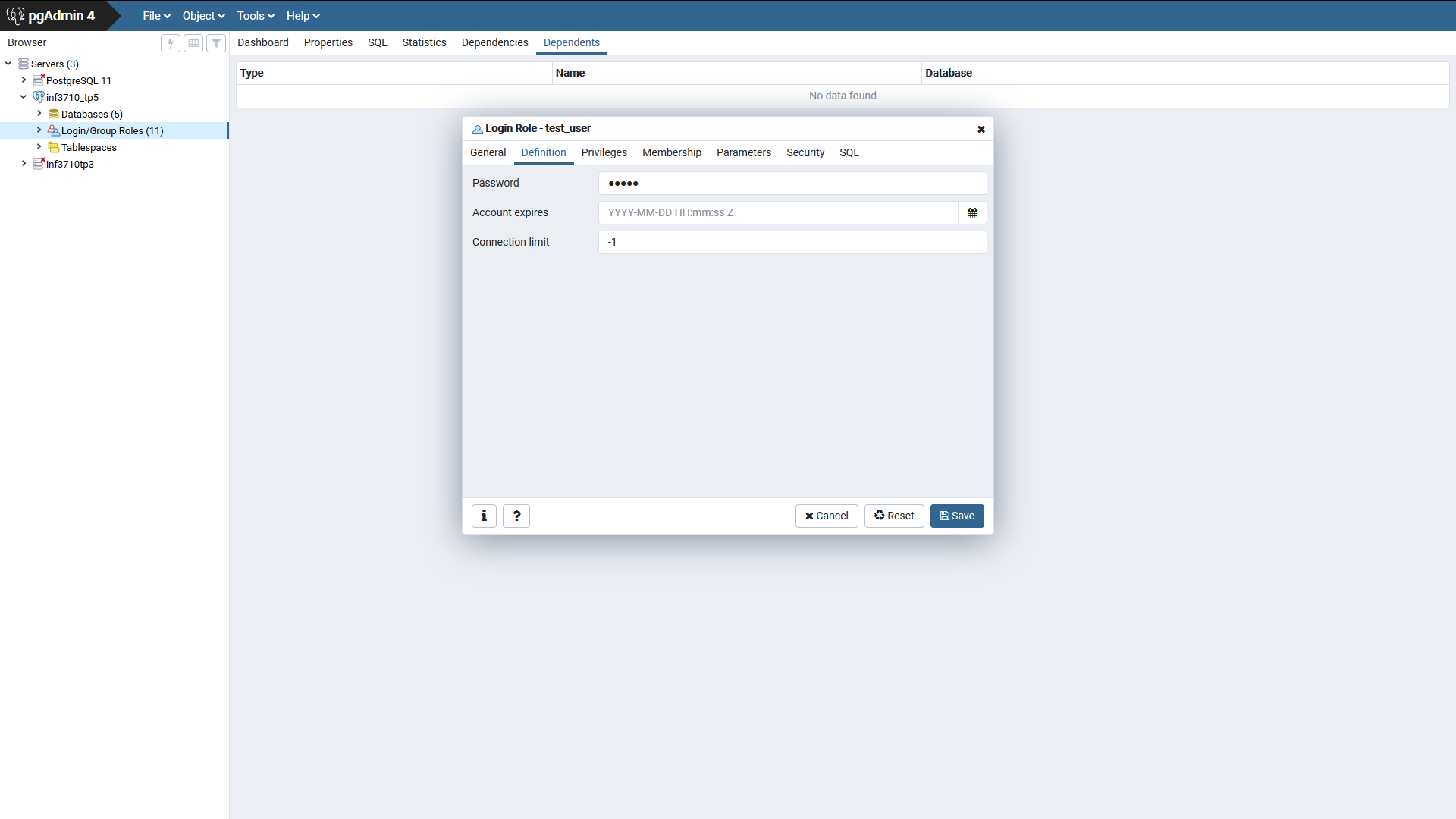




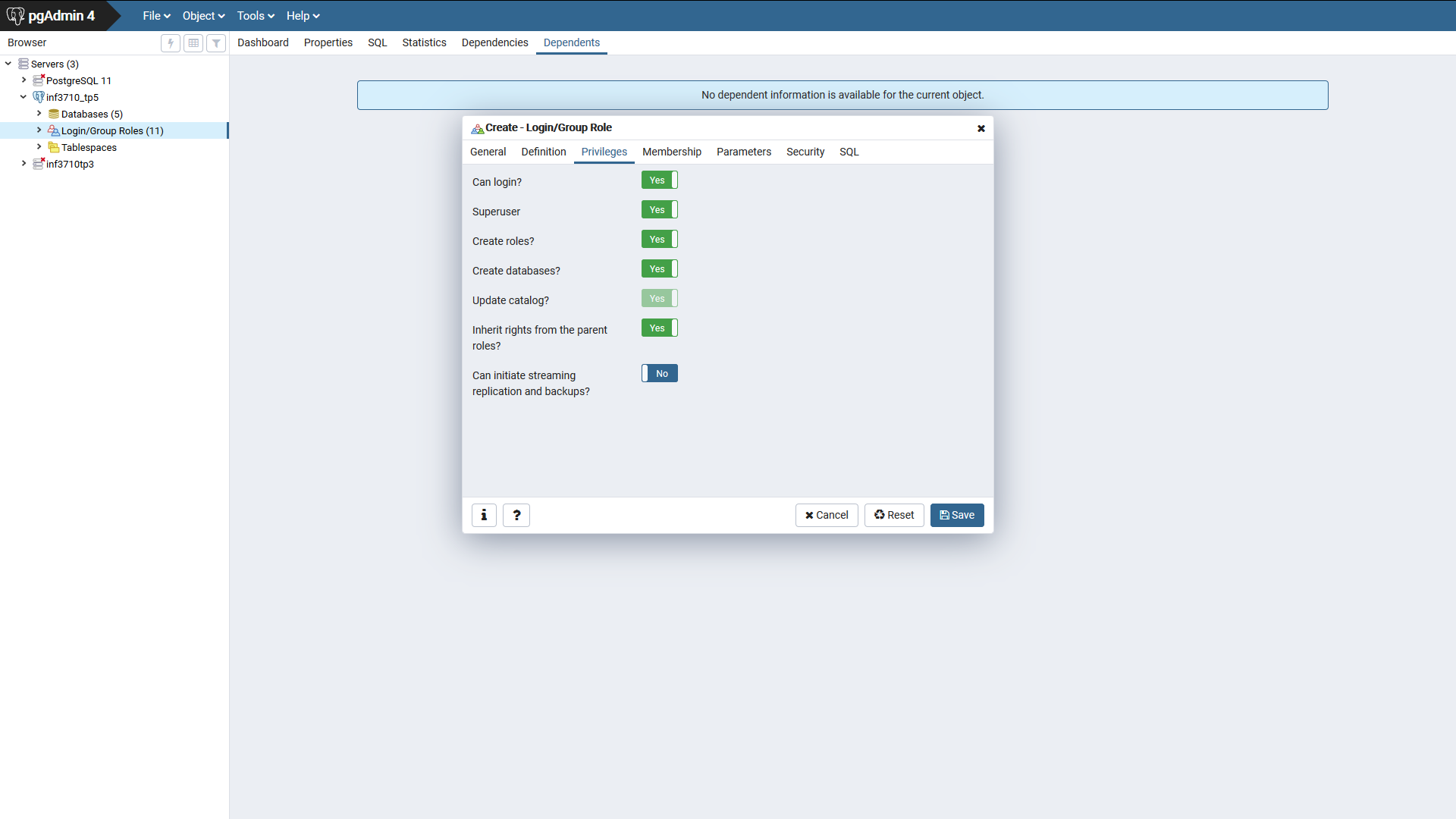
2) Veuillez créer un usager nommé ‘admin’’ dans l’onglet ‘’General’’ avec le mot de passe ‘’admin’’ dans le champ ‘’password’’ de l’onglet ‘’Definition’’.







Assurez-vous que l’usager ‘’admin’’ ait les droits suivants dans l’onglet ‘’Privileges’’ lors de la création de l’usager.

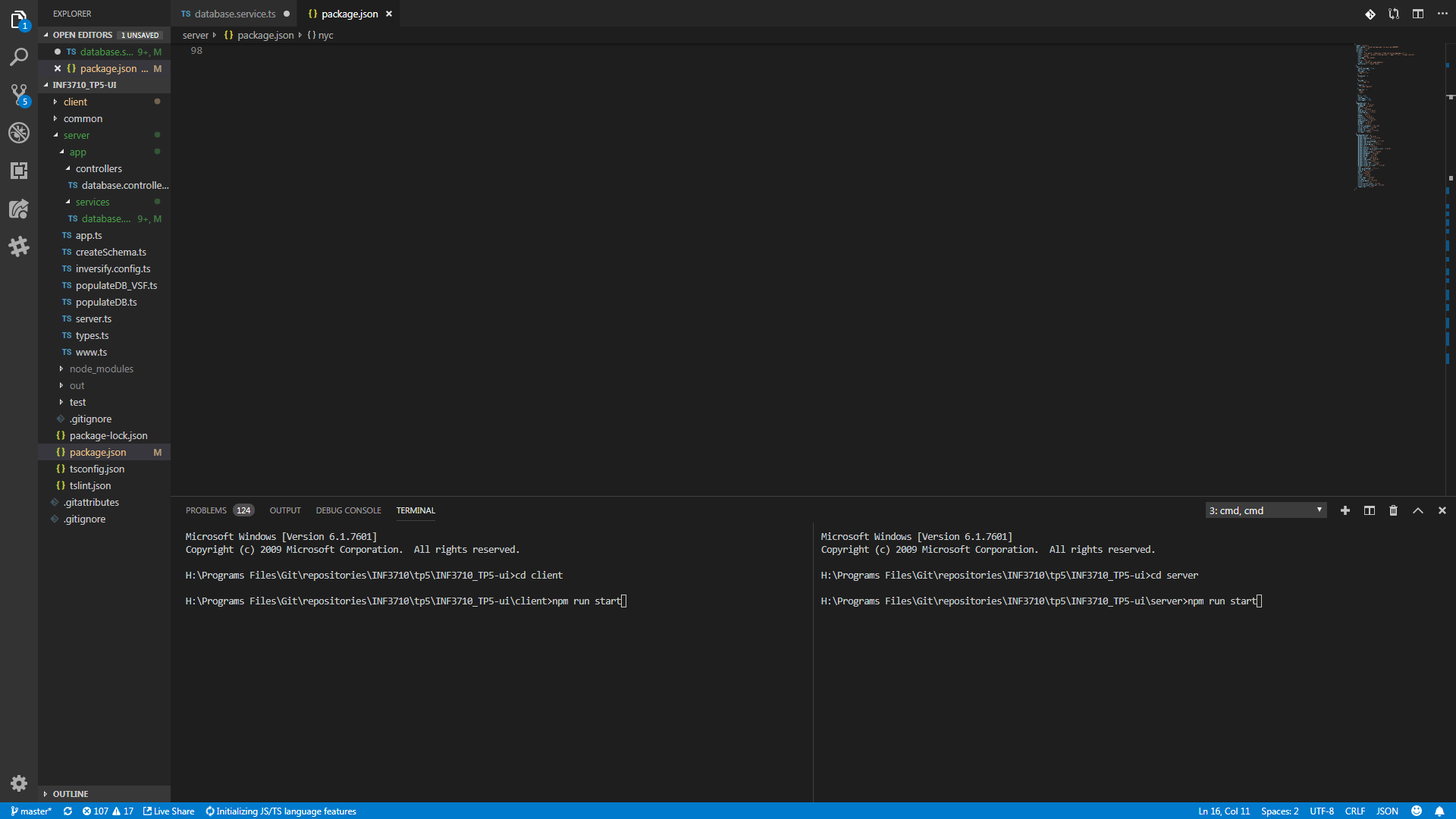


4) Ouvrez le dossier contenant l’application dans visual studio code. Ouvrez ensuite deux terminal dans les répertoires suivants:

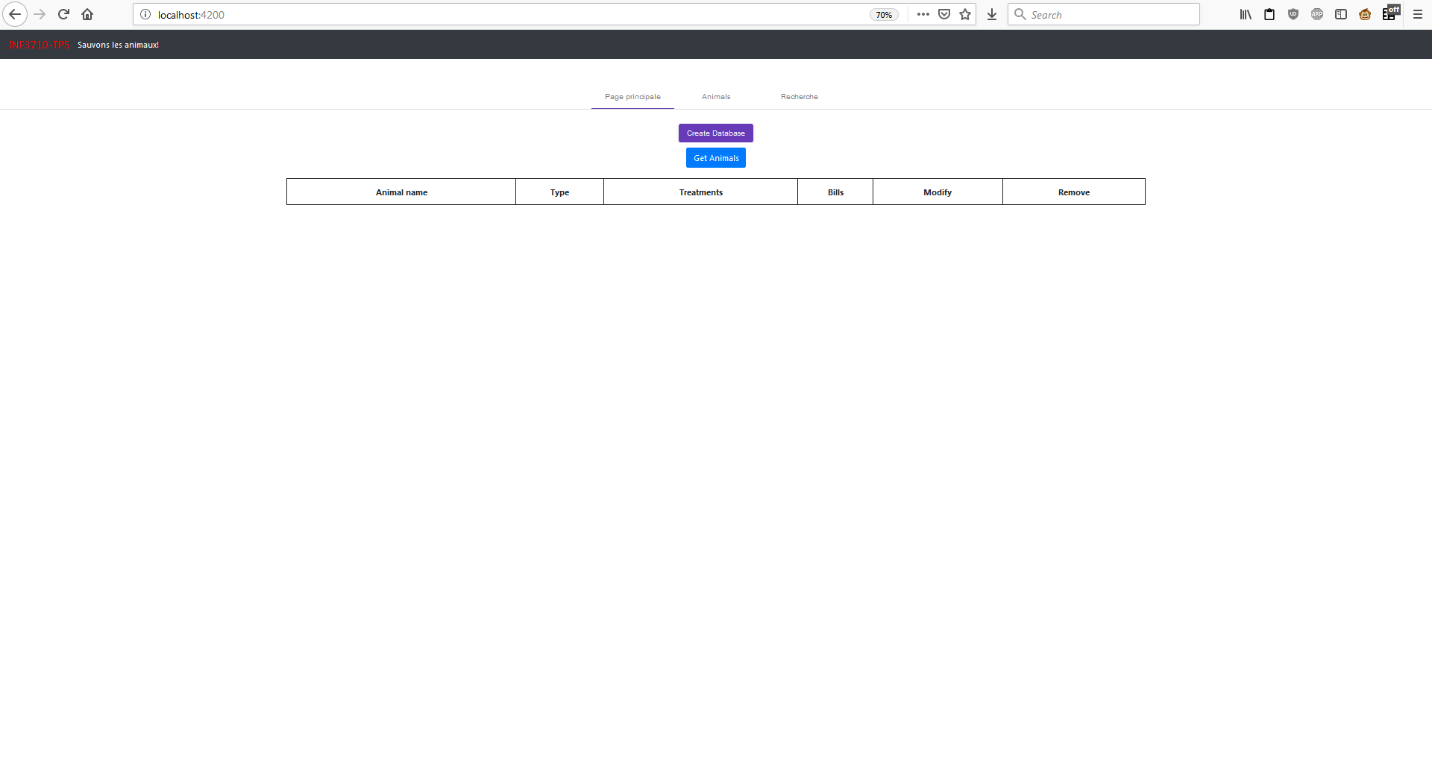
1)/client/

2)/server/

Entrez ensuite l’instruction ‘’npm run start’’ dans les deux terminal.

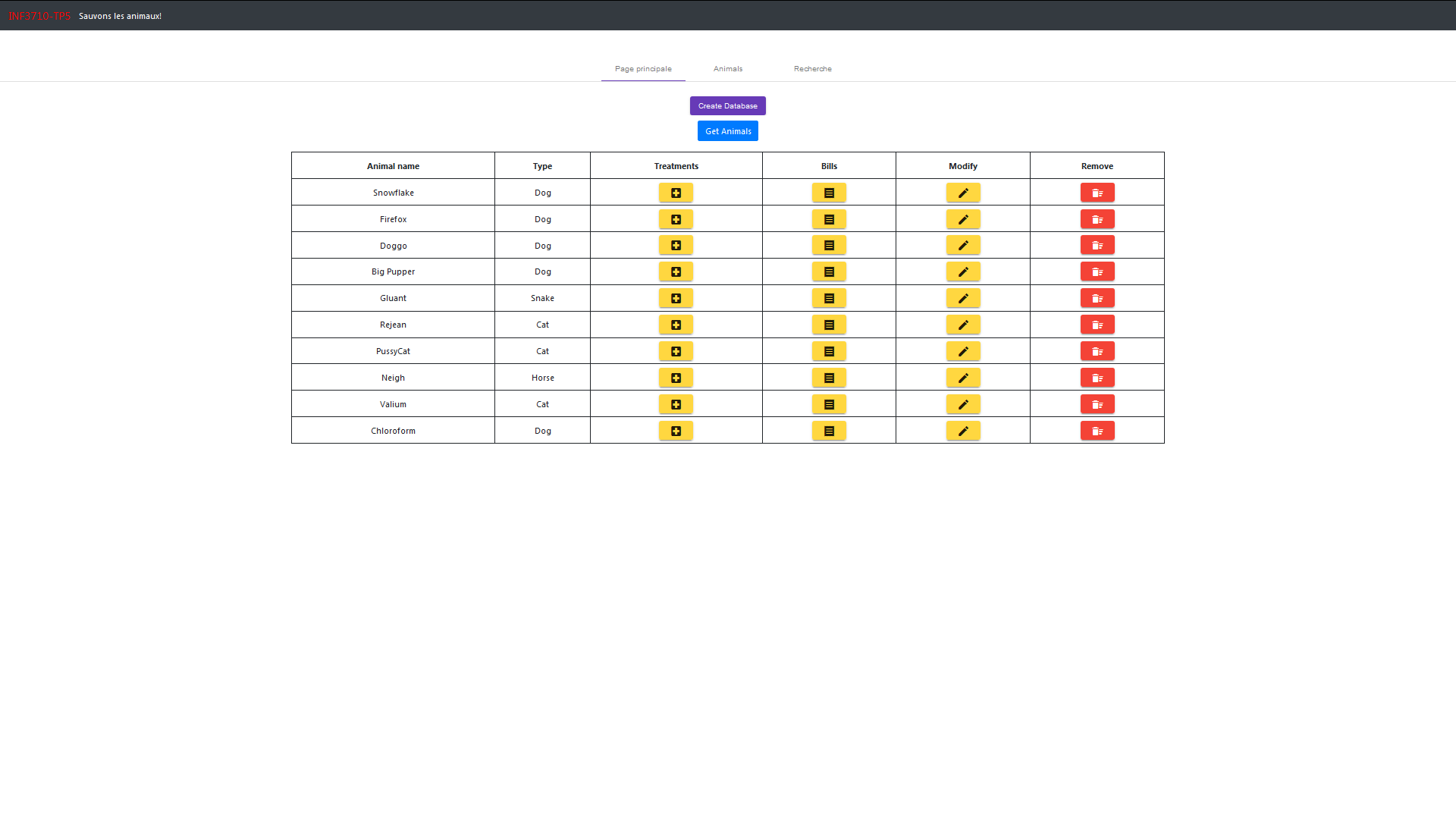


Si aucune page ne s’ouvre automatiquement, veuillez ensuite naviguer à l’adresse suivante: ‘‘http://localhost:4200/’’



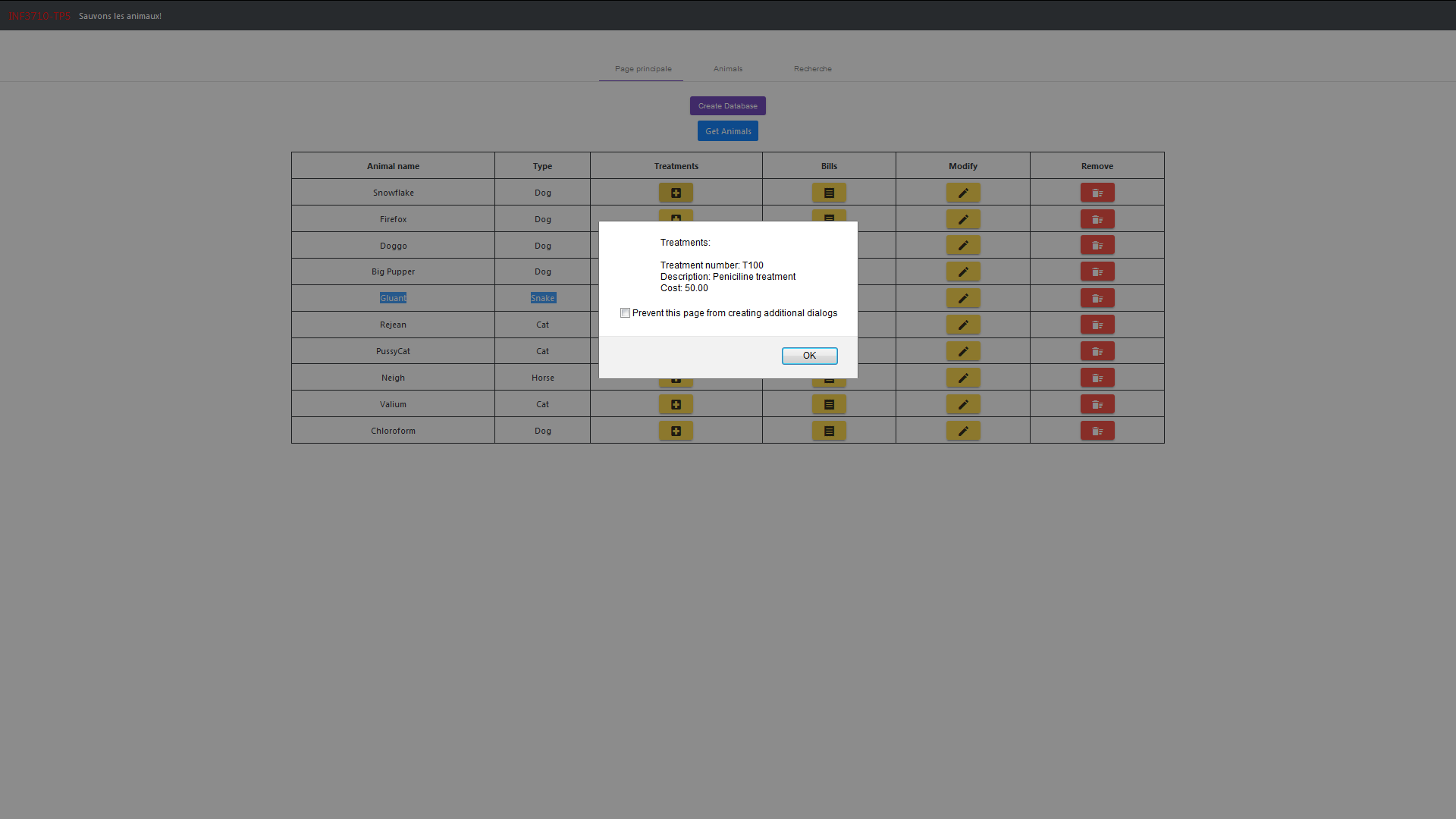
Si vous utilisez la base de données pour la première fois, il est nécessaire d’appuyer sur le bouton ‘’Create Database’’ pour créer la base de données.

Le bouton ‘’Get Animals’’ permet alors de présenter l’entièreté des animaux se retrouvant dans la base de données.

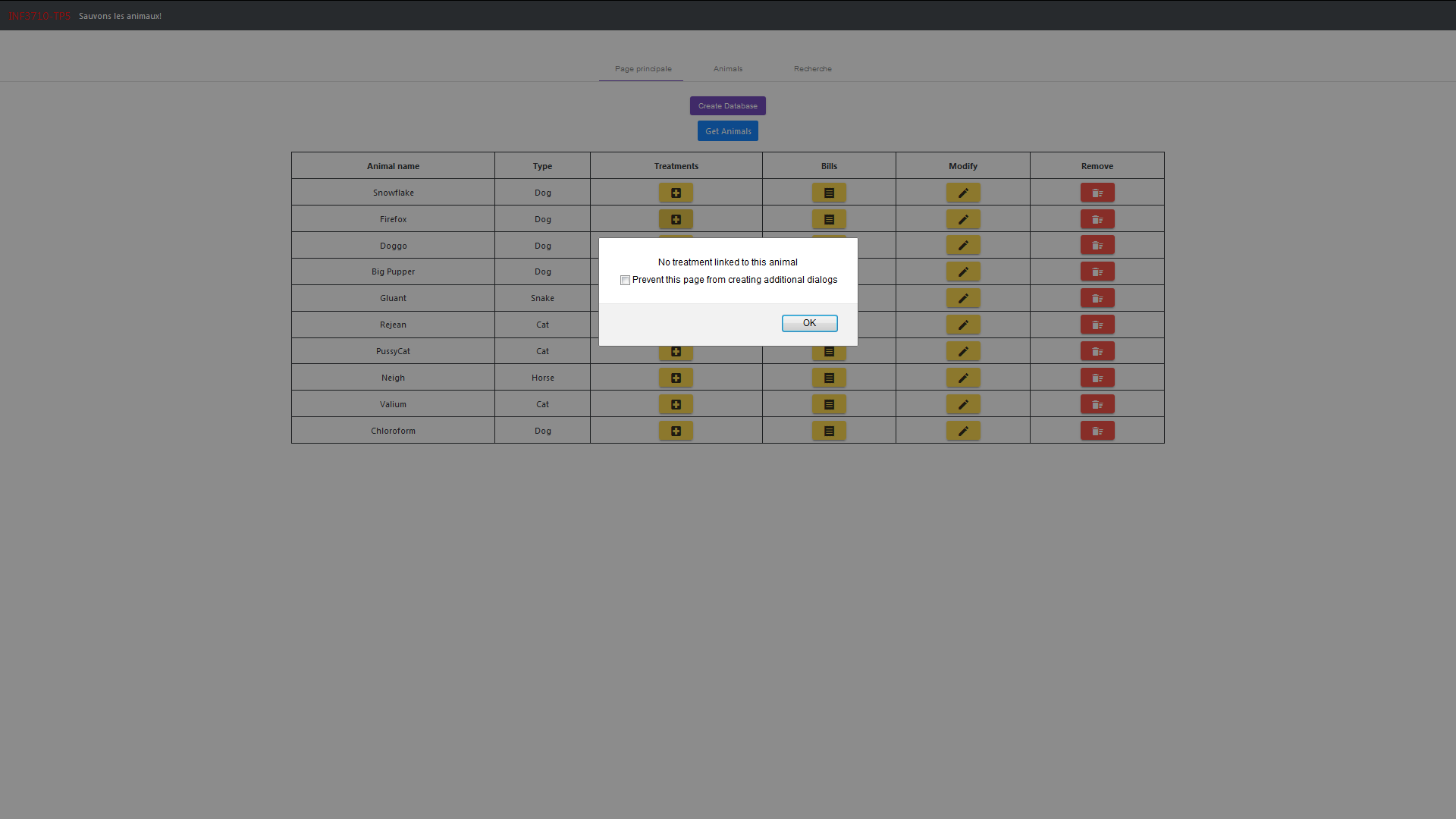


Il est également possible de naviguer dans l’application web à l’aide des onglets se retrouvant dans le haut de la page. On y retrouve les onglets ‘’Page principale’’, ‘’Animals’’ et ‘’Recherche’’. On était initialement à la page principale lors de l’arrivée sur le site de l’application web.

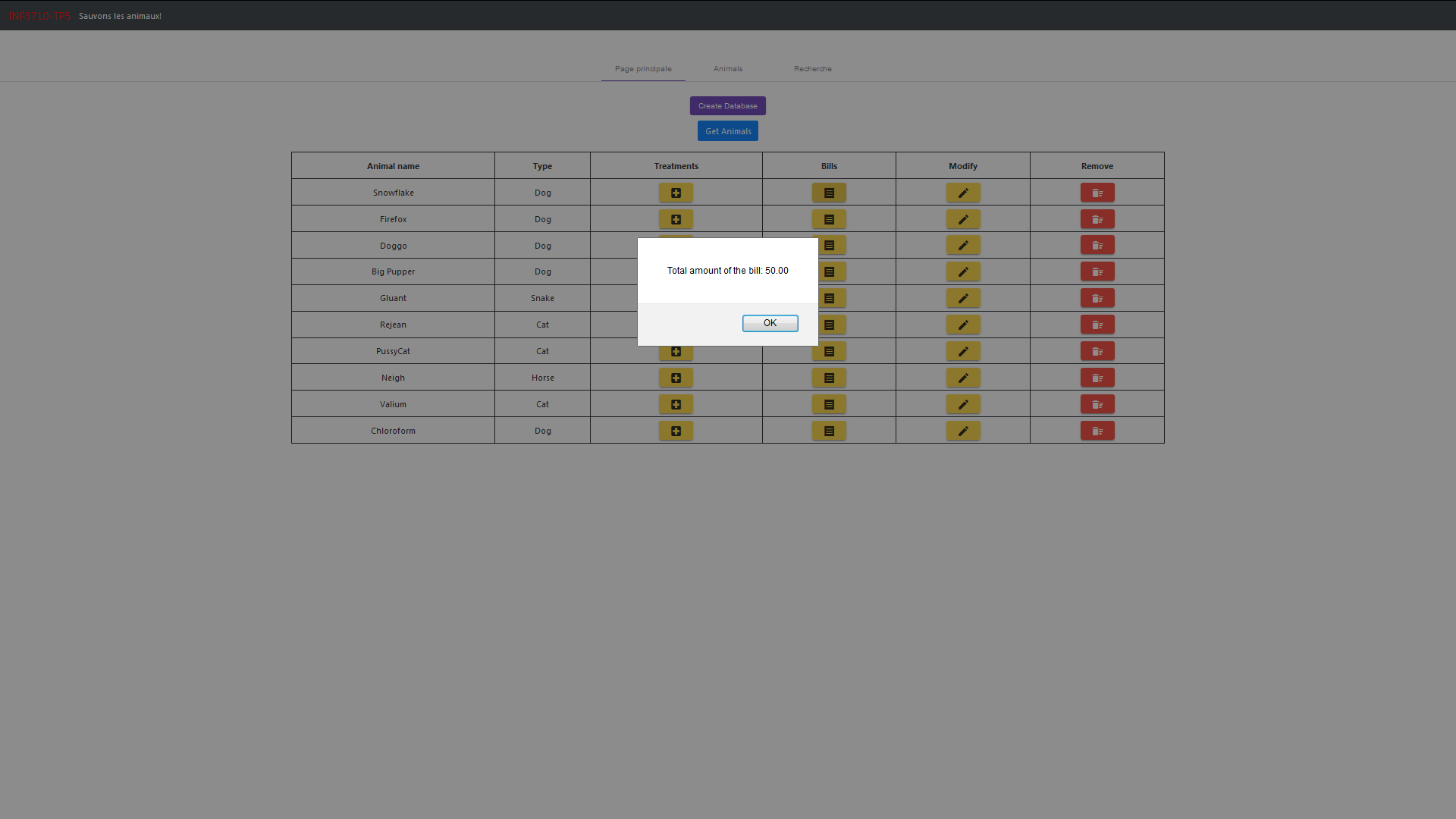
Dans la présentation des animaux dans la page principale, il est possible de générer plusieurs informations à propos de chaque animal inscrit. Un clic d’un des boutons dans la colonne ‘’Treatment’’ nous permet de générer une liste des traitements liés à un animal en particulier.

Voici un exemple de liste de traitement pour un animal ayant reçu des traitements :

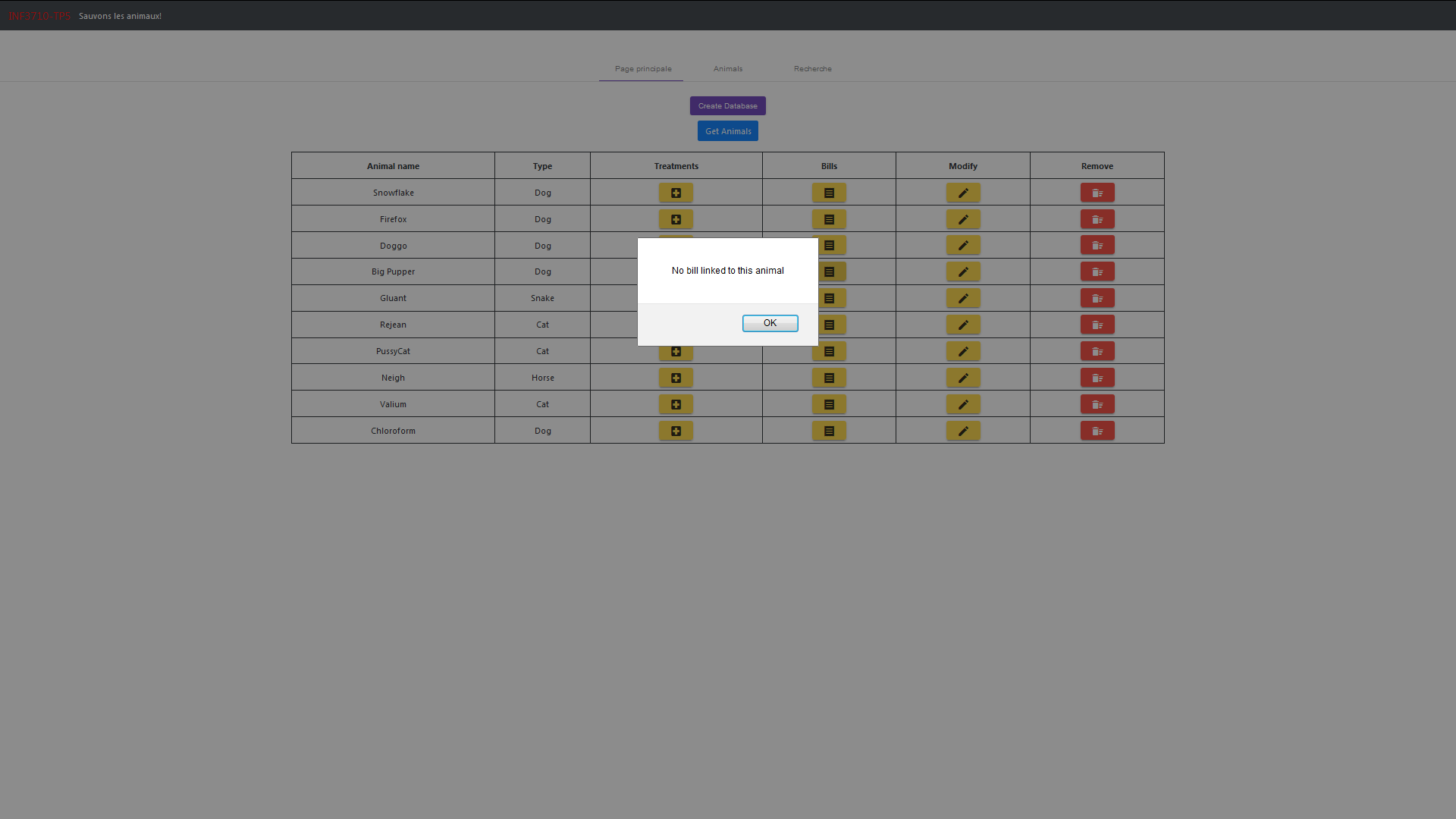
Voici un exemple de la génération d’une liste de traitement d’un animal n’ayant reçu aucun traitement :



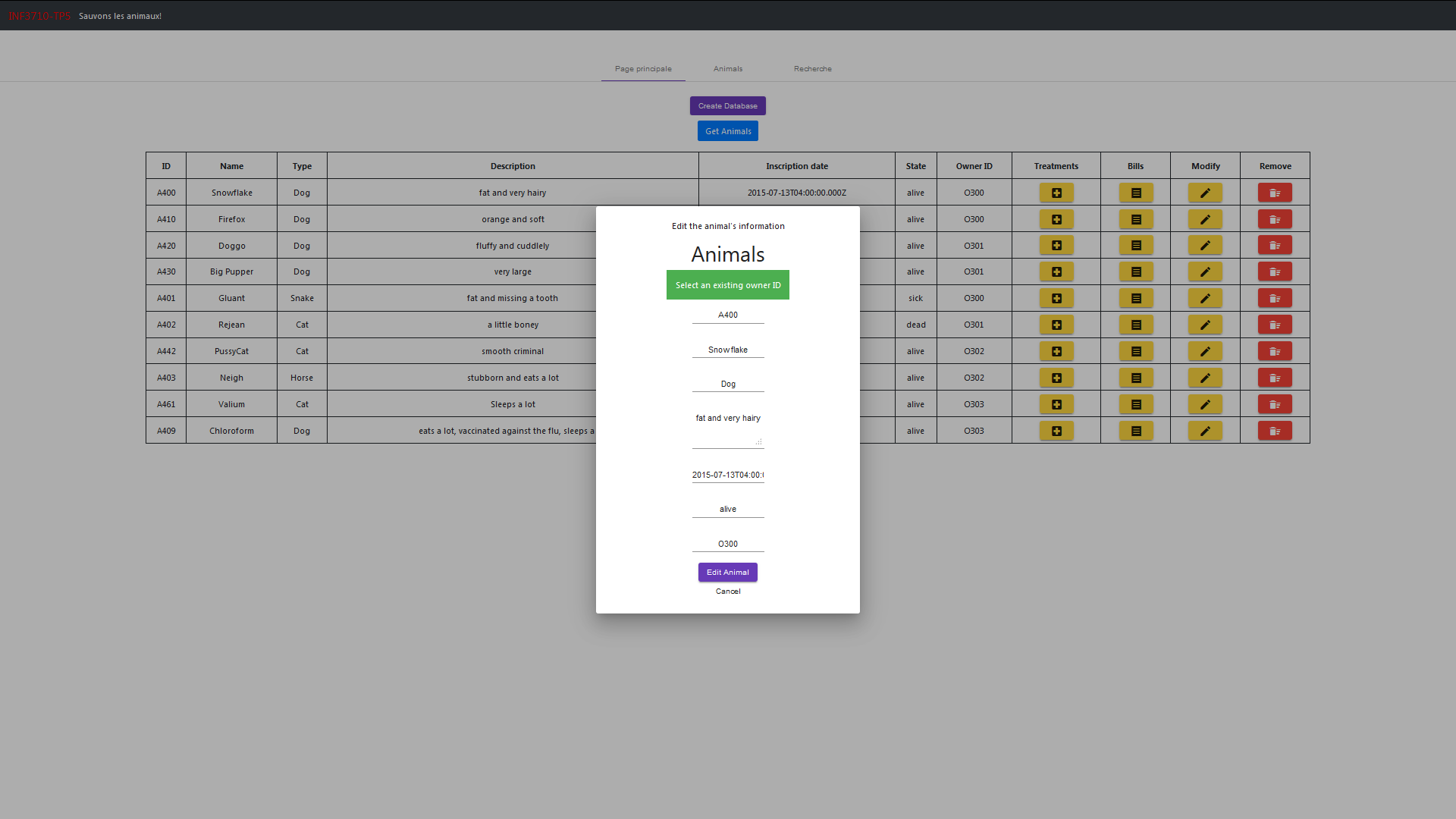
Un clic d’un bouton dans la colonne ‘’Bill’’ va permettre de générer la facture totale des traitements liés à un animal.

Voici un exemple de la génération de la facture des traitements d’un animal ayant reçu des traitements : 

Voici un exemple de la génération de la facture des traitements d’un animal n’ayant pas reçu de traitement :

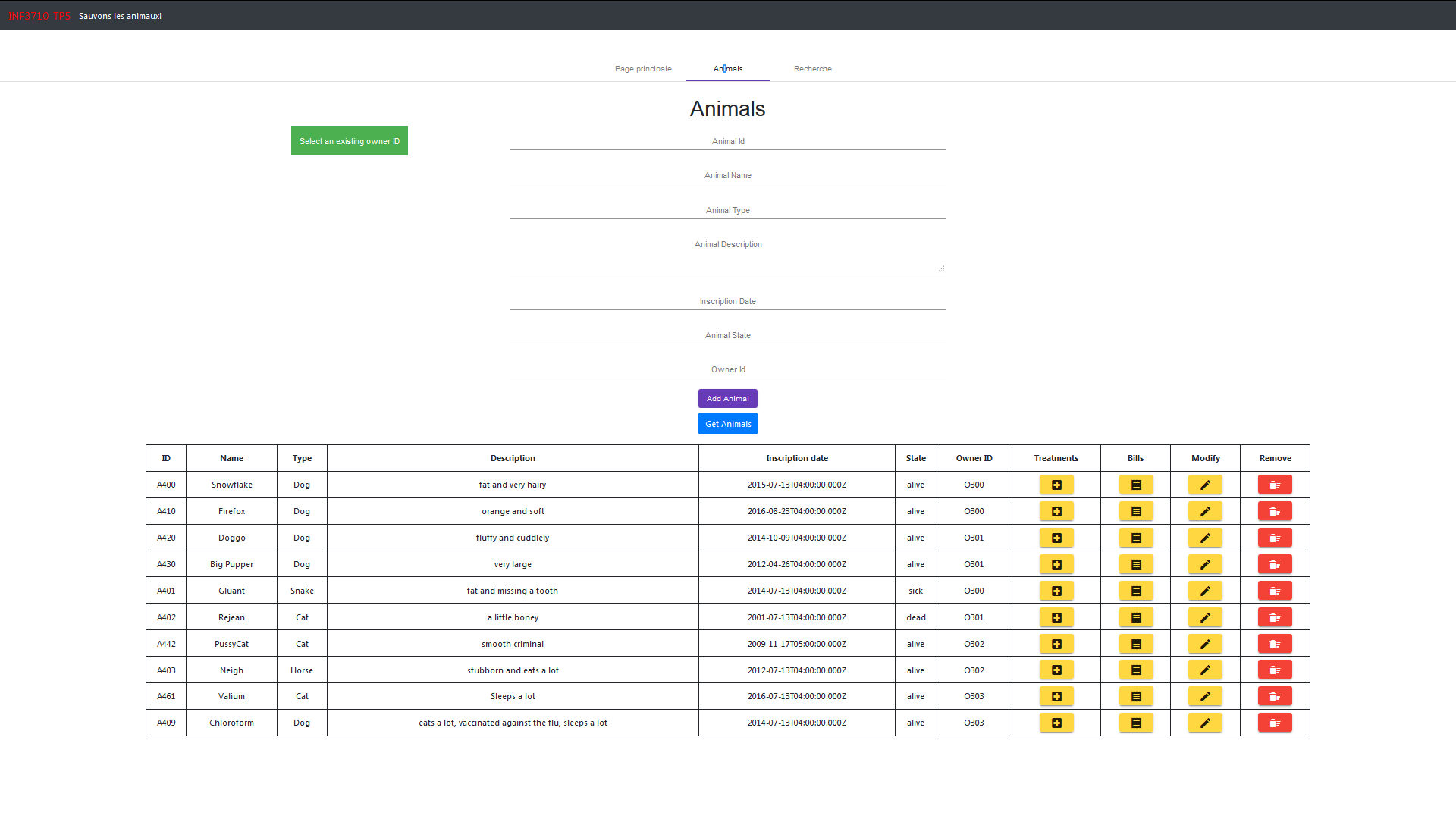


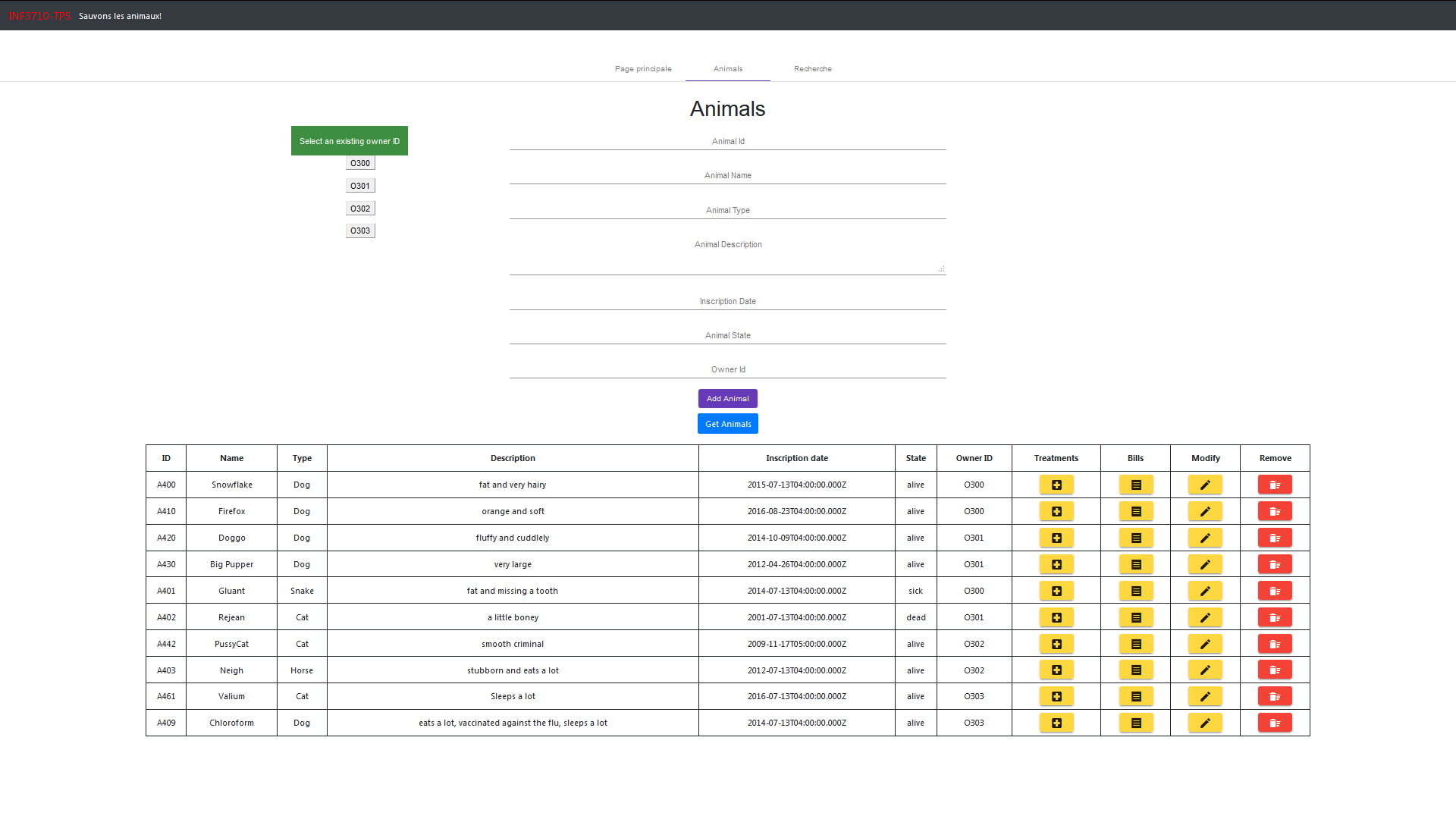
Un clic d’un bouton de la colonne ‘’Modifiy’’ va générer une fiche nous permettant de modifier les informations d’un animal.



Un clic d’un bouton de la colonne ‘’Remove’’ va supprimer l’animal de la base de données.

Dans l’onglet ‘’Animals’’, il est possible d’ajouter un animal dans la base de données. Il nous suffit que de remplir les champs et d’appuyer sur le bouton ‘’Add animal’’. De, plus, il est possible de sélectionner un propriétaire dans un menu déroulant.





L’onglet ‘’Recherche’’ permet d’effectuer une recherche des animaux inscrits dans la base de données sur la base d’une chaîne de caractères entrée par l’utilisateur.

