



INF3710 –Fichiers et Bases de données

Hiver 2021

TP No. 3

Groupe [1 ou 2 ou 3 ou 4]

2014604 – Nicholas Legrand

2016202 – Kervin Prinville

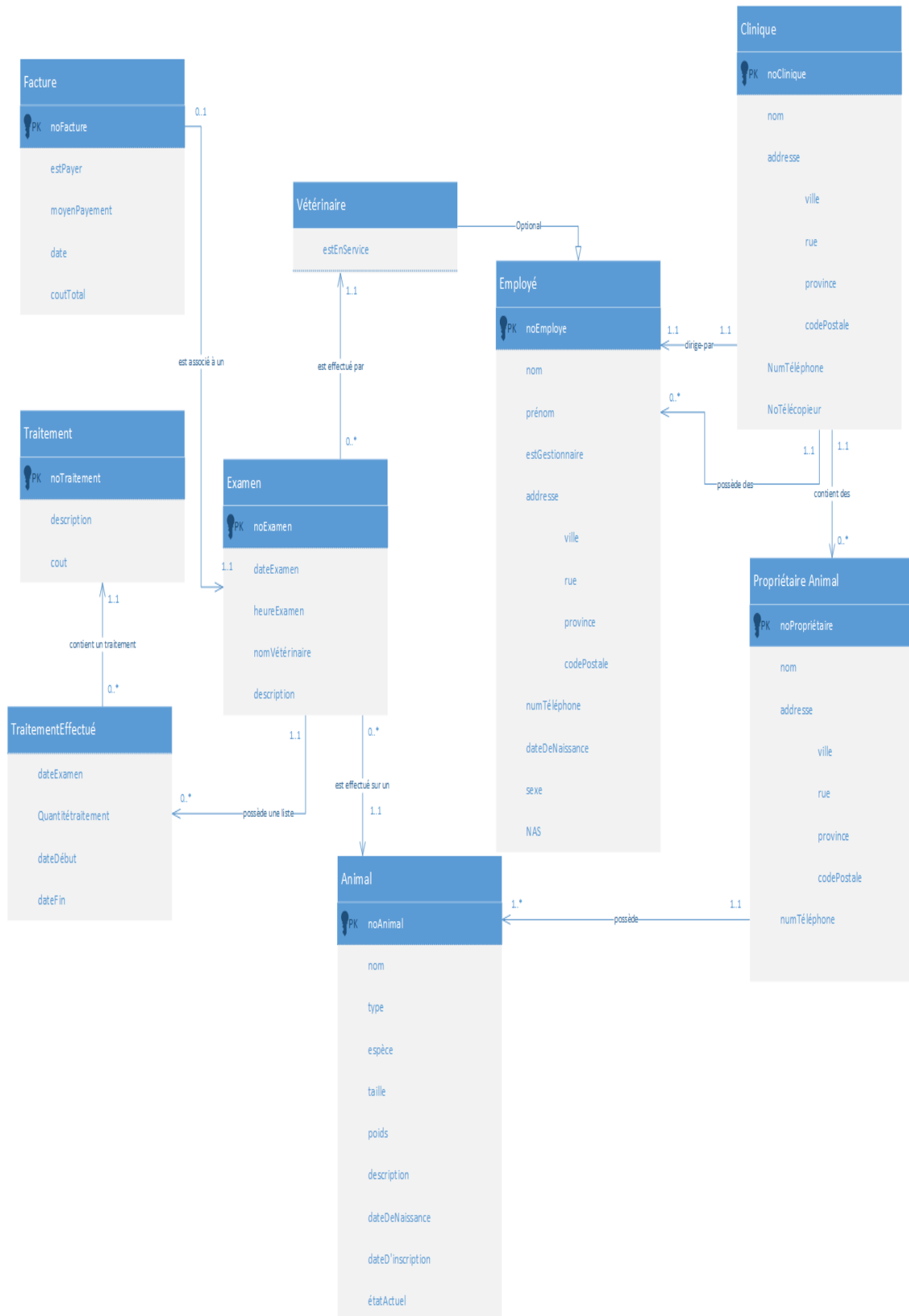
Soumis à : Amal Zouaq

17 avril 2021

Introduction

Le but de ce projet était de créer une application complète d'une organisation en se basant simplement sur une liste de requis donnés. Nous avons dû concevoir une base de données SQL qui respectait les demandes de l'organisation. Nous avons également dû créer une série de requête SQL qui permettait d'accomplir diverses actions importantes. Le projet devait être divisé en un front-end en utilisant Angular et un back-end en NodeJS. L'application permet à des clients d'entrée et modifier de l'information présent dans la base de données.

Modèle Conceptuel



Modèle Relationnel

- **Clinique** (noClinique, nom, rue, ville, province, codePostale, numTéléphone, numTélécopieur)
 - Primary Key : noClinique
- **ProprietaireAnimal** (noProprietaire, noClinique, nom, prenom, rue, ville, province, codePostale, numTéléphone)
 - Primary Key : (noProprietaire, noClinique)
 - Foreign Key : noClinique REFERENCES Clinique(noClinique)
Proprietaire Animal détient la clé principale de Clinique comme Foreign Key puisqu'il s'agit d'une relation de 0..* par rapport à une 1..1.
- **Animal** (noAnimal, noClinique, noProprietaire, nom, type, espece, taille, poids, description, dateNaissance, dateInscription, etatActuel)
 - Primary Key : (noAnimal, noClinique)
 - Foreign Key : (noProprietaire, noClinique) REFERENCES ProprietaireAnimal (noProprietaire, noClinique)
Animal détient la clé principale de Proprietaire Animal comme Foreign Key puisqu'un propriétaire peut posséder plusieurs animaux. C'est donc une relation de 1..* par rapport à 1..1.
- **Employe** (noEmploye, noClinique, nom, prenom, rue, ville, province, codePostale, numTelephone, dateNaissance, sexe, NAS, estGestionnaire)
 - Primary Key : noEmploye
 - Foreign Key : noClinique REFERENCES Clinique(noClinique)
Employe détient la clé primaire de Clinique comme foreign key puisqu'une clinique emploie plusieurs employés donc c'est une 1..* et un 1..1.
- **Veterinaire** (noEmploye, estEnService)
 - Primary Key Primary Key: noEmploye
 - Foreign Key: noEmploye REFERENCES Employe (noEmploye)
Vétérinaire détient la clé primaire d'employé comme foreign key puisqu'un vétérinaire est un employé et il y a une relation d'héritage.
- **Examen** (noExamen, noClinique, noAnimal, dateExamen, heureExamen, noVeterinaire, description)
 - Primary Key : (noExamen, noClinique)

- Foreign Key: noVeterinaire REFERENCES Veterinaire(noEmploye)
- Foreign Key: noAnimal,noClinique REFERENCES Animal(noAnimal, noClinique)
Examen détient les clé primaires d'animal et vétérinaire
puisque chaque examen est associé à un vétérinaire et un Animal.
Ce qui cause une relation de 0..* à 1..1 dans les deux cas.

- **Traitement (noTraitement, description, cout)**

- Primary Key: noTraitement

- **Facture (noFacture, estPaye, noExamen, noClinique, moyenPayement, date, coutTotal)**

- Primary Key : noFacture
- Foreign Key: noExamen, noClinique REFERENCES Examen (noExamen, noClinique)

Facture détient la clé primaire d'Examen puisque chaque Facture est associé à un examen et il y a 0..1 facture pour 1..1 examen.

- **TraitementEffectue(noExamen, noClinique, noTraitement, quantiteTraitement, dateDebut, dateFin)**

- Primary Key : (noExamen, noTraitement)
- Foreign Key : noExamen REFERENCES Examen(noExamen)
- Foreign Key : noTraitement REFERENCES Traitement(noTraitement)

TraitementEffectue détient les clés primaires de noExamen car chaque traitement est associé à un examen et il peut y avoir plusieurs traitements par examen.
La table traitementEffectue possède la clé primaire de la table traitement car chaque traitementEffectué est associé à un traitement.

Dépendance fonctionnelle

- **Clinique (noClinique, nom, rue, ville, province, codePostale, numTéléphone, numTélécopieur)**

DF noClinique -> nom, rue, ville, province, codePostale, numTéléphone, numTélécopieur

DF codePostale -> rue, ville, province

- **ProprietaireAnimal** (noProprietaire, noClinique, nom, prenom, rue, ville, province, codePostale, numTéléphone)
 - DF noProprietaire,noClinique -> , nom, prenom, rue, ville, province, codePostale, numTéléphone
 - DF codePostale -> rue, ville, province
- **Animal** (noAnimal, noClinique, noProprietaire, nom, type, espece, taille, poids, description, dateNaissance, dateInscription, etatActuel)
 - DF noAnimal, noClinique -> , noProprietaire, nom, type, espece, taille, poids, description, dateNaissance, dateInscription, etatActuel
 - DF espece->type
- **Employe** (noEmploye, noClinique, nom, prenom, rue, ville, province, codePostale, numTelephone, dateNaissance, sexe, NAS, estGestionnaire)
 - DF noEmploye, noClinique-> , nom, prenom, adresse, numTelephone, dateNaissance, sexe, NAS, estGestionnaire
 - DF NAS -> noEmploye, noClinique, nom, prenom, adresse, numTelephone, dateNaissance, sexe, estGestionnaire
 - DF codePostale -> rue, ville, province
- **Veterinaire** (noEmploye, estEnService)
 - DF noEmploye->estEnService
- **Examen** (noExamen, noClinique, noAnimal, dateExamen, heureExamen, noVeterinaire, description)
 - DF noExamen, noClinique -> noAnimal, dateExamen, heureExamen, noVeterinaire, description
- **Traitement** (noTraitement, description, cout)
 - DF noTraitement ->description, cout
- **Facture** (noFacture, estPaye, noExamen, noClinique, moyenPayement, date, coutTotal)

DF Facture noFacture -> estPaye, noExamen, noClinique, moyenPayement, date, coutTotal

DF noExamen,noClinique-> noFacture , estPaye, moyenPayement, date, coutTotal

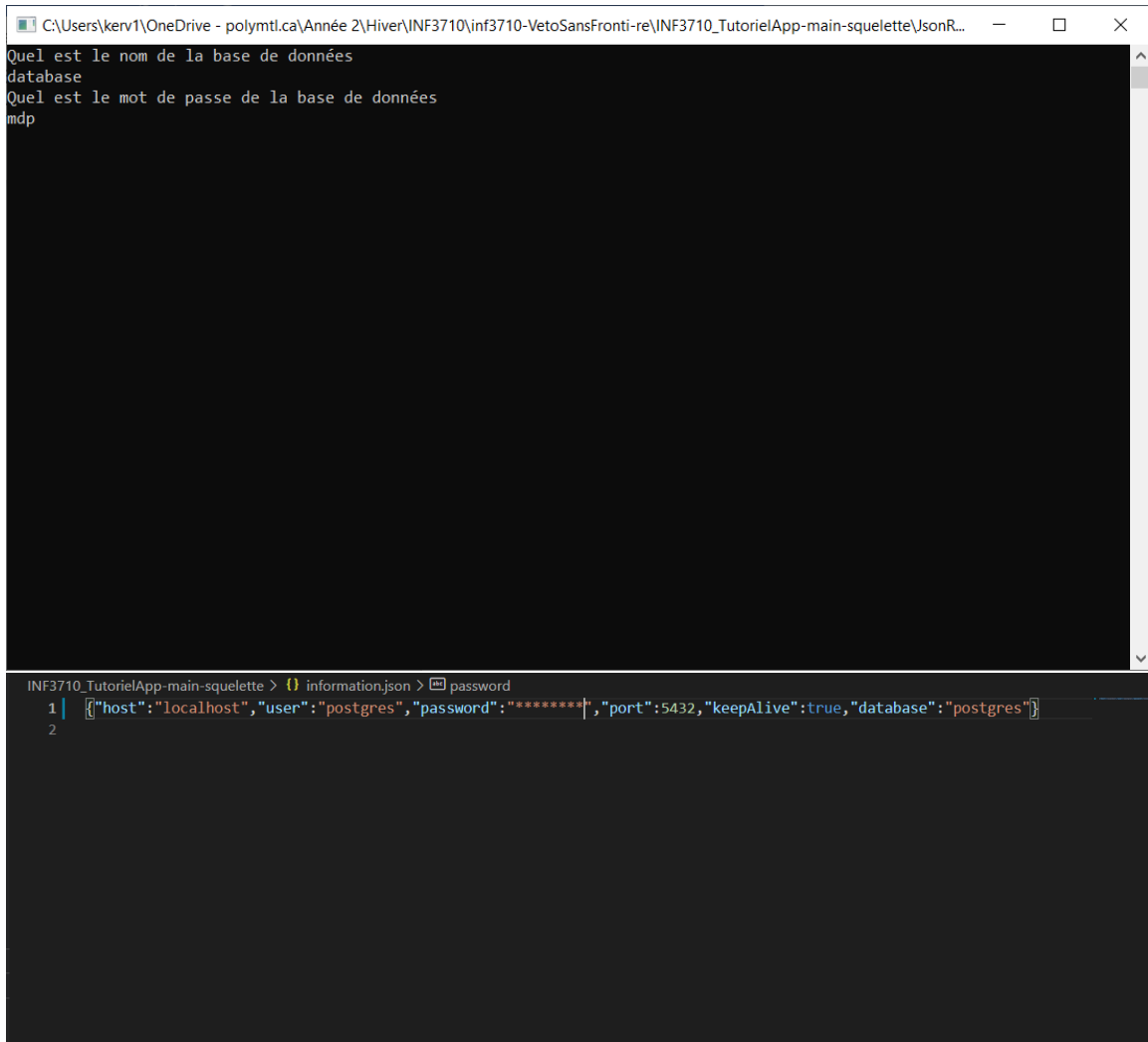
- **TraitementEffectue**(noExamen, noClinique, noTraitement, quantiteTraitement, dateDebut, dateFin)

DF noExamen, noClinique -> noTraitement, quantiteTraitement, dateDebut, dateFin

Notre base de données est en 2NF. La première raison est qu'elle est considérée comme une base de données en 1NF car tous les attributs de toutes les tables sont atomique. C'est-à-dire qu'il y a aucun attribut qui est composé de plusieurs éléments. La deuxième raison explique pourquoi notre base de données est une normale de deuxième forme et non de troisième forme est la présence des dépendances partielles dans la table clinique, propriétaire animal, animal et Employé. Les dépendances transitives sont la relation entre codePostal et le reste des informations de l'adresse. Le code postal permet de déduire le reste des informations par rapport à l'adresse même si le codePostale n'est pas la clé primaire de sa table.

Une autre dépendance transitive qui empêche la base de données d'être une 3NF est la relation entre type et espèce dans la table Animal parce que pour chaque espèce il y a un type et l'espèce n'est pas la clé primaire de la table. La base de données est donc de deuxième forme normale.

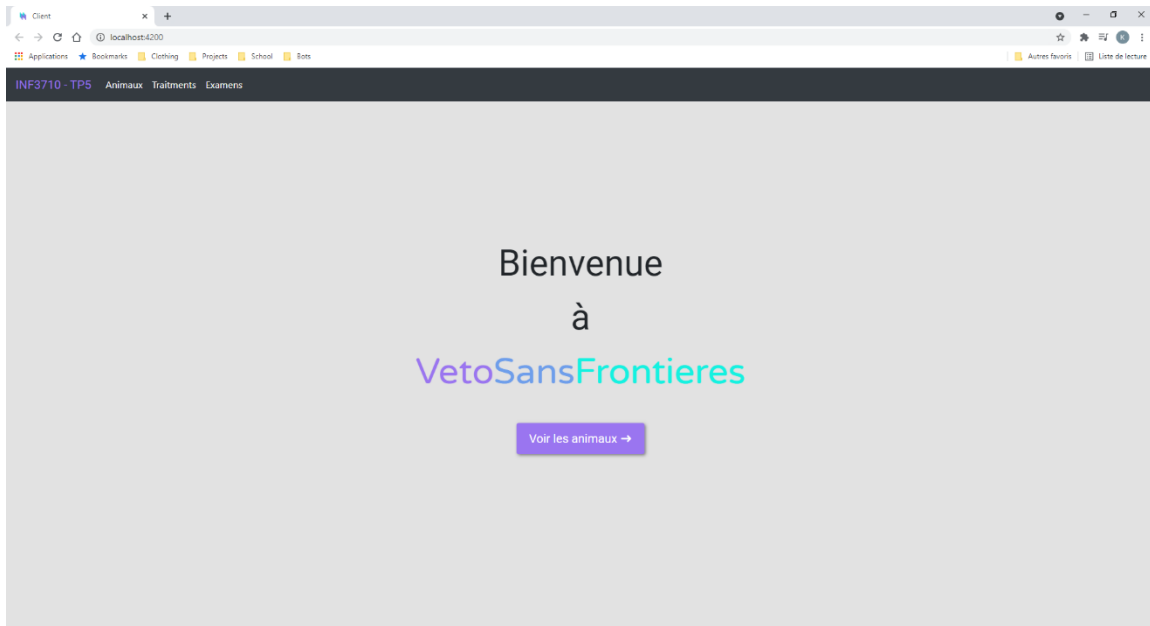
Présentation de l'application



```
C:\Users\kerv1\OneDrive - polymtl.ca\Année 2\Hiver\INF3710\inf3710-VetoSansFronti-re\INF3710_TutorielApp-main-squelette\JsonR...
Quel est le nom de la base de données
database
Quel est le mot de passe de la base de données
mdp

INF3710_TutorielApp-main-squelette > {} information.json > password
1 | [{"host": "localhost", "user": "postgres", "password": "*****", "port": 5432, "keepAlive": true, "database": "postgres"}]
2 |
```

Afin de connecter plus facilement le serveur à la base de données sur windows, nous avons créé un programme qui permet à un utilisateur de facilement entrer les informations de la base de données pour automatiquement créer les un fichier json d'informations de connections que le serveur pourra facilement lire.



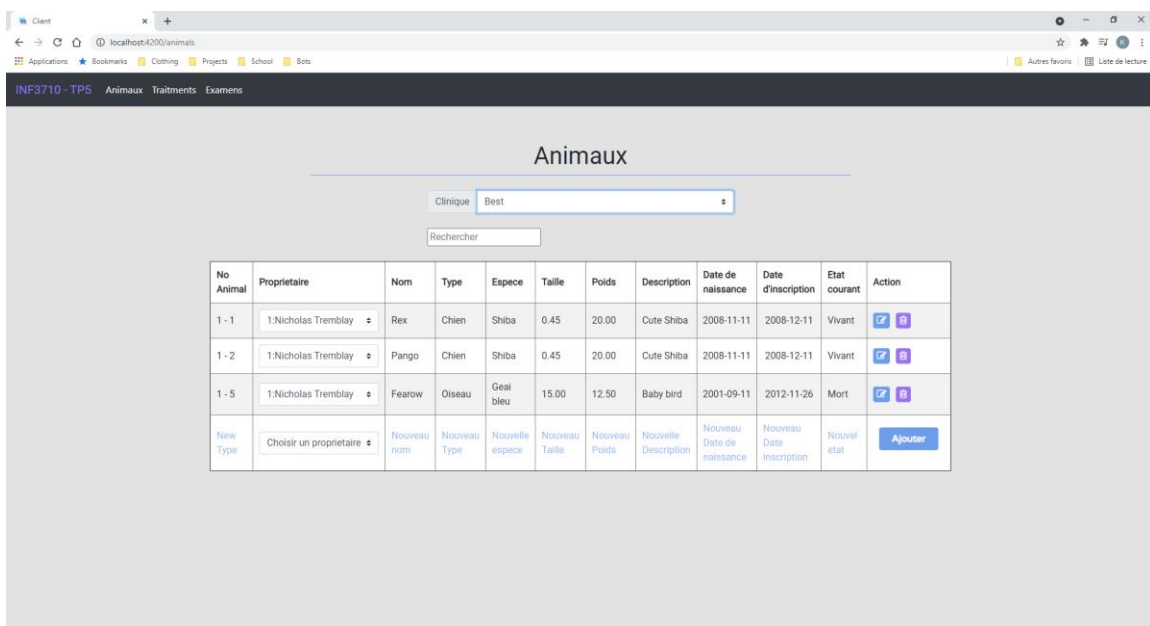
Ceci est la page principale, inspirée du site démo Hotels. C’est sur cette page que le site s’ouvre.

En haut complètement, il y a la barre de navigation avec les boutons suivants :

- INF3710-TP5, qui ramène l’utilisateur vers la page principale
- Animaux, qui amène vers la page des animaux
- Traitements, qui amène vers la page avec la liste des traitements d’un animal
- Examens, qui permet de voir la liste des examens effectués par un animal, ainsi que de générer les factures associées.

Cette barre de navigation apparaîtra sur toutes les pages du site.

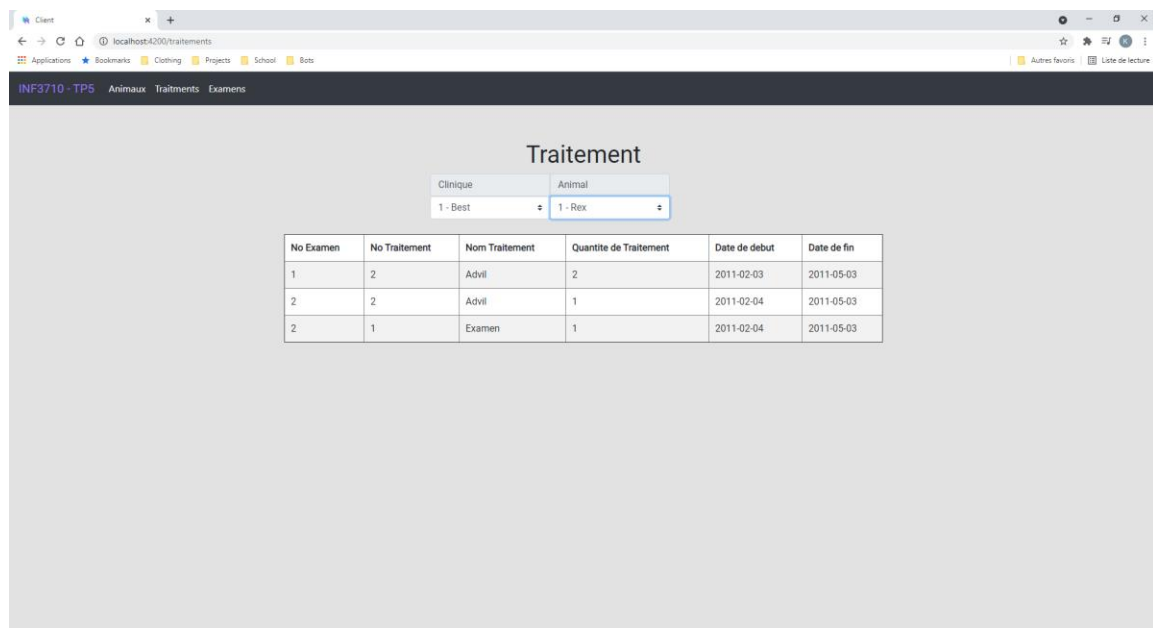
Au milieu de l’écran, il y a aussi un bouton qui amène vers la page des animaux.



La page des animaux permet de voir, de modifier, et de supprimer les informations des animaux dans la base de données. Elle permet aussi d'ajouter de nouveaux animaux dans la base de données.

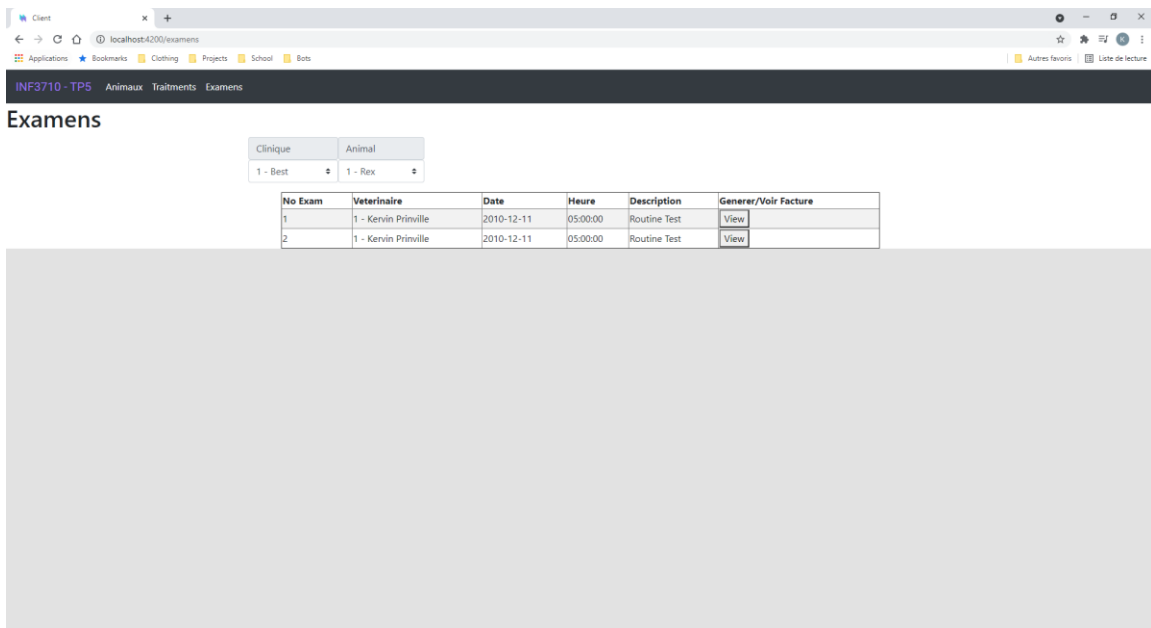
Pour voir les animaux, il est possible de sélectionner la clinique désirée, et cela affichera tous les animaux de cette clinique. Pour faciliter la recherche, il est possible d'utiliser la barre de recherche pour trouver les animaux à partir de leur nom.

Afin d'aider à la recherche la requête n'est pas sensible aux majuscules et minuscules.

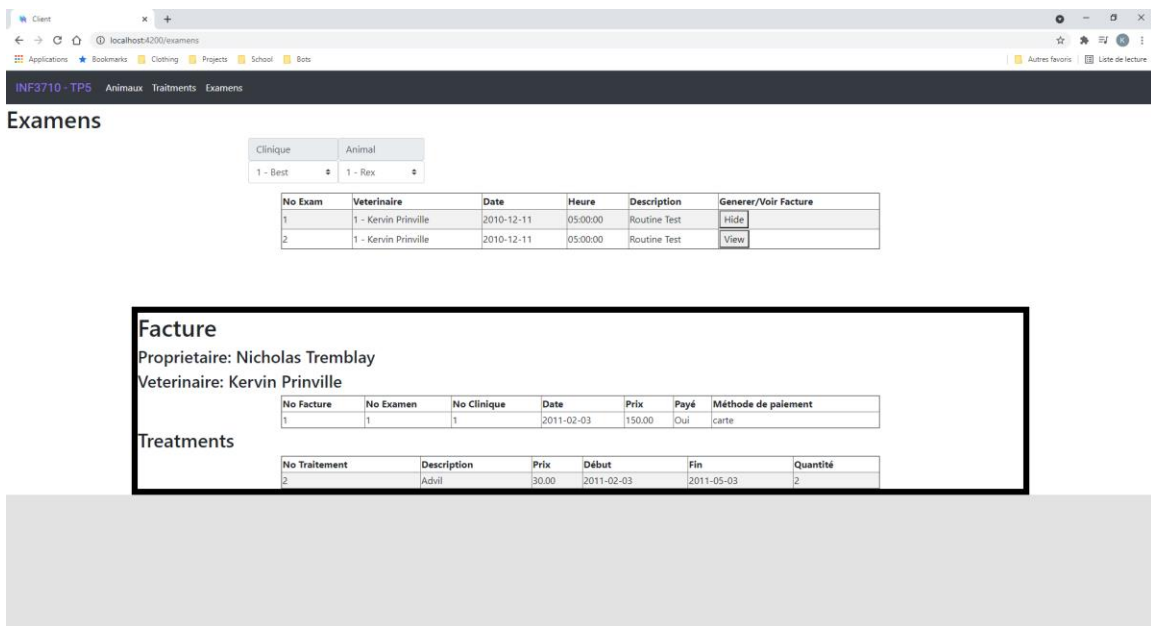


No Examen	No Traitement	Nom Traitement	Quantite de Traitement	Date de debut	Date de fin
1	2	Advil	2	2011-02-03	2011-05-03
2	2	Advil	1	2011-02-04	2011-05-03
2	1	Examen	1	2011-02-04	2011-05-03

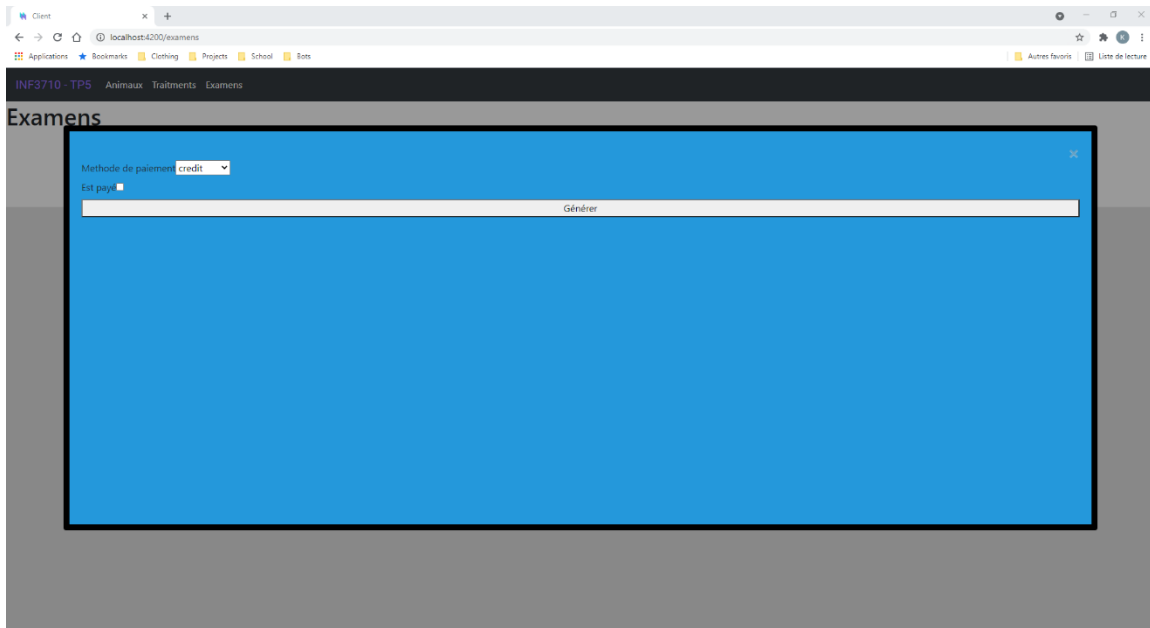
Cette page permet de voir la liste des traitements qui ont été donnés à un animal particulier. Pour sélectionner un animal, il faut tout simplement choisir une clinique à partir du menu déroulant correspondant, puis un animal. La liste de traitements s'affichera ensuite.



La page examens permet de voir la liste des examens, ainsi que de voir les factures qui leur sont associées, ou d'en générer une si elle n'existe pas encore. Il n'affiche qu'une seule facture à la fois.



Pour générer une facture, il faut juste appuyer sur le bouton générer à côté d'un examen, et une boîte demandera les informations manquantes (moyen de paiement, est-ce que la facture a déjà été payée).



Guide d'installation

INF3710_TutorielApp

INF3710

Avant de lancer le projet

- Assurez-vous que Postgres roule sur vos machines
- Vérifiez que vous avez NodeJs installé avec `node -v` en ligne de commande, si vous ne l'avez pas fait, veuillez suivre les étapes dans les dispo du labo
- Allez dans `/client` et lancez `npm install` en ligne de commande
- Allez dans `/server` et lancez `npm install` en ligne de commande
- Si vous êtes sur Windows, exécutez `JSONReader.exe` et remplissez les informations de la DB afin de générer le fichier d'informations.
- Sinon, entrez manuellement les informations dans le fichier `information.json`

Pour lancer le projet

- Allez dans `/server` et faites `npm start` en ligne de commande
- Allez dans `/client` et faites `npm start` en ligne de commande

Screenshots