Projet
Gestion d’un centre d’activité

Rapport de projet info0303

Réalisé par :

Eliot CALDERON-Y-MORA

Marceau CONRAUX

Groupe : S3F5B

Table des matières

Introduction

Dans le cadre de ce troisième semestre dans le module info0303 : Technologie Web 2, un projet nous a été proposé faisant appel aux compétences acquises durant les cours en particulier le langage PHP et le framework Laravel.

Le projet est un site permettant la gestion d'un centre loisir proposant aux parents ou tuteurs d'inscrire leurs enfants à une ou plusieurs activités dans la semaine via ne plateforme simple d'utilisation.

Cahier des charges

Nous avons voulu simplifier le travail administratif d'inscription pour les gérants de centre loisir ou d'école par exemple, plus besoin de passer par un formulaire à remplir avec des documents à envoyer par la poste. Cela simplifiera aussi les utilisateurs qui pourront obtenir toutes les informations qu'ils recherchent simplement en allant sur la page du centre.

Objectifs

A la finalité de ce projet, un visiteur pourra s'inscrire sur le site via le champ d'enregistrement dans la barre de navigation, il pourra également voir toutes les activités proposées par le centre et voir les horaires, le nombre de place et un descriptif de l'activité.

Un utilisateur inscrit, pourra lui ajouter son ou ses enfant(s) et les inscrire à une ou plusieurs, il a aussi la possibilité de désinscrire son enfant d'une activité. Il peut supprimer un de ses enfants de la plateforme et consulter/ modifier ces informations.

L'administrateur du site à la possibilité de créer des activités, d'attribuer à ces activités des horaires, il peut également modifier ou supprimer une activité. Le compte administrateur a été pensé pour être utilisé à l'accueil du centre, car un tuteur peut faire inscrire son enfant sans passer par le site mais via une personne associée au centre (secrétariat). C'est pour cela que l'administrateur à la possibilité d'ajouter un enfant, lui attribuer un utilisateur (tuteur) seulement si un compte tuteur est déjà créé, il peut aussi le modifier et agir comme un utilisateur.

Contraintes

Nous avons plusieurs contraintes dans le cadre de ce projet :

Les associations

- Chaque activité doit avoir un horaire.
- Les enfants doivent être associés à leurs tuteurs et un tuteur doit être capable de voir tous ses enfants.
- Un horaire doit être attribué à une activité via une table pivot, dans le activite.create.
- Lors de l'ajout d'un enfant par l'administrateur, le tuteur attribué ne peut pas être le compte administrateur, il faut que l'admin ait la possibilité de choisir un tuteur.
- Les activités sont attribuées à des enfants via une table pivot, dans le formulaire du enfant.create.

La visibilité

- Un visiteur ne peut uniquement voir que les activités.
- Un utilisateur peut voir toutes les activités et uniquement son ou ses enfants avec leurs activités associées.
- L'administrateur peut voir toutes les activités, tous les enfants et les activités associées et avoir une interaction avec tous.
- Seul l'administrateur peut avoir une interaction avec les activités comme les modifier ou les supprimer.

Authentification

Tout ce qui touche à l'authentification donc : la création de compte, la connexion et la déconnexion sont gérés par Jetstream/Livewire qui est installé dans le projet. Les tables sont créées et gérées automatiquement, mais afin d'avoir la possibilité d'utiliser un compte administrateur une colonne booléenne « admin » a été ajouté et seuls les comptes créés par le seeder peuvent être administrateurs.

L'authentification permet d'afficher ou non des éléments sur la page ou bien de donner l'accès à certaines méthodes à l'utilisateur. Il est également possible de choisir un ou plusieurs utilisateurs en particulier selon leur id ou autre caractéristique de la table « *users* ».

Les problèmes

- Lors de la création des models avec leur migrations, Laravel met automatiquement au pluriel le nom de la table, ce qui peut créer des erreurs lorsqu'on appel ces tables.
- De même pour les tables pivots qui doivent avoir un nom dont les mots sont placés dans l'ordre alphabétiques.
- Pour supprimer une activité ou un enfant, il faut s'assurer de supprimer la liaison avec la foreign key grâce à la migration de la table pivot.

Les ajouts potentiels

Les limites d'inscription

Un enfant peut être inscrit à une activité sous conditions :

1. Il faut que cet enfant soit inscrit sur la plateforme. (réalisé)
2. Il faut qu'il soit inscrit à l'activité au moins 2 jours avant le jour indiqué.
3. Il faut qu'une place soit disponible dans l'activité.

Lien entre enfant et activité

Un formulaire présent dans enfant.show affichant toutes les activités aurait permis de pouvoir sélectionner les activités auquel l'enfant aurait participé.

Choix d'un horaire unique

Donner la possibilité de sélectionner un horaire en particulier pour une activité, car à l'inscription, l'enfant va faire tous les horaires.

Modification de l'horaire

Pour l'instant, il est possible d'ajouter un horaire à une activité grâce à l'edit, mais impossible d'en retirer, il faut donc supprimer l'activité pour retirer le lien avec un horaire.

Gestion des duplicatas

Eviter les horaires dupliqués dans la base tout comme les lignes en doubles dans les tables pivot.

Les views

Le site comporte plusieurs views. Elles sont toutes en majorité faites avec Bulma et Bootsrap. Cela facilite grandement la responsive et la mise en page via une mise en forme avec les class des balises.

Le template

La template est constituée d'un header avec une navbar et d'un footer. La template permet que toutes les views héritent du code présent dans la template et donc n'avoir qu'une seule base sur laquelle des blocs vont être imbriqués pour tout l'affichage du site.

La nav barre est sur toutes les views et permet de retourner à la page d'accueil. Pour les visiteurs, ils peuvent se connecter ou créer un compte. Pour les utilisateurs le volet déroulant leur offre la possibilité de se déconnecter et d'ajouter un enfant. Quant à l'administrateur, il a les même options que l'utilisateur, mais il peut en plus ajouter des activités et naviguer sur tout le site.

Les visuels pour un CRUD

Les fichiers list permettent d'afficher tous les enfants inscrits sur le site et toutes les activités en récupérant tous les attributs des tables, le fichier est en lien avec l'index du controller.

Les fichiers edit permettent d'afficher la page permettant de modifier une activité ou un enfant, le fichier est en lien avec le update du controller.

Les fichiers create sont les pages de création afin de créer et d'ajouter une activité ou un enfant, le fichier est en lien avec le create du controller.

Les fichiers show sont les pages d'affichage afin d'afficher en détail une activité ou un enfant, le fichier est en lien avec le show du controller.

Dans le dossier auth le fichier login affiche la page de connexion et le fichier register affiche la page pour créer un compte. Ces pages ont été modifié afin de pas réutiliser les modèles proposés par Laravel/Jetstream.

La base de données

Les tables

La BDD comporte 6 tables que nous avons et 6 tables également crée par Laravel et surtout Jetstream, afin de gérer les comptes utilisateurs. La table « users » a été modifier afin d'insérer une colonne booléenne « admin » qui permet la création de comptes administrateur sur le site. Chacune des tables est associées à un « model ».

La gestion d'ne base de données se fait via des méthodes CRUD (create, read, update, delete) que l'on retrouve dans les contrôleurs associés aux tables. Ces méthodes sont appelées par les views dans le cadre de leur fonction.

Par exemple : pour la view enfant.create qui possède un formulaire, lors de l'envoi de celui-ci c'est la méthode create dans EnfantController qui va être appelé pour faire le lien avec la base de données.

Les tables classiques

- **Enfants**, cette table comporte comme attributs : nom, prénom, date de naissance et user_id car elle à une relation 1 :1 (CIF) avec la table « users » car un enfant ne peut avoir qu'un seul et unique tuteur qui est le user dans ce cas-ci. Elle permet de créer un enfant, elle a également une relation n :n avec *« activites »*.

- **Activites**, cette table comporte comme attributs : nom, description et taille, avec (taille) la nombre de places disponible dans une activité. Cette table a une relation n :n avec *« horaires »* et *« enfants ».*

- **Horaires**, cette table comporte comme attributs : jour et journee, cette table permet de créer un une tranche horaire dans un jour de la semaine, c'est-à-dire choisir parmi [Matin, Après-midi et Matin/Après-midi]. Cette table est en relation n :n avec *« activites ».*

Les tables pivots

Elles sont utiles pour les relations de type n :n dans une base de données, dans ces tables on associe les id des deux éléments à associer.

- **Activite_enfant**, c'est une table pivot, afin de faire la liaison entre *« activités »* et *« enfants »* dans leur relation n :n.
- **Activite_horaire**, c'est une table pivot, afin de faire la liaison entre *« activités »* et *« horaires »* dans leur relation n :n.
- **User_to_enfants**, c'est une table pivot, afin de faire la liaison entre *« users »* et *« enfants »* dans leur relation 1 :1.

Le remplissage de la BDD

Seule 4 des tables sont rempli grâces à un seeder et un factory, les autres sont passe par un seeder déjà existant sans avoir besoin du leur. Les tables : « *activités, enfants, horaires* et *users »* sont remplis en amont afin d'avoir une base fonctionnelle au lancement, mais il va de soi qu'en temps normal la base serait remplie manuellement.

Il est possible de remplir la base en ajoutant une activité seule l'administrateur peut le faire, et par la même occasion il remplit la table *horaire* car une activité doit obligatoirement avoir lieu durant ne tranche horaire, c'est également lui qui prend la décision du nombre de place disponible pour l'activité.

Un autre moyen est de passer par l'ajout d'un enfant sur le profil d'un utilisateur, cette action est possible à partir d'un compte utilisateur ou bien administrateur. Sachant qu'un utilisateur enfant sera obligatoirement lié par la relation 1 :1 avec l'utilisateur connecté, alors qu'un administrateur pourra choisir quel utilisateur attribuer à l'enfant, mais il faudra que le compte utilisateur du tuteur soit préalablement crée.

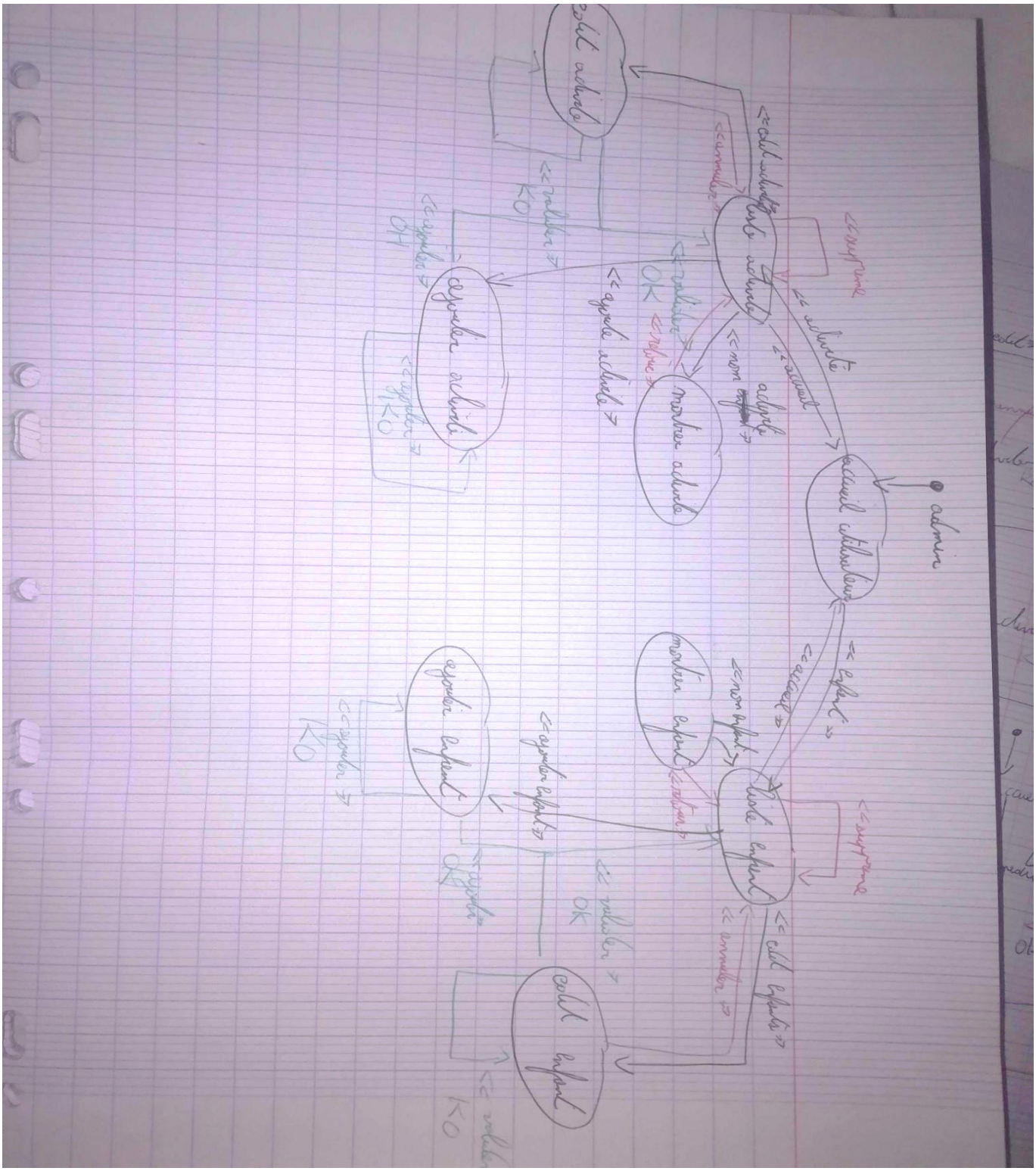
On peut aussi ajouter à la base un compte utilisateur via le formulaire d'inscription, qui permet par la suite de se connecter à la page et de pouvoir inscrire et consulter la page de ses enfants.

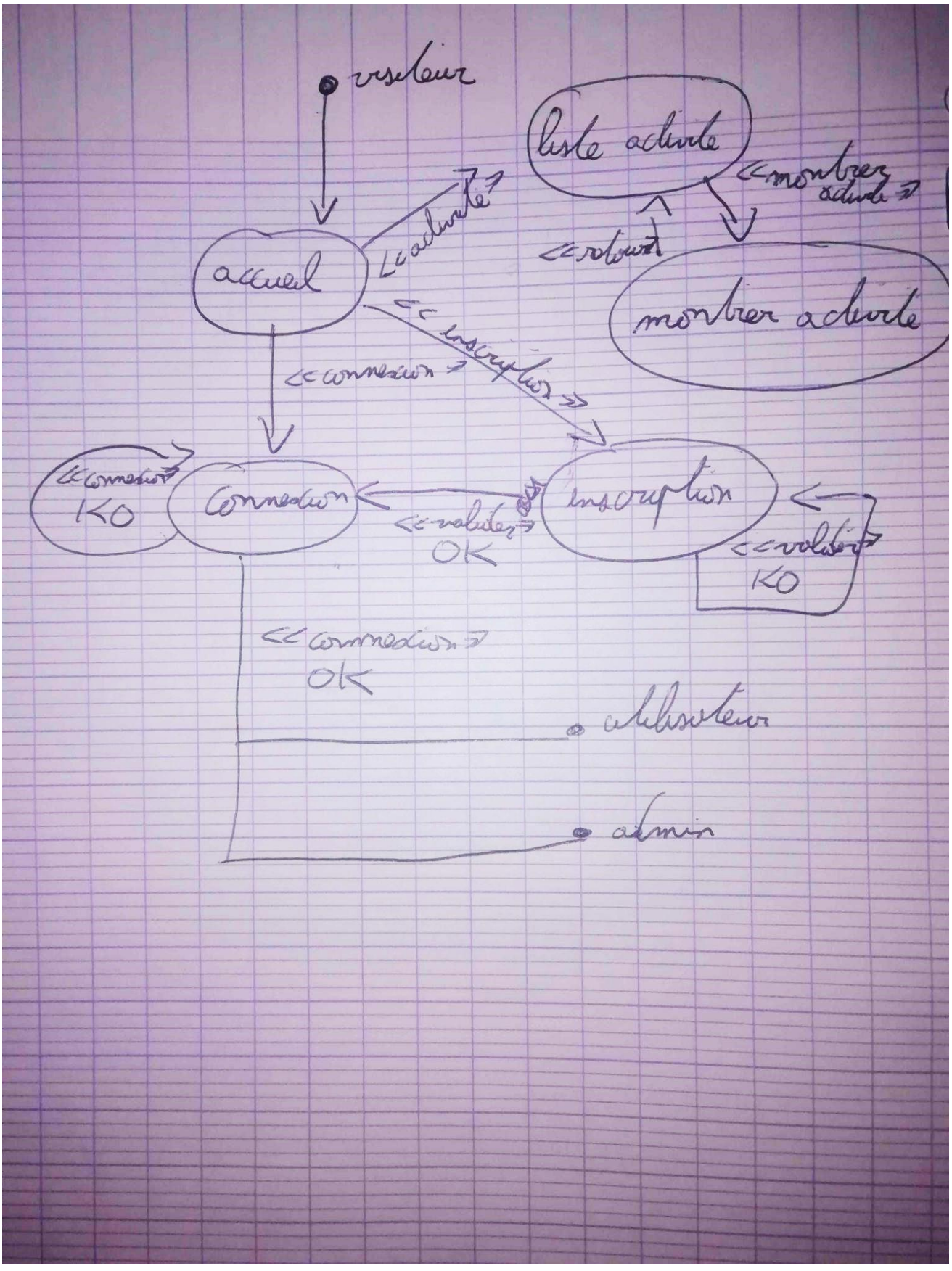
La modification de la BDD

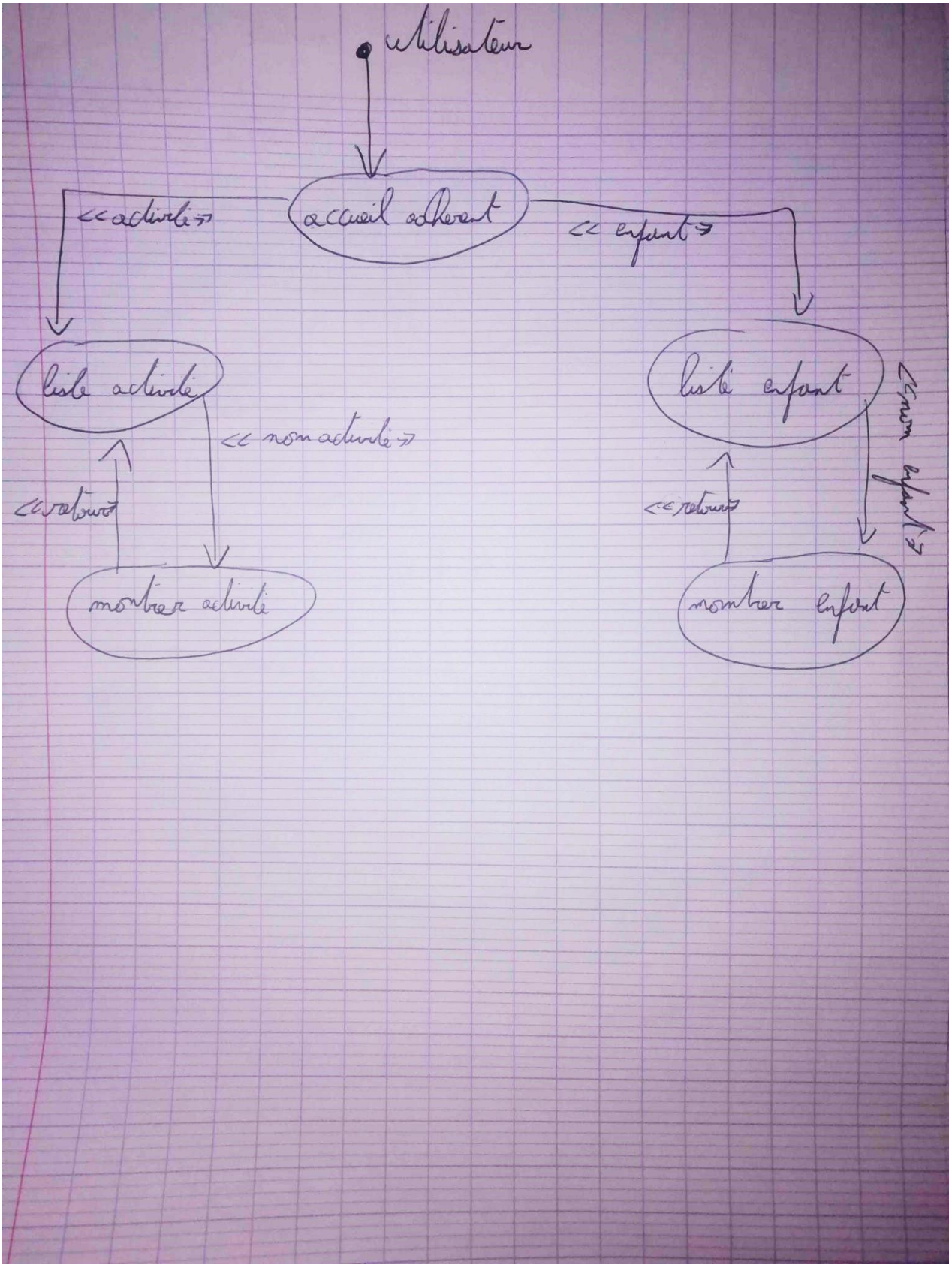
La base de données peut-également être modifié selon certaines conditions. Pour cela nous faisons appel aux méthodes update pour la mise à jour d'une ligne et delete pour la suppression complète.

Une activité peut être modifiée ou une ligne supprimée, en agissant sur la table activité, cela agis également sur la table horaire qui peut être modifié ou bien une ligne supprimée. Pour que la suppression d'une activité se fasse il faut que la relation entre « *activités* » et *« horaires »* soit brisé afin de ne pas avoir de foreign key qui pourrait obstruer la bonne suppression.

Diagrammes de navigation




visiteur
liste activité
<<montrer activité>>
accueil
Lr activité
<<activité>>
montrer activité
<<inscription>>
<<connexion>>
<<connexion KO
connexion
<<valider>>
OK
inscription
<<valider>>
KO
<<connexion>>
OK
utilisateur
admin


utilisateur
accueil adherent
<< activité >>
<< enfant >>
liste activité
<< nom activité >>
<< retour >>
montrer activité
liste enfant
<< nom enfant >>
<< retour >>
montrer enfant

Bibliographie

Faker : https://github.com/fzaninotto/Faker

CSS : https://bulma.io

Laravel :TP7, TP8, TP9

Authentication : Jetstream

Langage : PHP, Blade