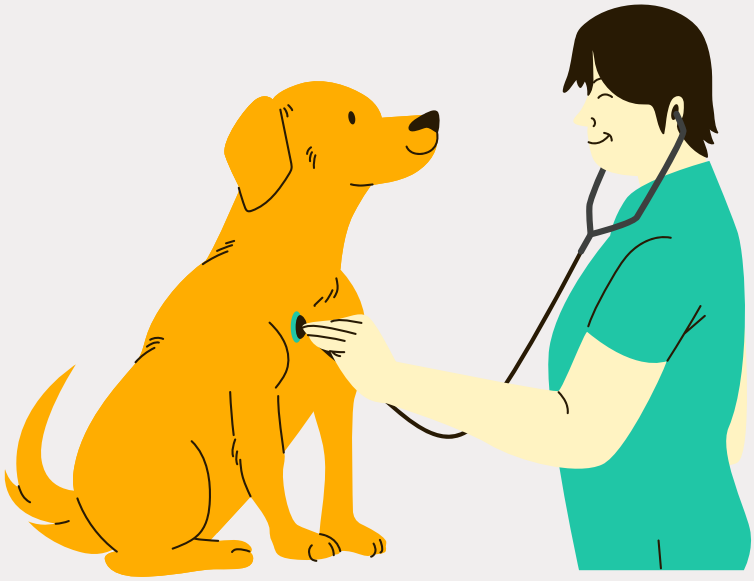
CARE

FUREVER

Sommaire



1. Introduction

FurEverCare est une application Android innovante qui aide les propriétaires d'animaux à mieux prendre soin de leurs animaux. Il permet aux utilisateurs de garder une trace des informations importantes, telles que le nom, la taille et l'historique de santé de leur animal, tous associées à leur compte dans l'application. Il aide également les propriétaires à planifier des visites chez le vétérinaire.

Après une visite au vétérinaire, il peut télécharger toujours dans l'application une fiche médicale qui vous indique tout ce qu'il a fait pendant la visite (une description des procédures) et les médicaments qu'il doit prendre après. Cela permet aux propriétaires d'animaux d'avoir un accès facile aux antécédents médicaux et de rester à jour sur l'état de santé de leur animal.
FurEverCare offre également une fonctionnalité unique qui permet aux propriétaires d'animaux de vérifier le stock de la clinique de leur vétérinaire via l'application. Cela permet de commander facilement les choses importantes dont l'animal a besoin, comme les médicaments. Il aide votre animal à recevoir les bons soins au bon moment.

L'interface conviviale de FurEverCare permet aux propriétaires d'animaux de gérer facilement les informations, les rendez-vous et les besoins en médicaments de leur animal, tous dans un seul endroit. Avec FurEverCare, les propriétaires d'animaux peuvent s'assurer que leurs animaux bien-aimés reçoivent les meilleurs soins possibles, tandis que les vétérinaires peuvent suivre efficacement leurs rendez-vous et leur approvisionnement en médicaments.

2. Fonctionnalités

1. **Login :**
   - FurEverCare fournit un système de connexion sécurisé et facile à utiliser permettant aux propriétaires d'animaux de compagnie de créer un compte et d'accéder à l'application.

2. **Ajout et consultation des données d'animaux :**
   - Une fois connectés, les utilisateurs peuvent facilement ajouter et afficher les données de leurs animaux. Cela comprend la saisie du nom, de la race, de l'espèce, de l'âge, des conditions médicales, des vaccins et des traits de comportement de l'animal. Les utilisateurs peuvent mettre à jour et gérer les informations de leurs animaux de compagnie selon leurs besoins, ce qui facilite la mise à jour des dossiers de leur animal de compagnie.

3. **Planification des rendez-vous :**
   - FurEverCare permet aux propriétaires d'animaux de compagnie de prendre des rendez-vous pour leurs animaux avec des vétérinaires. Les utilisateurs peuvent choisir le vétérinaire, la date et l'heure du rendez-vous. Ils peuvent également sélectionner le service spécifique dont ils ont besoin, comme une consultation au cardioloque. L'application fournit un coût, permettant aux propriétaires d'animaux de planifier en conséquence.

4. **Consultation des détails du rendez-vous :**
   - Une fois un rendez-vous planifié, les propriétaires d'animaux de compagnie peuvent consulter les détails du rendez-vous, y compris l'emplacement, le vétérinaire, la date, l'heure, le service et le coût. Cela permet aux utilisateurs d'avoir une

vue d'ensemble claire des rendez-vous à venir de leur animal et de planifier leur emploi du temps en conséquence.

5. Consultation des médicaments et des procédures :

- Après le rendez-vous, le vétérinaire peut télécharger un reçu détaillé avec des informations sur les médicaments et les procédures effectués lors de la visite. Les propriétaires d'animaux peuvent consulter ces informations dans l'application, ce qui les aide à rester informés sur l'historique médical et le traitement de leur animal.

6. Vérification du stock et commande de médicaments :

- FurEverCare offre une fonctionnalité unique qui permet aux propriétaires d'animaux de vérifier le stock de la clinique de leur vétérinaire via l'application. Cela permet aux propriétaires d'animaux de commander facilement les médicaments et les fournitures nécessaires pour leurs animaux, en s'assurant qu'ils disposent d'un approvisionnement suffisant. Les propriétaires d'animaux peuvent passer des commandes via l'application et suivre leur statut en temps réel.

3. Diagrammes

L'application ***FurEverCare*** est conçue à l'aide d'une approche orientée objet et est implémentée à l'aide de plusieurs classes qui interagissent les unes avec les autres pour obtenir les fonctionnalités de l'application (*Figure 1*).

Le diagramme de classes se compose de plusieurs classes, notamment une classe Utilisateur, une classe Animal, une classe Rendez-vous, une classe Service, une classe Type de service, une classe Clinique, une classe Traitement, une classe Médecine et une classe Stock.

La classe ***User*** représente les utilisateurs de l'application et contient des informations telles que leur nom, leurs coordonnées et leurs identifiants de connexion. La classe Pet représente les animaux de compagnie de l'utilisateur et contient des informations telles que leur nom, leur race, leur espèce, leur âge, leur état de santé, leurs vaccins et leur comportement.

La classe ***Appointment*** représente les rendez-vous pris par les utilisateurs pour leurs animaux de compagnie et contient des informations telles que la date, l'heure et le lieu du rendez-vous.

La classe ***Rendez-vous*** est associée à la classe Animal de compagnie, permettant aux utilisateurs de planifier des rendez-vous pour leurs animaux de compagnie.

La classe ***Service*** représente les services offerts par la clinique et contient des informations telles que le type de service et le coût. La classe Service est associée à la classe Type de service, qui représente les différents types de services offerts par la clinique.

La classe ***Clinique*** représente la clinique où les rendez-vous ont lieu et contient des informations telles que le nom, l'adresse et les coordonnées. La classe Clinique est associée à la classe Rendez-vous, permettant aux utilisateurs de planifier des rendez-vous à la clinique.

La classe ***Traitement*** représente les traitements donnés aux animaux lors de leurs rendez-vous et contient des informations telles que le nom, la description et le coût. La classe Traitement est associée à la classe Rendez-vous, permettant aux vétérinaires d'ajouter des traitements au dossier de rendez-vous.

La classe ***Médicament*** représente le médicament prescrit aux animaux de compagnie lors de leurs rendez-vous et contient des informations telles que le nom, la posologie et le coût. La classe Médecine est associée à la classe Traitement, permettant aux vétérinaires de prescrire des médicaments dans le cadre du plan de traitement de l'animal.

La classe ***Stock*** représente le stock de médicaments de la clinique et contient des informations telles que le nom et la quantité du médicament. La classe Stock est associée à la classe Médicament, permettant aux utilisateurs de vérifier la disponibilité des médicaments avant de les commander.

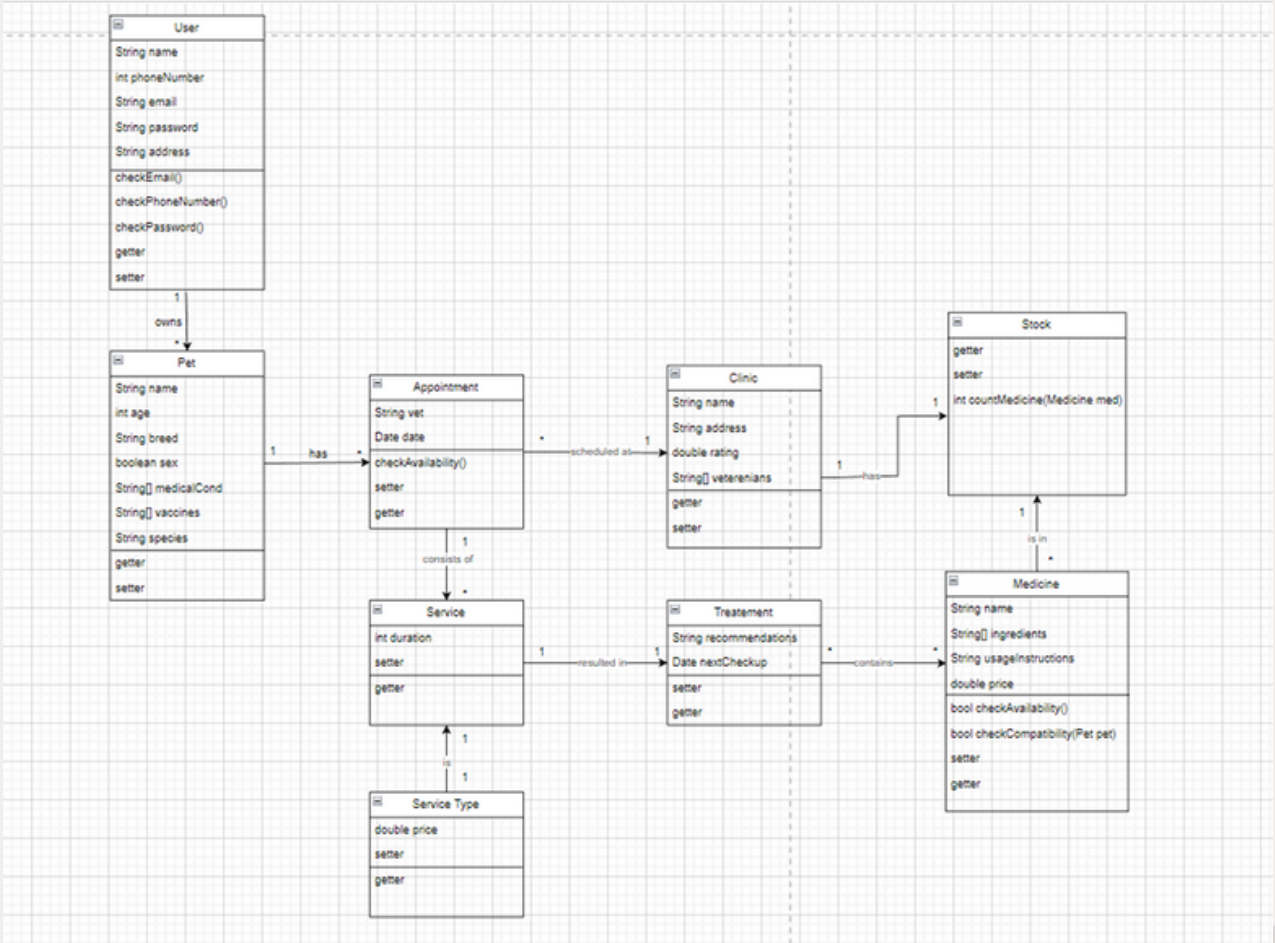


Figure 1. Diagramme du classe

Le diagramme d'activité de l'application FurEverCare illustre le flux d'activités que les utilisateurs peuvent effectuer dans l'application (*Figure 2*).

Le diagramme commence par l'activité Connexion/Inscription, qui permet aux utilisateurs de se connecter avec leur compte existant ou de créer un nouveau compte. Les activités Vérifier l'e-mail et Vérifier le mot de passe vérifient les détails du compte de l'utilisateur et authentifient ses identifiants de connexion.

Une fois connectés, les utilisateurs sont redirigés vers l'activité Menu principal, qui leur offre plusieurs options telles que la création/mise à jour des détails de leur animal, la mise à jour de leur propre profil d'utilisateur, la prise de rendez-vous et la vérification des détails de leur compte.

Les activités *Créer/Mettre à jour* l'animal et *Modifier les détails* de l'animal permettent aux utilisateurs d'ajouter et de modifier respectivement les informations de leur animal. L'activité *Afficher les détails* de l'animal permet aux utilisateurs de consulter les informations sur leur animal.

Les activités *Mettre à jour* l'utilisateur et *Modifier les détails* de l'utilisateur permettent aux utilisateurs de modifier respectivement leurs propres informations de profil d'utilisateur et les détails de leur compte. L'activité *Afficher* les détails de l'utilisateur permet aux utilisateurs de consulter leurs propres informations de profil utilisateur.

L'activité *Prendre un rendez-vous* permet aux utilisateurs de planifier un rendez-vous dans une clinique. L'activité Vérifier la disponibilité permet aux utilisateurs de vérifier la disponibilité de la clinique avant de planifier le rendez-vous. L'activité Créer un rendez-vous permet aux utilisateurs de programmer le rendez-vous une fois qu'ils ont choisi une date et une heure appropriées. L'activité *Recevoir* une confirmation confirme le rendez-vous de l'utilisateur et lui envoie une notification.

L'activité *Donner un traitement* permet à la clinique d'administrer un traitement à l'animal de l'utilisateur lors de son rendez-vous. L'activité *Afficher le traitement* permet aux utilisateurs de revoir les traitements que leur animal a reçus lors

du rendez-vous.

L'activité *Acheter des médicaments* permet aux utilisateurs d'acheter des médicaments pour leur animal s'ils sont prescrits lors de leur rendez-vous. L'activité *Vérifier le stock* de médicaments permet aux clinique de vérifier s'il dispose des médicaments nécessaires en stock. L'activité *Recevoir la confirmation de paiement* confirme le paiement effectué par l'utilisateur.

Enfin, l'activité de *Déconnexion* permet aux utilisateurs de se déconnecter de l'application.

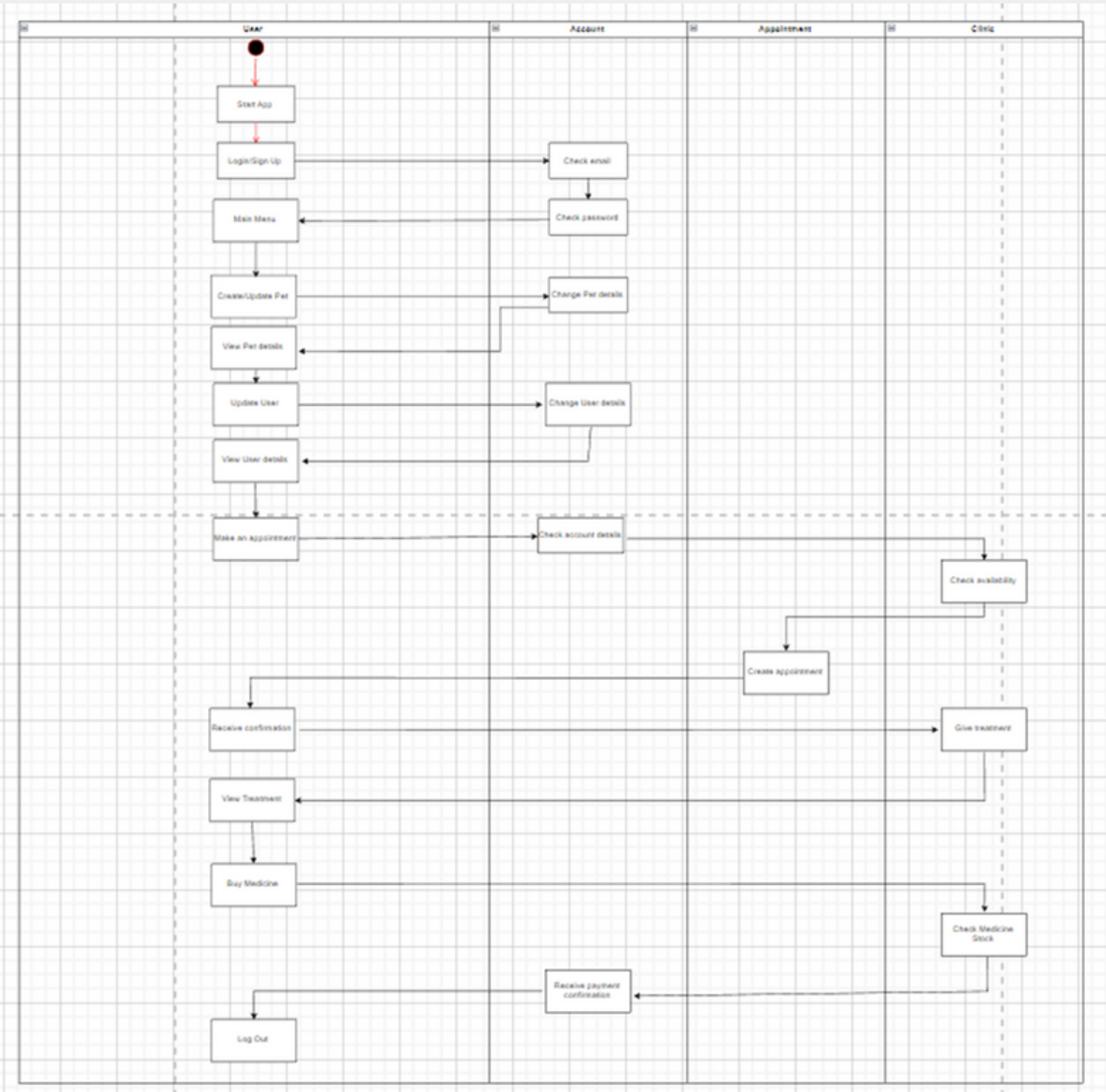


Figure 2. Diagramme d'activité

4. Interface et Implementation

4.1 PAGE DE REGISTER

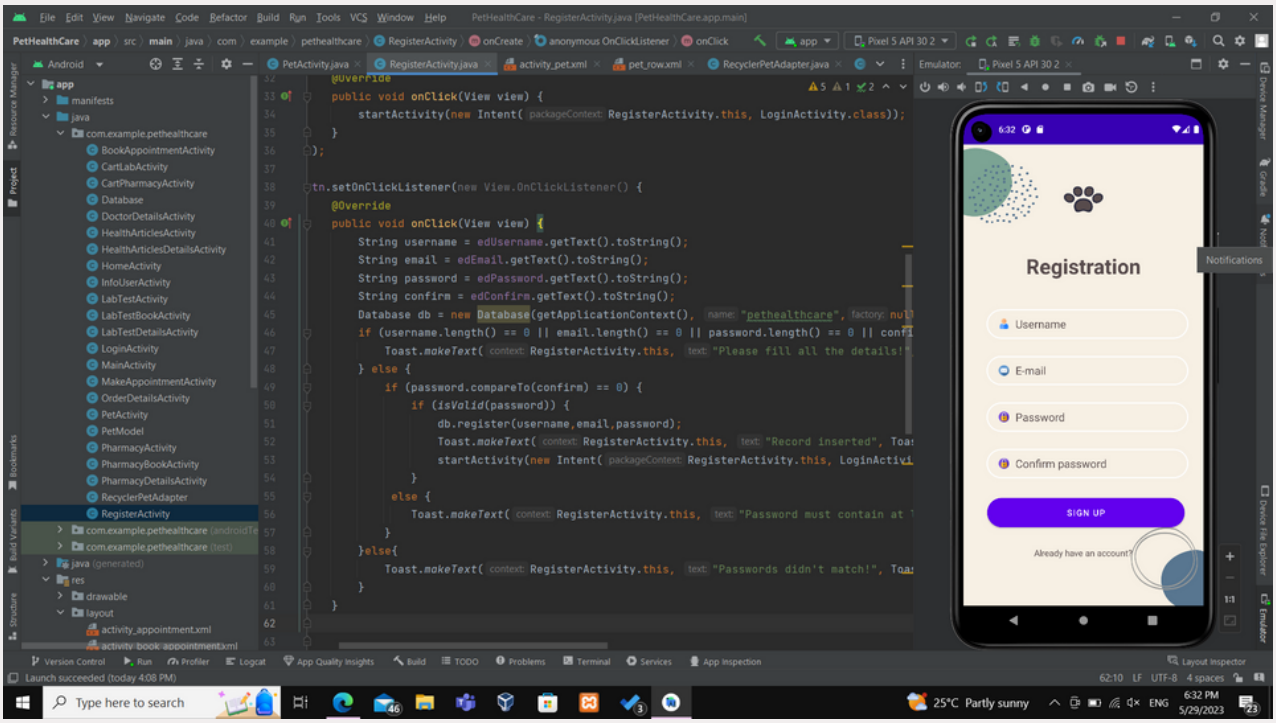


Figure 3. RegisterActivity.java

Ce code s'agit d'une classe d'activité Android (*Figure 3*) chargée de gérer la fonctionnalité d'inscription des utilisateurs.

En ce qui concerne l'interface graphique, cette activité doit comporter les éléments suivants :

- **EditText** :
  - *edUsername* - champ pour saisir le nom d'utilisateur (nom complet) ;
  - *edEmail* - champ pour saisir l'adresse e-mail ;
  - *edPassword* - champ pour saisir le mot de passe ;
  - *edConfirm* - champ pour confirmer le mot de passe.

- **Button** :
  - *btn* - bouton pour l'inscription.
- **TextView** :
  - *tv* - texte indiquant qu'un utilisateur existe déjà.

Le code commence par initialiser et assigner les éléments de l'interface utilisateur (EditText, Button, TextView) aux variables correspondantes.

Après l'initialisation des éléments de l'interface utilisateur, deux écouteurs d'événements sont ajoutés pour deux actions différentes :

1. **tv.setOnClickListener()** - Un écouteur de click pour le texte **tv** (un utilisateur existe déjà). Lorsque ce texte est cliqué, l'activité **LoginActivity** est ouverte;
2. **btn.setOnClickListener()** - Un écouteur de click pour le bouton **btn** (inscription). Cet écouteur gère l'action d'inscription.

Ensuite, il est vérifié si tous les champs ont été remplis, sinon un message Toast est affiché pour demander de remplir tous les détails.

Si tous les champs sont remplis, il est vérifié si le mot de passe et la confirmation du mot de passe correspondent (**password.compareTo(confirm) == 0**). Si tel est le cas, il est vérifié si le mot de passe répond aux critères définis dans la méthode **isValid()**. Si cette vérification est réussie, l'utilisateur est enregistré dans la base de données (**db.register(username,email,password)**), un message Toast de confirmation est affiché et l'activité **LoginActivity** est ouverte.

Si les mots de passe ne correspondent pas, un message Toast est affiché pour indiquer que les mots de passe ne sont pas identiques.

La méthode **isValid(String passwordhere)** est une méthode auxiliaire définie pour vérifier si un mot de passe est valide.

4.2 PAGE DE LOGIN

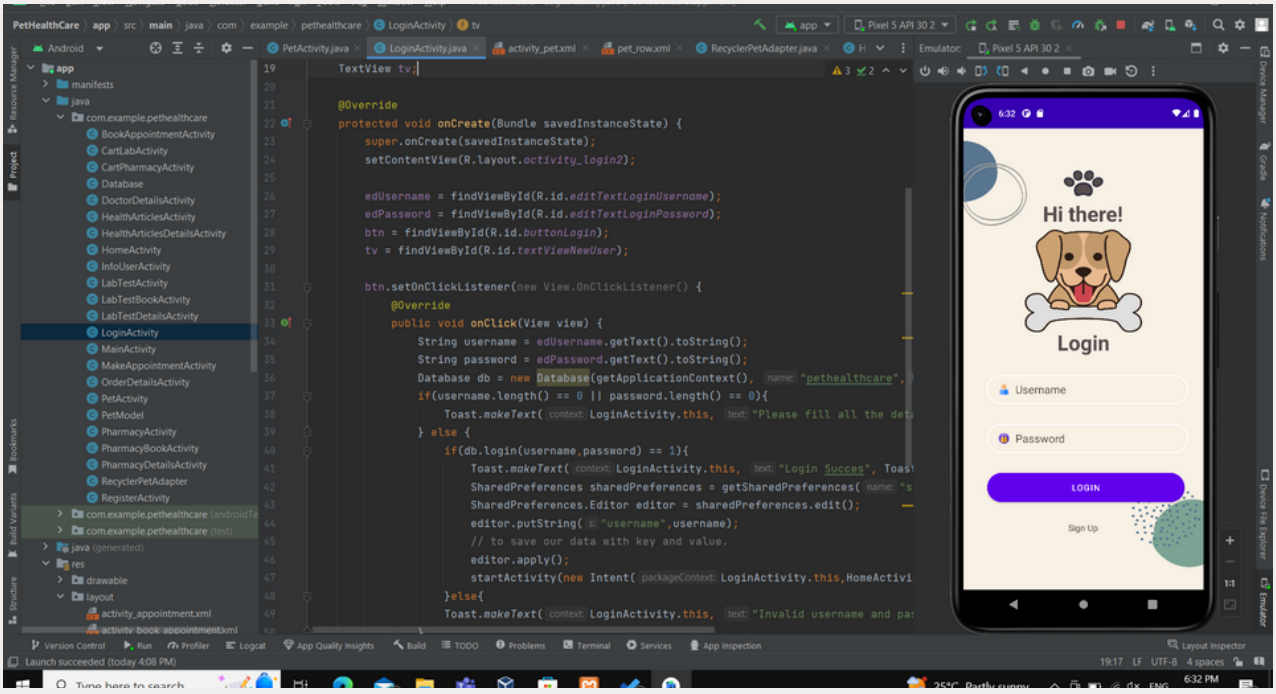


Figure 4. LoginActivity.java

Le code s'agit d'une classe d'activité Android (*Figure 4*) chargée de gérer la fonctionnalité de connexion des utilisateurs.

En ce qui concerne l'interface graphique, cette activité doit comporter les éléments suivants :

- **EditText** :
  - *edUsername* - champ pour saisir le nom d'utilisateur ;
  - *edPassword* - champ pour saisir le mot de passe.
- **Button** :
  - *btn* - bouton pour la connexion.
- **TextView** :
  - *tv* - texte indiquant qu'un nouvel utilisateur peut s'inscrire.

Le code commence par initialiser et assigner les éléments de l'interface utilisateur (EditText, Button, TextView) aux variables correspondantes, en utilisant la méthode **findViewById()** pour obtenir les références à ces éléments en fonction de leurs identifiants dans le fichier XML de l'activité (**R.layout.activity_login2**).

Ensuite, deux écouteurs d'événements sont ajoutés pour deux actions différentes :

1. **btn.setOnClickListener()** - Un écouteur de click pour le bouton btn (connexion). Cet écouteur gère l'action de connexion. Ensuite, il est vérifié si tous les champs ont été remplis, sinon un message Toast est affiché pour demander de remplir tous les détails. Si tous les champs sont remplis, une vérification de la connexion est effectuée en appelant la méthode db.login(username,password) qui renvoie une valeur de retour (1 pour une connexion réussie). Si la connexion est réussie, un message Toast de succès est affiché, les informations de l'utilisateur sont sauvegardées à l'aide de SharedPreferences, et l'activité **HomeActivity** est ouverte. Si la connexion échoue, un message Toast est affiché pour indiquer que le nom d'utilisateur et le mot de passe sont invalides.
2. **tv.setOnClickListener()** - Un écouteur de click pour le texte **tv** (nouvel utilisateur). Lorsque ce texte est cliqué, l'activité **RegisterActivity** est ouverte.

4.3 PAGE D'ACCUEIL

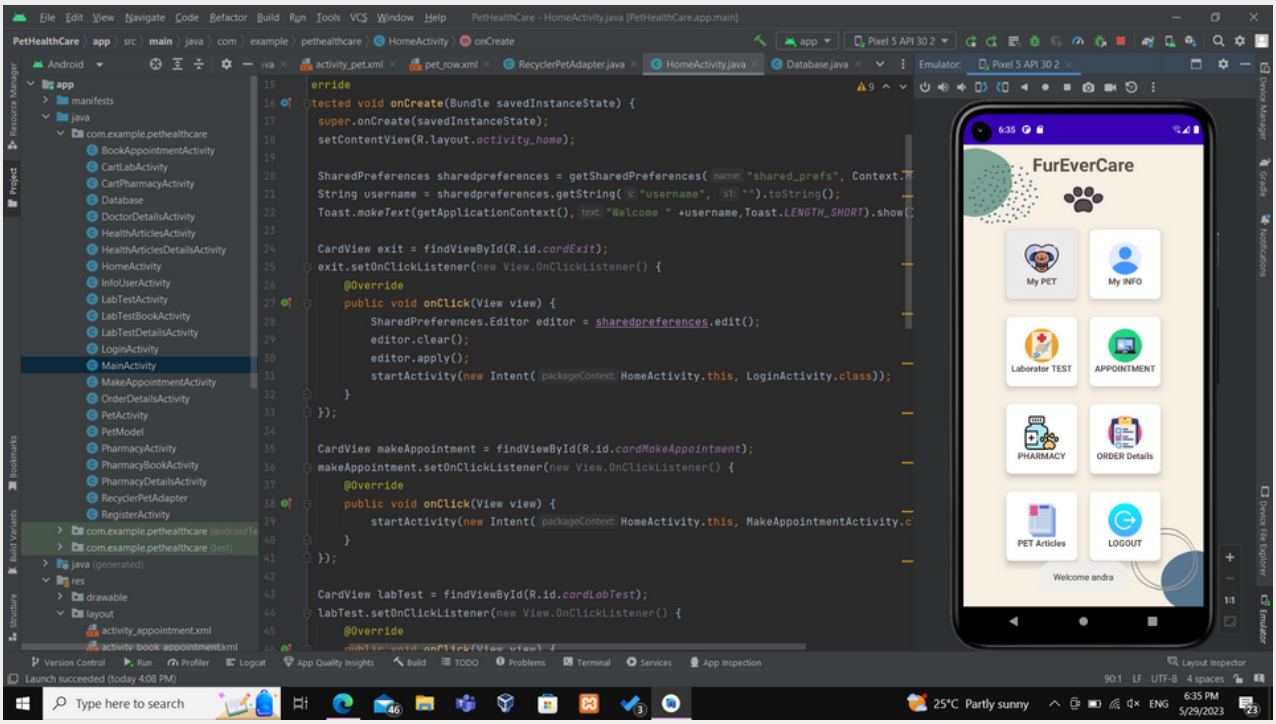


Figure 5. HomeActivity.java

Ce code s'agit d'une classe d'activité Android (*Figure 5*) chargée de gérer l'écran d'accueil de l'application.

Le code commence par récupérer le nom d'utilisateur à partir des **SharedPreferences** et affiche un message Toast de bienvenue.

Ensuite, des écouteurs de click sont ajoutés à plusieurs éléments de type **CardView**. Chaque écouteur gère le click sur un élément spécifique et ouvre une nouvelle activité correspondante en créant et en démarrant une nouvelle instance de la classe **Intent**.

Les éléments CardView et leurs correspondances avec les activités sont les suivants :

- **exit** : LoginActivity.class ;
- **makeAppointment** : MakeAppointmentActivity.class ;
- **labTest** : LabTestActivity.class ;
- **orderDetails** : OrderDetailsActivity.class ;
- **pharmacy** : PharmacyActivity.class ;
- **health** : HealthArticlesActivity.class ;
- **infoUser** : InfoUserActivity.class ;
- **myPet** : PetActivity.class .

Ainsi, lorsque l'utilisateur clique sur l'un des éléments de la liste, il est redirigé vers l'activité correspondante.

4.4 PAGE MYINFO

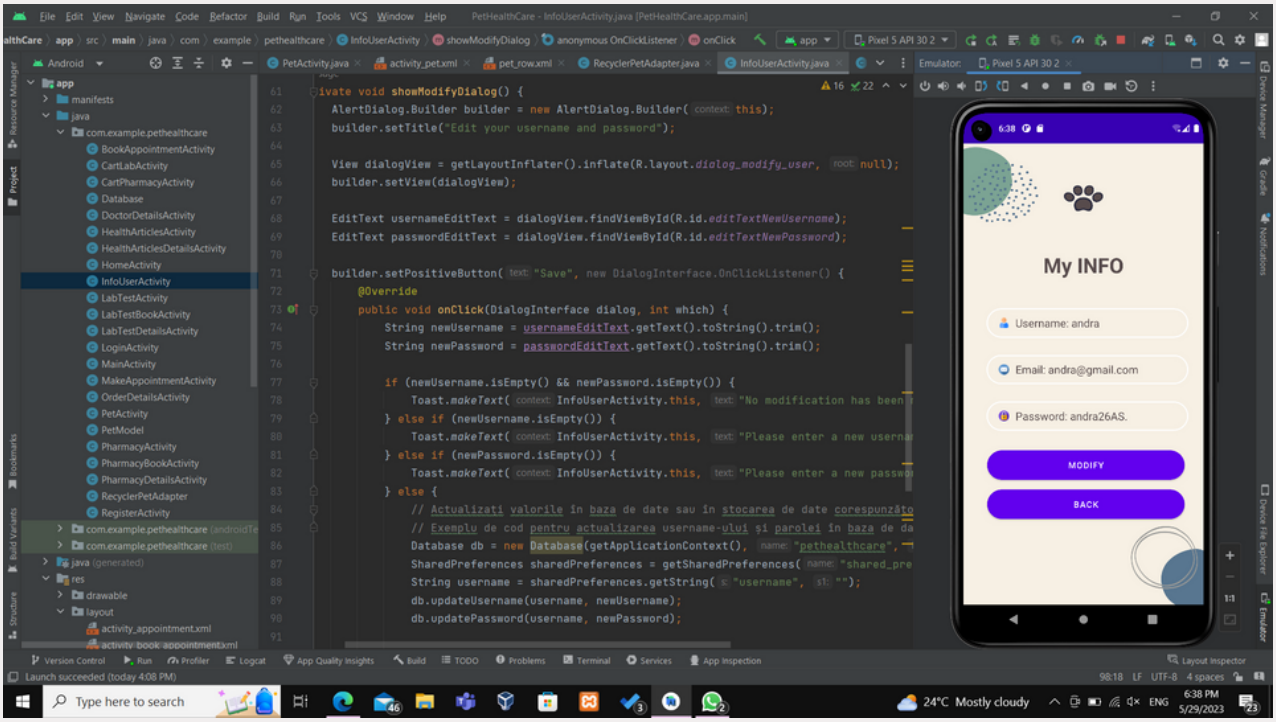


Figure 6. InfoUserActivity.java

La classe **InfoUserActivity** (*Figure 6*) est responsable de l'affichage des informations de l'utilisateur connecté et de la possibilité de les modifier. Elle comporte les éléments d'interface utilisateur suivants :

- **EditText** :
  - *ed1* - champ pour afficher et modifier le nom d'utilisateur
  - *ed2* - champ pour afficher l'email de l'utilisateur
  - *ed3* - champ pour afficher et modifier le mot de passe de l'utilisateur
- **TextView** :
  - *tv* - titre de l'application
- **Button** :
  - *btnBack* - bouton pour revenir à l'activité d'accueil (HomeActivity)

- btnModify - bouton pour ouvrir une boîte de dialogue permettant de modifier le nom d'utilisateur et le mot de passe

Dans la méthode **onCreate()**, les éléments de l'interface utilisateur sont initialisés et des écouteurs sont ajoutés pour les boutons btnBack et btnModify.

La méthode **showModifyDialog()** est responsable de l'affichage de la boîte de dialogue de modification, qui permet à l'utilisateur de mettre à jour son nom d'utilisateur et son mot de passe.

4.5 PAGE MYPET

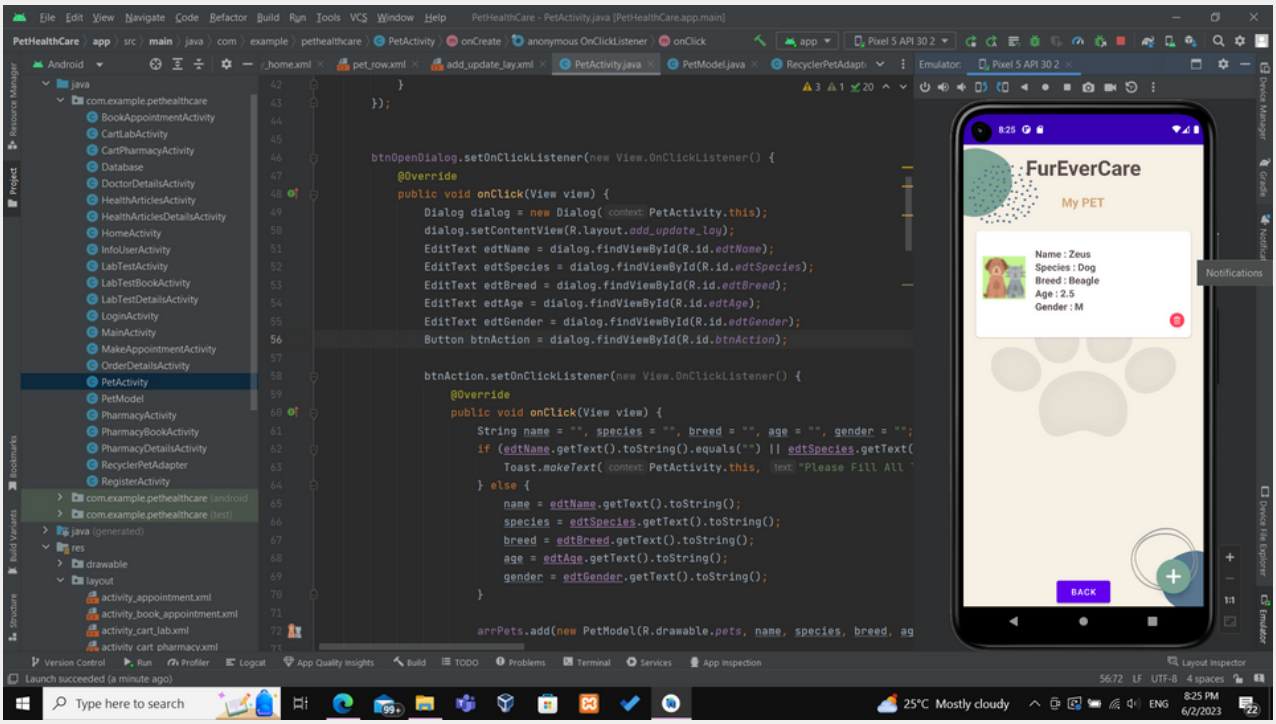


Figure 7. PetActivity.java

La classe **PetActivity** (*Figure 7*) gère l'affichage et l'ajout des animaux de compagnie dans l'application. Elle utilise une liste arrPets pour stocker les objets représentant les animaux de compagnie.

Dans la méthode **onCreate()**, les éléments de l'interface utilisateur sont initialisés et des auditeurs sont ajoutés aux boutons btnBack et btnOpenDialog.

La méthode **showModifyDialog()** affiche une boîte de dialogue pour ajouter un nouvel animal de compagnie. Le nom, l'espèce, la race, l'âge et le genre de l'animal sont saisis dans les champs de saisie (EditText), et lorsque le bouton btnAction est pressé, ces informations sont ajoutées à la liste arrPets.

Le RecyclerView est configuré pour afficher la liste des animaux de compagnie en utilisant un adaptateur (**RecyclerPetAdapter**).

4.6 PAGE MAKEAPPOINTMENT

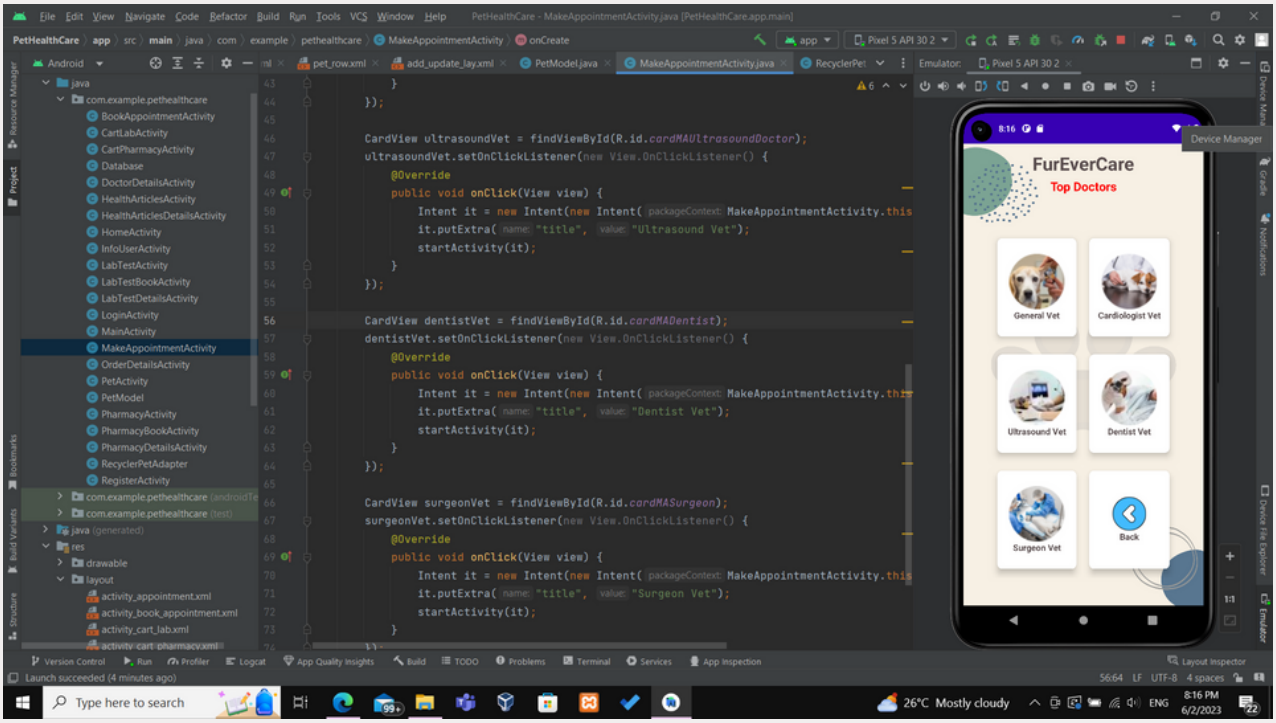


Figure 8. MakeAppointmentActivity.java

La classe **MakeAppointmentActivity** (*Figure 8*) gère la création des rendez-vous dans l'application. Elle utilise des éléments de type CardView pour afficher différentes options de rendez-vous.

Dans la méthode **onCreate()**, les CardViews sont initialisés et des auditeurs sont ajoutés à chaque CardView pour détecter les clics. Lorsqu'un CardView est cliqué, une intention (Intent) est créée pour ouvrir l'activité **DoctorDetailsActivity** avec des informations spécifiques sur le type de rendez-vous sélectionné.

Par exemple, si l'utilisateur clique sur le CardView pour un vétérinaire généraliste, l'intention contiendra le titre "General Vet". L'activité DoctorDetailsActivity sera ensuite lancée avec cette intention.

Ce processus est répété pour chaque type de rendez-vous disponible, tels que le cardiologue, l'échographiste, le dentiste et le chirurgien vétérinaires.

5. Conclusions

En conclusion, **FurEverCare** est une application Android utile et efficace pour la santé des animaux de compagnie. Elle offre aux propriétaires un moyen centralisé de gérer les informations concernant leurs animaux, de prendre rendez-vous chez les vétérinaires, d'accéder à des détails médicaux importants et de commander des médicaments et du matériel nécessaires. Cela facilite les soins appropriés et à jour pour les animaux de compagnie.