Relatório Técnico - Android Module

VetCare

Unidade Curricular:	Computação Móvel
Data:	Aveiro, 12 de Abril de 2018
Autores:	76538: Raquel Oliveira Ramos 72099: Ana Rita Antunes Santiago
Resumo:	O VetCare é uma aplicação móvel que permite uma melhor realização dos planos de tratamentos aos animais internados em clínicas veterinárias. Estes planos de tratamentos consistem essencialmente em lembretes de medicações a administrar e tratamentos a realizar apesar de a aplicação ter outras funcionalidades também importantes e úteis para todas as clínicas veterinárias.

Índice

1 Introdução

O VetCare, aplicação móvel desenvolvida pelo grupo, vem de encontro à necessidade de auxiliar nos cuidados a animais internados.

Nos dias de hoje, o que se vê essencialmente em clínicas veterinárias passa por informação em papel ou dados presentes nos computadores em softwares desenvolvidos para estas clínicas. Mas e se fosse possível haver maneira de lembrar os médicos e enfermeiros veterinários de que têm realizar determinada tarefa ou administrar determinada medicação em determinado animal sem recorrer a *post-it*? E se fosse possível identificar animais rapidamente? E se fosse possível ainda obter toda a informação do animal através de um clique rápido sem ter procurar entre enormes quantidades de papelada ou sem ter que se deslocar para ir ao computador? É aqui que surge o VetCare.

O objetivo do VetCare é essencialmente agir como lembrete e fonte de informação dos tratamentos a serem realizados e da medicação que terá que ser administrada aos animais. A aplicação desenvolvida conta também com a implementação de outras funcionalidades bastantes úteis para os médicos e enfermeiros veterinários que irão ser referidas seguidamente.

2 Conceito da Aplicação

A aplicação móvel desenvolvida encontra-se orientada para todos os trabalhadores de clínicas veterinárias sejam estes médicos ou enfermeiros veterinários.

O principal cenário de uso da aplicação será para relembrar um médico ou enfermeiro veterinário acerca de tratamentos a efetuar a determinado animal ou até mesmo que medicação administrar, quando e qual a dose da mesma. Após efetuar a respetiva tarefa, o funcionário que a efetuou, consegue dar essa mesma tarefa por concluída apenas com um toque.

Um outro cenário de uso bastante crucial e pedido como requisito da aplicação móvel antes do desenvolvimento da mesma, foi a rápida identificação do animal, seja este qual for e em qualquer momento ou lugar.

Para ajudar ao principal objetivo da aplicação, ou seja, para ajudar a tratar e nomeadamente monitorizar os animais internados, é ainda possível observar os biosinais de cada animal em forma de gráfico (mais especificamente, batimentos cardíacos e temperatura) ao longo de cada dia.

3 Design da Interface Visual

Fragmentos

O trabalho é constituído apenas por duas atividades: a actividade de Login e a Main. A Main Activity é a *Activity* principal onde se encontram inseridos todos os fragmentos desenvolvidos para a aplicação dizendo respeito às diferentes funcionalidades. A decisão do uso de vários fragmentos em vez de *Activities* baseou-se numa pequena pesquisa em que foi possível concluir que é um bom padrão de programação em Android o uso de vários fragmentos interligados entre si, bem como facto de a inicialização de uma *Activity* consumir mais tempo e memória que a inicialização de um Fragmento. Juntando estes factos, ao que

foi leccionado nas aulas e sabendo que os Fragmentos surgiram em Android como uma maneira de criar interfaces reutilizáveis, foi então optado o uso de vários Fragmentos em vez de várias *Activities*.

Esta decisão facilitou-nos o uso de um *Navigation Drawer* (que será explicado de seguida), visto que este é comum a todos os fragmentos e estando inserido na Main Activity.

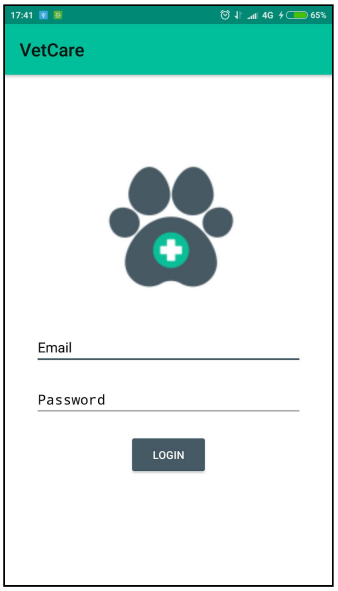

Fig. 1: Ecrã de Login

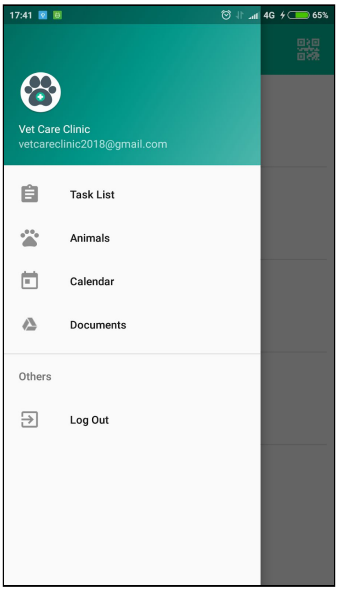

Fig. 2: Navigation Drawer

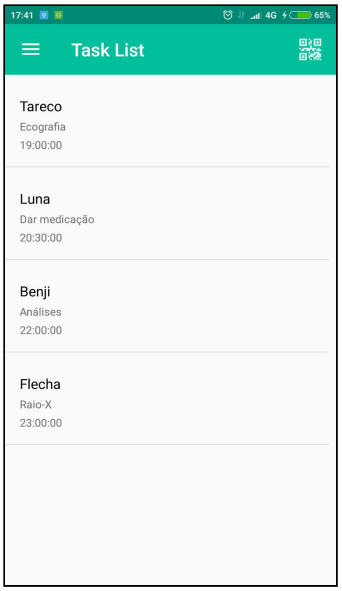

Fig. 3: Task List

Layouts

Todas as UI foram construídas com o uso de *ConstraintLayout*. Apesar de a *ConstraintLayout* ser semelhante à *RelativeLayout* no sentido em que todos os componentes da *View* são dispostos de acordo com relações com os outros componentes e de acordo com o parent, o uso de *ConstraintLayout* é mais flexível que a *RelativeLayout* e é de mais fácil uso quando usado com o editor das *layout* do Android Studio.

Em muitas das UI desenvolvidas para a aplicação, é possível observar-se o uso de *RecyclerView*. O uso de *RecyclerView* foi preferido em relação ao uso de *ListView* porque apesar de estes dois componentes serem muito parecidos, uma *RecyclerView* é como que uma melhoria da *ListView*. Na *RecyclerView*, esta reusa as células ao haver scroll da lista tanto para cima como para baixo (uso mandatório do *ViewHolder* enquanto que na *ListView* o uso do mesmo era apenas opcional) e separa a lista do seu *container*, ou seja, é mais fácil colocar itens da lista em runtime em diferentes containers usando o *LayoutManager*. Com *RecyclerView* também se podem implementar animações usando o *ItemAnimator* mas tal não foi implementado.

Navigation Drawer

O *Navigation Drawer* é um elemento importante da UI principal pois permite navegar para qualquer um dos fragmentos onde estão implementadas as principais funcionalidades desenvolvidas. Este componente mostra informação útil ao utilizador tais como o nome e mail da clínica pela qual o utilizador fez login e os botões que pode utilizar para navegar pela aplicação.

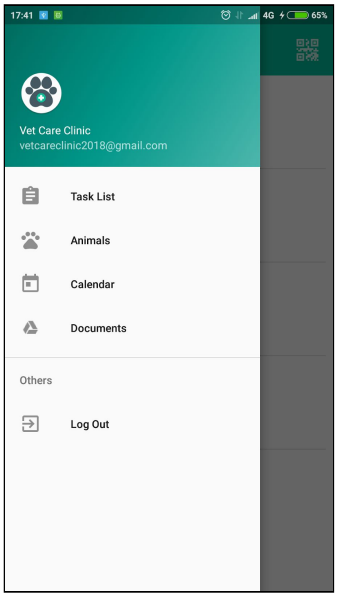

Fig. 4: Navigation Drawer

Cores

As cores usadas na aplicação tiveram em conta as típicas cores dos espaços de saúde bem como dos uniformes dos funcionários que trabalham nestes locais. Nomeadamente a *primary color*, as cores escolhidas são muito frequentes em estabelecimentos de saúde (quer humana quer animal) daí a escolha da cor e o enquadramento da mesma na aplicação.

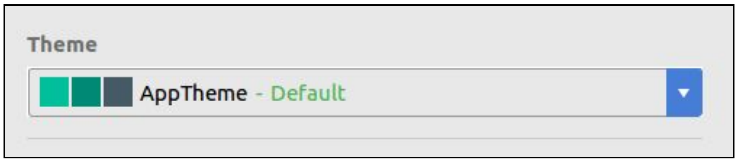

Fig. 5: Esquema de Cores

Ícones

Todos os ícones escolhidos para a UI têm como principal objetivo serem intuitivos para o utilizador ter uma boa experiência com a aplicação.

Para além deste facto, todos os ícones se encontram presentes nos diretórios *drawable* dentro do diretório *res,* havendo ícones com diferentes tamanhos para as diferentes densidades (mdpi, hdpi, xhdpi, entre outros).

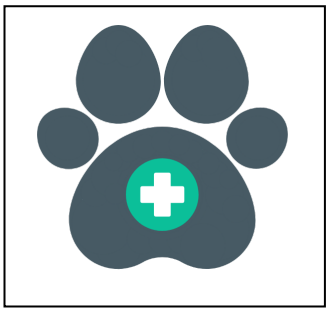

Fig. 6: Ícone da Aplicação

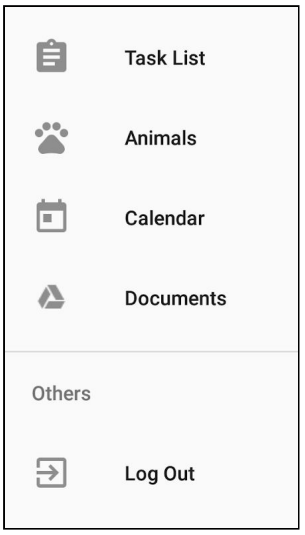

Fig. 7: Ícones do Navigation Drawer

Botões

Para os botões foram usados botões simples e *Floating Action Buttons*. O uso dos *Floating Action Buttons* partiu da leitura da documentação do Android de como se devia proceder à criação das UI juntamente com o Material Design e nomeadamente as *guidelines* da disposição dos botões na UI. Todos os botões são intuitivos e se encontram com a cor indicada ao tema (neste caso, a *accent color*).

Fig. 8: Botões Simples

Fig. 9: Floating Action Buttons

4 Arquitetura da Solução

Diagrama

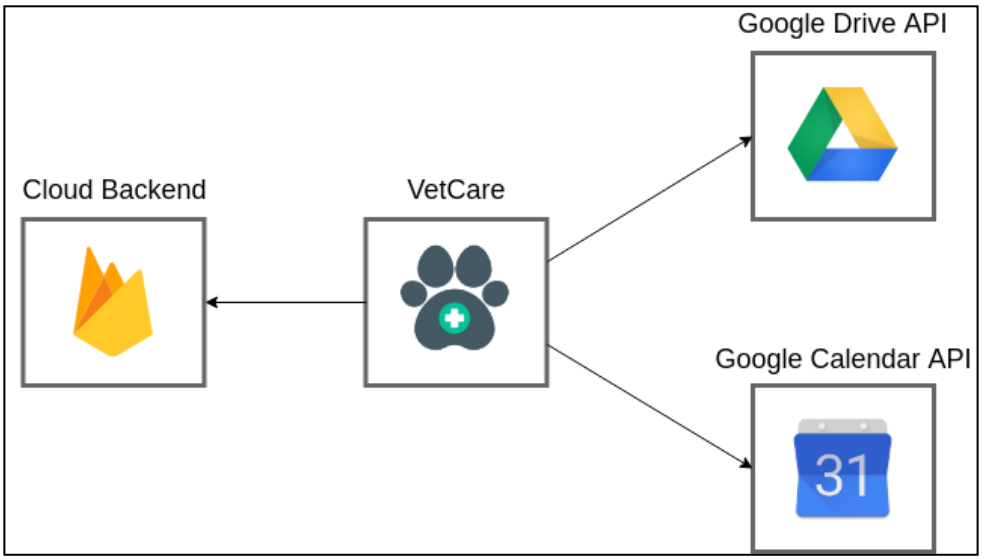

Fig. 10: Arquitetura

A imagem apresentada acima representa a arquitetura global do projeto desenvolvido pelo grupo. É possível observar-se a aplicação móvel VetCare no centro que tem como back-end a plataforma Firebase (lado esquerdo do diagrama) onde se encontram presentes vários dados da aplicação; é possível ainda verificar-se que a aplicação móvel comunica com duas API: Google Drive API e Google Calendar API (lado direito do diagrama). Posteriormente irá ser explicado o porquê de se ter implementado o acesso a estas duas API.

Modelos de Dados

Na aplicação móvel desenvolvida, estão envolvidos 6 tipos de dados diferentes. Estes tipos de dados são:

- Animal: informações do animal quando este dá entrada na clínica veterinária como por exemplo o seu nome, uma imagem identificadora do mesmo e o peso quando deu entrada;
- Owner: informações acerca do dono nomeadamente número de telemóvel caso seja necessário contactar o mesmo;
- Internment: informações básicas do internamento, como por exemplo o motivo pelo qual foi à clínica e que médico veterinário viu o animal na sua entrada;
- Procedure: procedimentos realizados ao animal enquanto este se encontra internado;
- Medicine: medicamentos administrados ao animal enquanto este se encontra internado;
- Historic: procedimentos de rotina realizados em consultas prévias ao seu internamento na clínica;

Os tipos de dados encontram-se ainda representados na imagem seguinte:

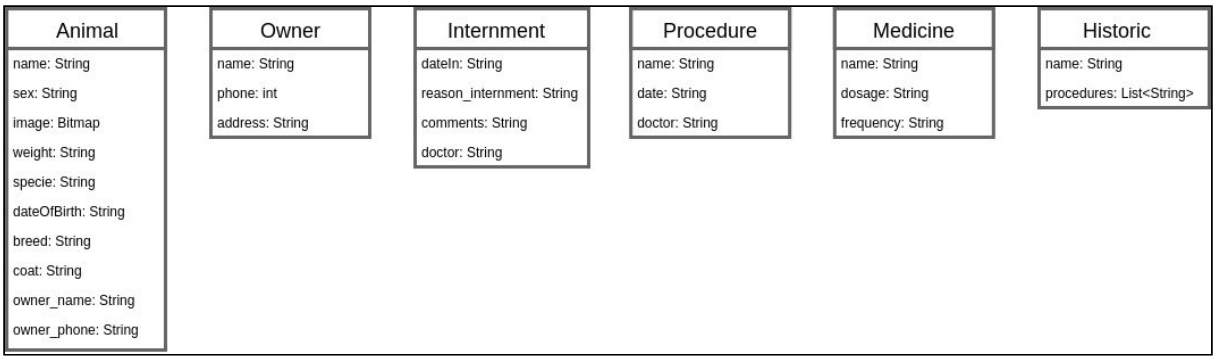

Fig. 11: Modelos de Dados

Back-end

A parte de autenticação e armazenamento de dados foi realizada através da plataforma Firebase. Consideramos que o uso desta plataforma foi de extrema importância na nossa aplicação pois permite a autenticação de várias clínicas veterinárias e permite o armazenamento de dados importantes acerca dos animais, informações essas dispostas nas diferentes UI para fácil acesso.

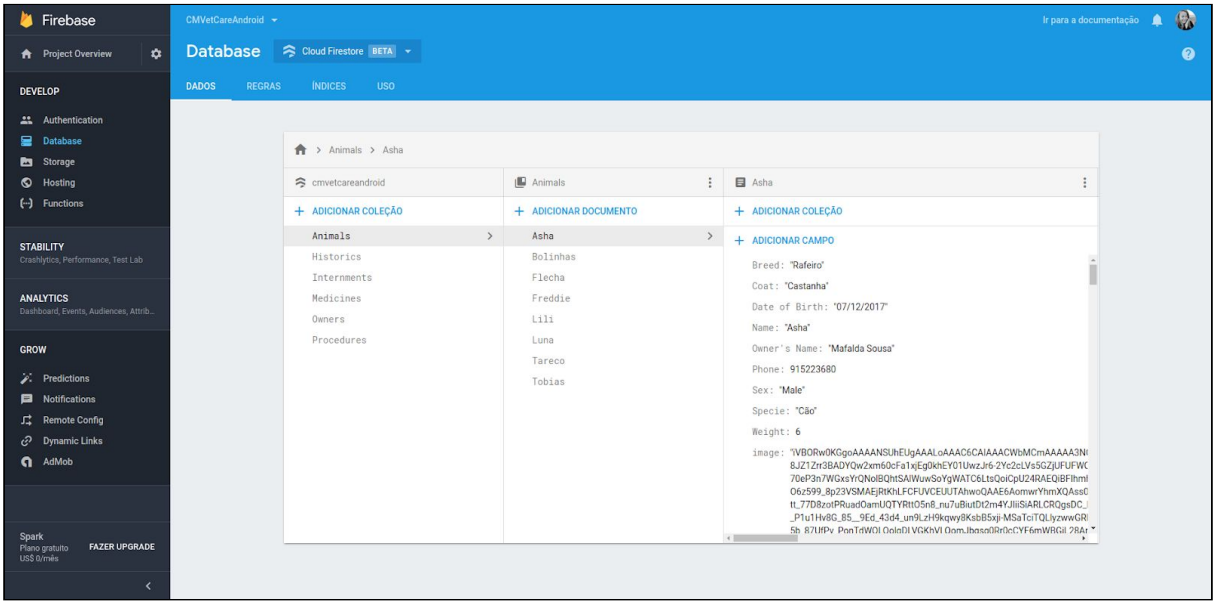

Fig. 12: Dados no Firebase

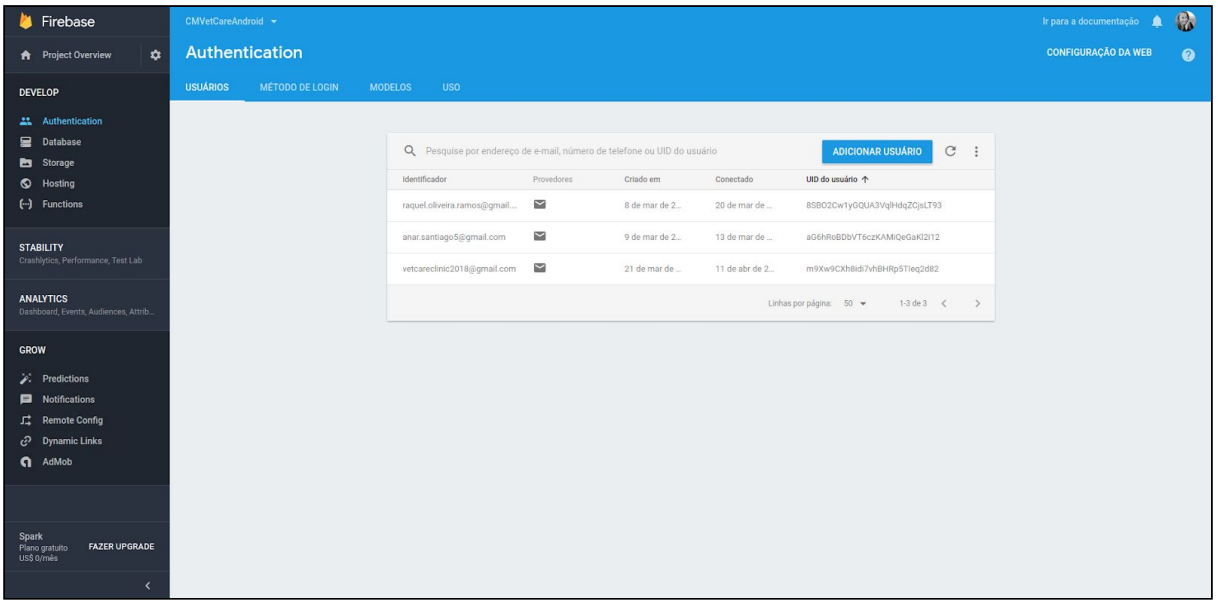

Fig. 13: Autenticação no Firebase

Persistência

Relativamente à persistência, como já foi referido, os dados são todos armazenados na plataforma Firebase levando a um fácil acesso aos mesmos.

Acesso a API

Como se pode observar no diagrama, foi feito o acesso e uso de duas API da Google: Google Calendar e Google Drive.

Hoje em dia o uso da Internet e das tecnologias existentes, encontra-se num crescimento rápido e em grande escala. Desta maneira e como uma forma de haver um fácil e rápido acesso, a partir de qualquer local, dos vários funcionários da clínica veterinária a diversas informações, implementou-se o acesso ao Google Calendar para o armazenamento de tarefas a realizar e o acesso à Google Drive onde estarão dispostos documentos de alta ou documentos com conclusões acerca de exames realizados como por exemplo, análises ao sangue. Deste modo, em qualquer altura e em qualquer lugar, desde que haja acesso à Internet, qualquer funcionário da clínica veterinária consegue ter acesso a estas informações, seja para consulta ou possíveis alterações, não estando as mesmas apenas associadas a armazenamento local de ficheiros ou armazenamento das tarefas no calendário do telemóvel.

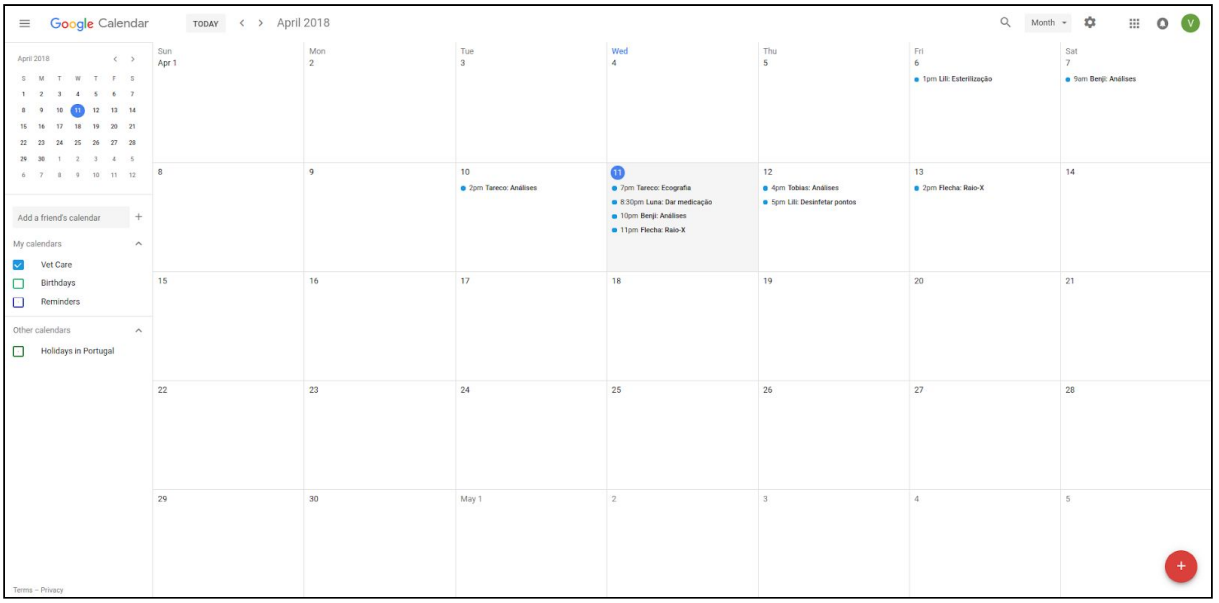

Fig. 14: Google Calendar

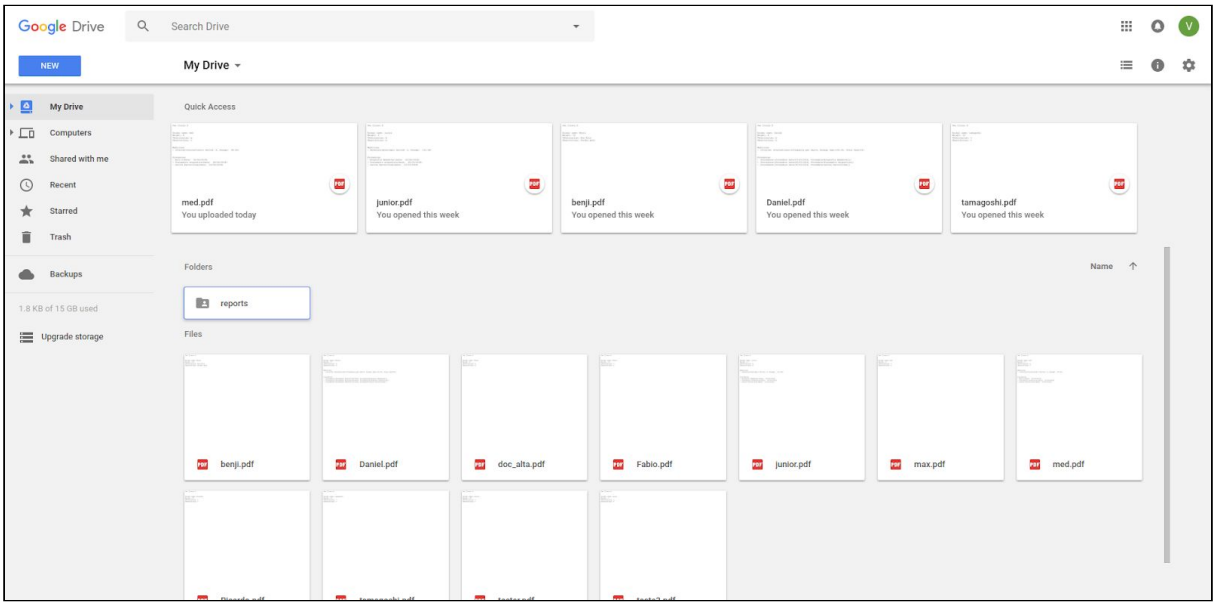

Fig. 15: Google Drive

Mecanismos de Notificações

Notificações são uma boa maneira de manter o utilizador a par do que a aplicação vai realizando. No VetCare, consideramos que uma boa maneira de manter o utilizador a par do que se vai realizando, seria através de *Toast*. Por exemplo, quando um animal é inserido na lista de animais internados, é possível visualizar-se um Toast em como o animal foi adicionado com sucesso.

Relativamente às tarefas a realizar, com o uso da API do Google Calendar, encontra-se implementado a recepção de notificações quando está a chegar a hora de

realizar determinada tarefa. Esta era sem dúvida uma das funcionalidades que tinha que ser implementada pois o objectivo principal da aplicação era ajudar na realização dos cuidados aos animais internados, e foi implementada com sucesso.

Outro tipo de notificações que se encontra implementado, é o aparecimento de um *Alerter* na parte superior do ecrã do dispositivo móvel assim que se identifica um animal. Esta funcionalidade será descrita no capítulo acerca da solução implementada.

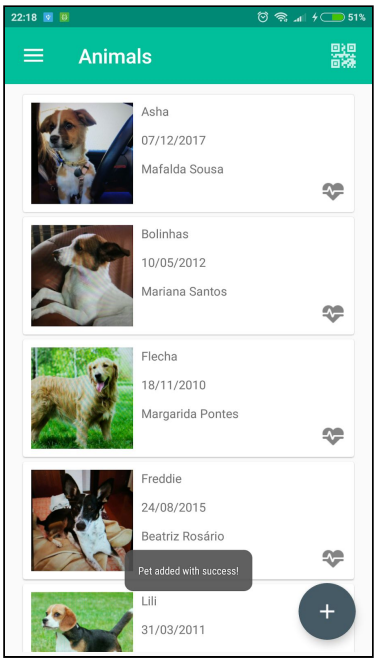

Fig. 16: Toast

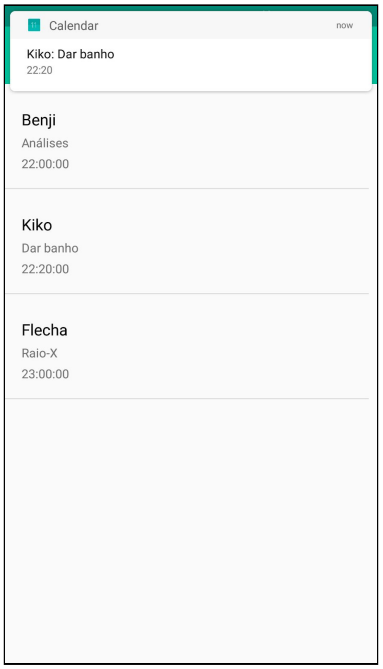

Fig. 17: Notificações do Calendário

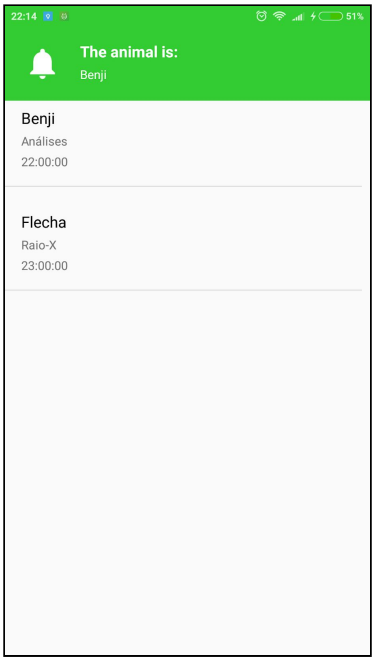

Fig. 18: Alerter

Padrões e Componentes da Arquitetura

O padrão de arquitetura implementada foi o MVC. Este foi o padrão que foi implementado desde o início do desenvolvimento do projeto pois o grupo ainda não tinha conhecimento de outros padrões. Quando foi realizada a aula acerca de padrões de arquitetura em Android, o projeto já se encontrava numa fase bastante avançada e ainda se encontravam algumas funcionalidades bastante importantes e trabalhosas por se desenvolver daí que não foi possível alterar-se o padrão de arquitetura para MVVM.

5 Solução Implementada

Organização do Código

Quando se desenvolve um projeto, todo o código deve estar organizado. Não só por uma questão de organização mas sim porque se por exemplo, o projeto não for desenvolvido apenas individualmente, uma boa organização do código será vital para que outras pessoas envolvidas no projeto, consigam encontrar os ficheiros que pretendem.

Desta maneira, os ficheiros Java foram agrupados em diretórios consoante a sua funcionalidade e o “local” onde se encontram. A organização do código pode ser observada na seguinte imagem.

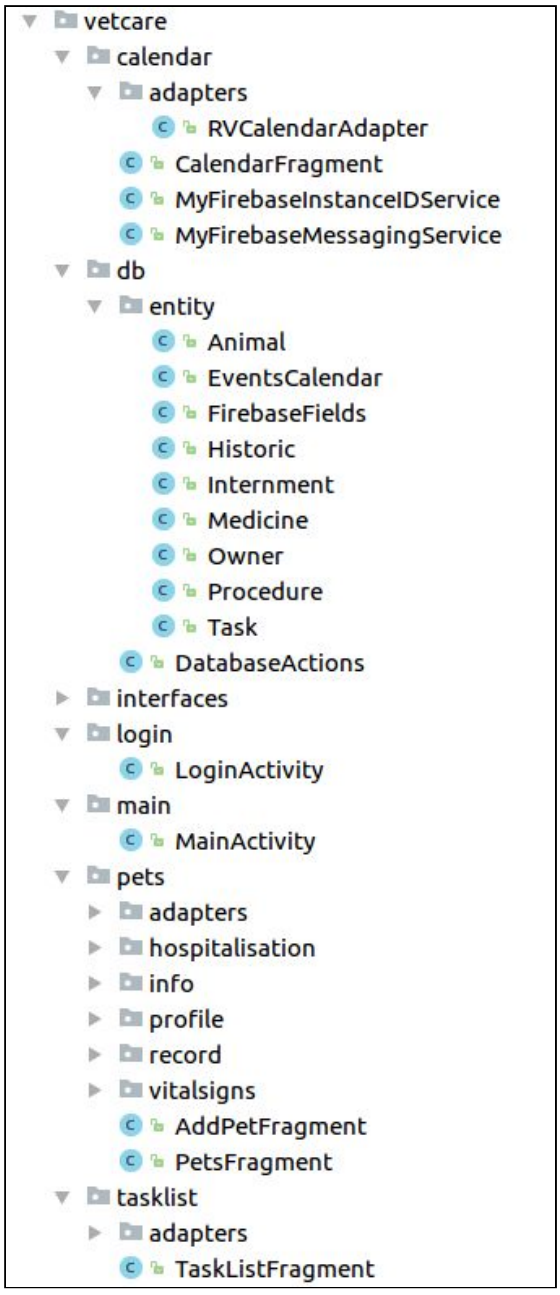

Fig. 19: Organização do Projeto

Relativamente aos diretórios de código:

- calendar: diretório onde se encontra o fragmento que tem definido no seu *layout* o calendário (sincronizado com o Google Calendar) com todos os eventos, sejam do presente dia ou futuros;
- db: diretório onde se encontram definidas as várias entidades do projeto;
- interfaces: diretório com interfaces definidas;
- login: diretório que contém a LoginActivity, ou seja, a *Activity* que diz respeito ao ecrã inicial da aplicação onde é feito o login da clínica veterinária;
- main: diretório que contém a MainActivity, ou seja, a *Activity* a que estão ligados todos os fragmentos definidos que dizem respeito às várias funcionalidades da aplicação;
- pets: diretório que contém todos os fragmentos respetivos aos animais nomeadamente o fragmento do perfil de cada um e por exemplo os simuladores dos seus sinais vitais;
- tasklist: diretório que diz respeito ao fragmento inicial quando se inicia sessão na aplicação, e que contém uma *RecyclerView* com todas as tarefas para o presente dia;

Os vários diretórios *adapters* dizem respeito ao vários adapters que vão preencher as *RecyclerViews* a que estão associados.

Funcionalidades Implementadas

Desde o dia da escolha do projeto, que foi definida a história da aplicação e as funcionalidades a implementar. Com a ajuda do professor, algumas funcionalidades foram sofrendo alterações de modo a ficarem melhores para a experiência do utilizador seja em termos visuais ou em termos técnicos.

A primeira funcionalidade implementada foi o Login. Isto é possível realizar-se com a autenticação disponibilizada pelo Firebase.

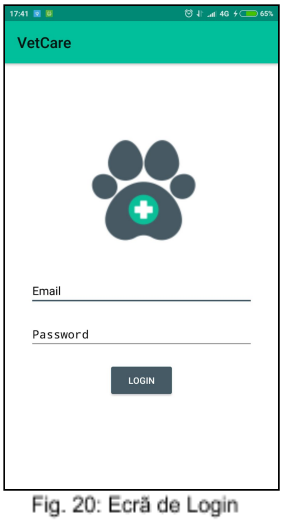

Fig. 20: Ecrã de Login

Após o utilizador fazer o login, o primeiro fragmento a ser disposto ao mesmo, será a Task List. Esta lista de tarefas irá conter todas as tarefas do dia que determinado funcionário deverá realizar. A cada tarefa estará associada uma hora, um animal e a tarefa a realizar. Após a realização da tarefa, o funcionário em questão que a realizou, dispõe de um leitor de QR code implementado que ao ler-se um QR Code associado a um animal (QR code que contém como informação o nome do animal cuja tarefa vai ser realizada), dará check na tarefa associado a esse mesmo animal. Para que o utilizador aceda a esse leitor, existe um ícone na *Toolbar* (cuja imagem é um QR Code) que o utilizador deverá premir, levando a que o leitor seja inicializado. Um *Alerter* será disposto ao olho do utilizador no topo do ecrã do dispositivo móvel, dizendo que a tarefa foi concluída com sucesso. Para se atualizar a lista de tarefas, como esta tem que fazer acesso à API do Google Calendar (efetuado por uma *AsyncTask*), o utilizador terá que fazer um pequeno deslize com o seu dedo em direcção ao fundo do ecrã do dispositivo móvel, levando a aparecer um ícone de *refresh* (implementação do *Swipe-to-Refresh*) cuja ação associada a este movimento, é um novo acesso à API para ir buscar os eventos atualizados.

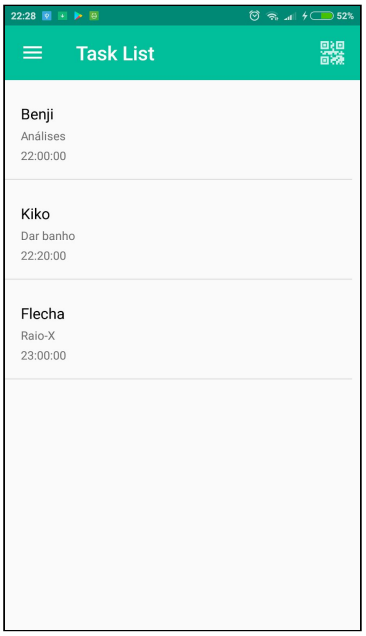

Fig. 21: Task List

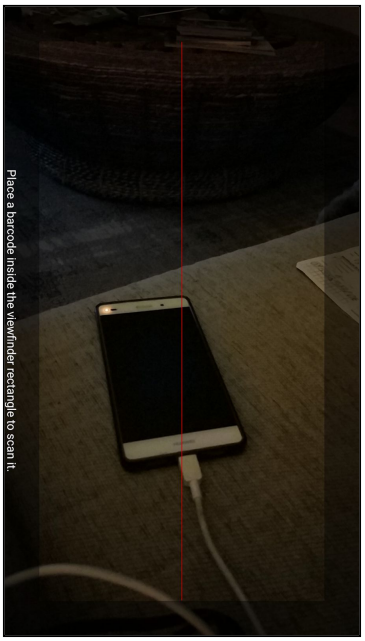

Fig. 22: Leitor QR Code

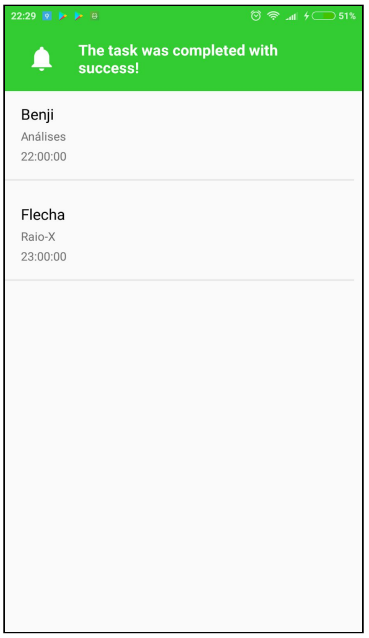

Fig. 23: Alerter

Uma das principais e mais importantes funcionalidades implementadas foi a rápida identificação dos animais. Esta funcionalidade foi implementada recorrendo ao NFC. Foi assumido que quando o animal dá entrada na clínica veterinária, é-lhe fornecida uma coleira com uma tag NFC que contém como informação interna o nome do animal em questão. Assim, sempre que um funcionário pretender identificar um animal rapidamente, basta encostar o seu dispositivo móvel à coleira do animal e será disposto um *Alerter* no topo da UI com a identificação do animal. Esta funcionalidade foi simulada com dois telemóveis pois nenhum dos elementos do grupo continha tags NFC. Sendo assim, num telemóvel foi desenvolvida uma pequena aplicação que simula uma tag NFC (sendo introduzido o nome que estaria incluído na tag) e ao encostar o outro telemóvel, esta tag NFC é lida e é apresentado ao utilizador o resultado da mesma.

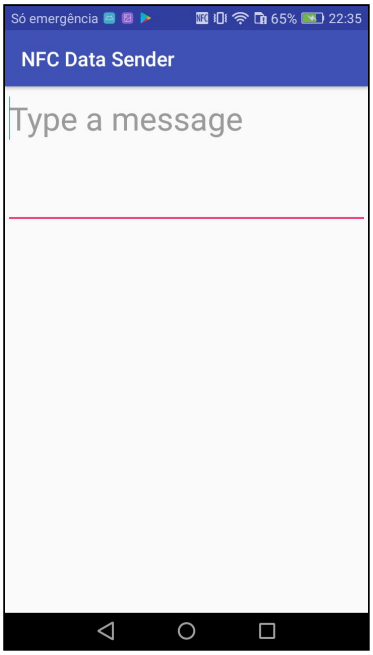

Fig. 24: Simulador Tag NFC

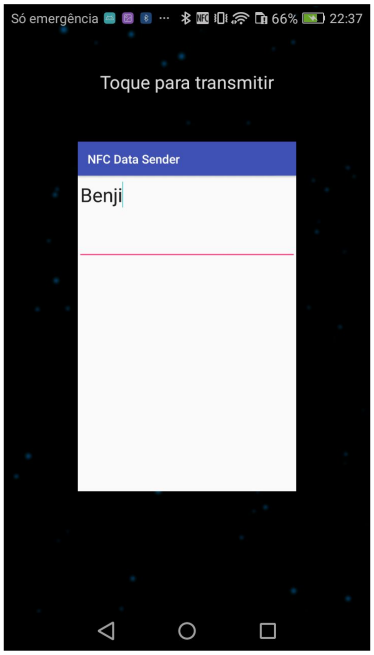

Fig. 25: Android Beam

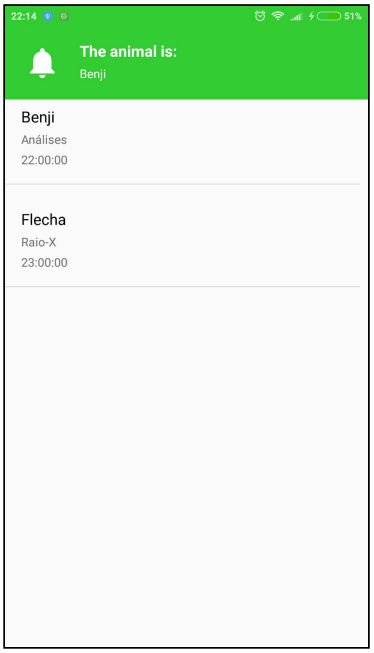

Fig. 26: Alerter

Com o *Navigation Drawer,* é possível navegar entre todos os fragmentos implementados. O próximo fragmento é o dos Animais. Este fragmento é possivelmente um dos mais importantes pois contém toda a informação relativa aos animais internados. Ao navegar para o fragmento dos animais (*Animals* no *Navigation Drawer*), é apresentado uma *RecyclerView* com todos os animais e algumas informações. Ao carregar-se num *card*, o utilizador é redireccionado para o perfil do animal em questão. Caso no *card* seja pressionado o ícone dos sinais vitais, o utilizador é redireccionado para um fragmento que apresenta dois gráficos com os sinais vitais do animal nas últimas 24 horas, nomeadamente os batimentos cardíacos e a temperatura.

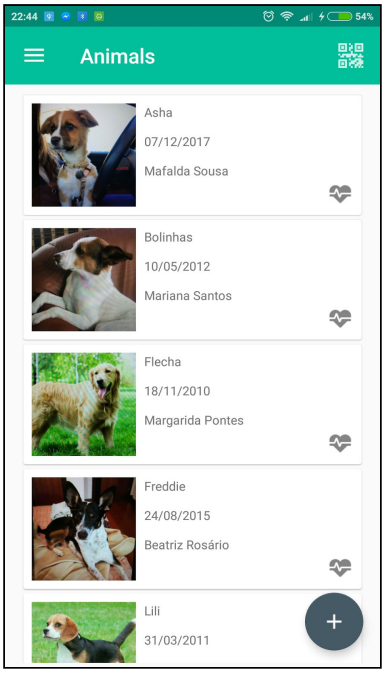

Fig. 27: Lista de Animais

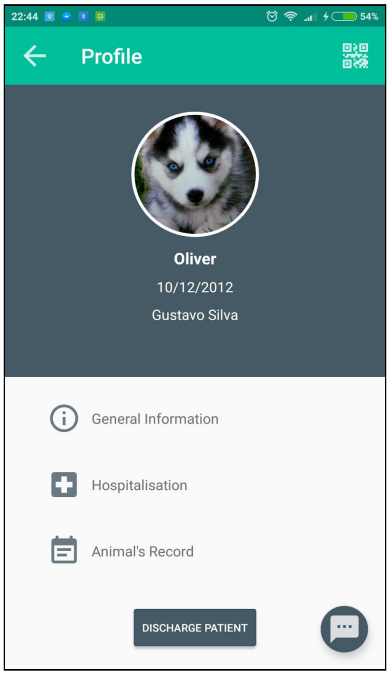

Fig. 28: Perfil do Animal

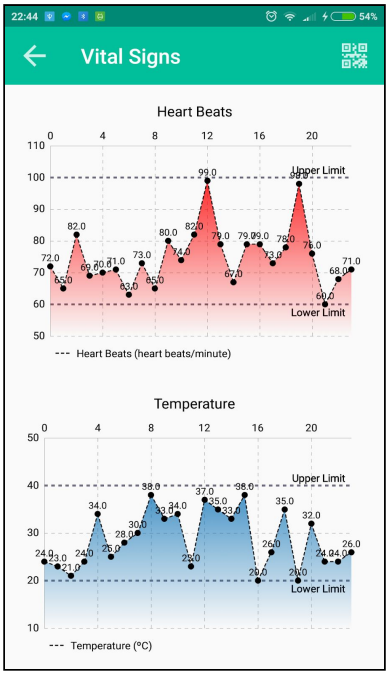

Fig. 29: Sinais Vitais

Dentro do perfil de cada animal, podem ser observadas 3 categorias principais: General Information, Hospitalisation e Animal's Record. O fragmento respectivo à General Information dispõe de todas as informações básicas acerca do animal em questão.

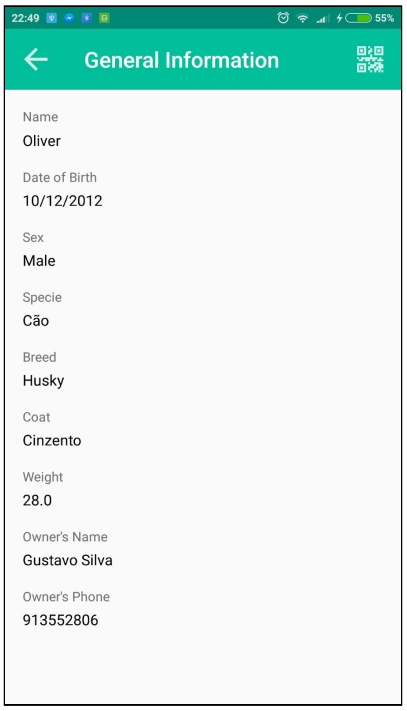

Fig. 30: Informação Geral

Ainda dentro do perfil é possível aceder-se ao fragmento da Hospitalisation. Neste fragmento é possível observar-se 3 *tabs*: General, Medication e Procedures. Na *tab* General, encontram-se informações respectivas ao dia em que o animal deu entrada na clínica veterinária. Na *tab* Medication encontra-se informação acerca de medicamentos que o animal se encontra a tomar. E na última *tab*, a tab dos Procedures, encontram-se todos os procedimentos já realizados pelo animal desde que este foi internado.

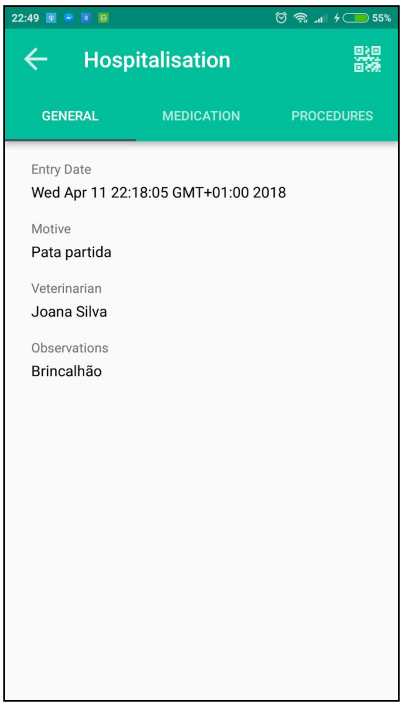

Fig. 31: Informação de Entrada

Fig. 32: Medicação

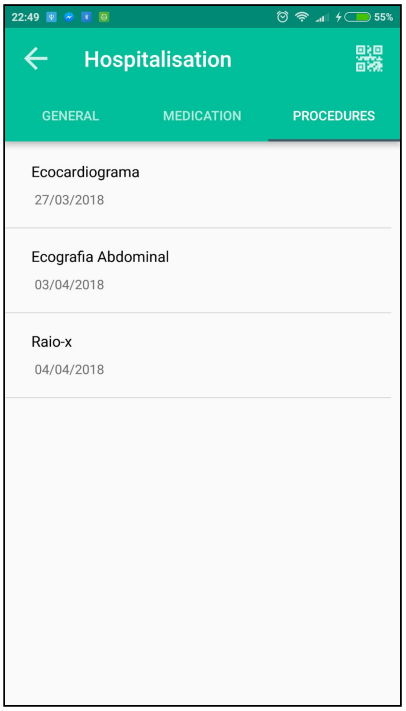

Fig. 33: Procedimentos

Voltando ao perfil do animal, é possível ainda ver-se mais informação acerca do mesmo, pressionando a secção Animal's Record. Este clique levará o utilizador para o fragmento onde se encontra disposta informação acerca de consultas prévias do animal e nomeadamente que procedimentos foram realizados nessas consultas.

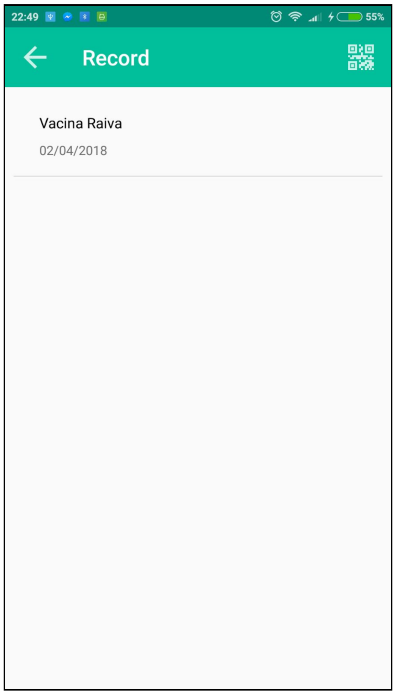

Fig. 34: Histórico

No perfil do animal, o utilizador tem ainda ao seu dispor dois botões: um botão com um ícone de mensagem e um botão com o texto “Discharge Patient”. O primeiro botão aparece na forma de *Floating Action Button* (no canto inferior direito) e ao ser premido, encaminha o utilizador para um novo fragmento onde pode ser escrita uma mensagem. Este botão e este novo fragmento trazem ao utilizador a possibilidade de poder mandar mensagens para os donos dos animais (sem ter que inserir o número de telemóvel do dono pois esta informação é obtida diretamente do Firebase) para ir dando actualizações do estado do animal ao dono ou para por exemplo, avisar após um procedimento, que correu tudo bem e o animal se encontra melhor. A primeira vez que o utilizador tentar utilizar esta funcionalidade, será pedida permissão para aceder ao envio de mensagens através do dispositivo móvel que se encontra a ser usado. Após a mensagem ser enviada, é mostrado ao utilizador um Toast para este saber que a mensagem foi enviada com sucesso.

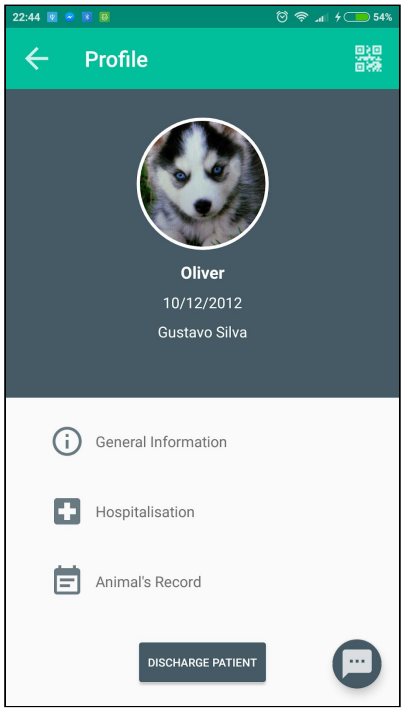

Fig. 35: Perfil do Animal

Fig. 36: Envio de Mensagem

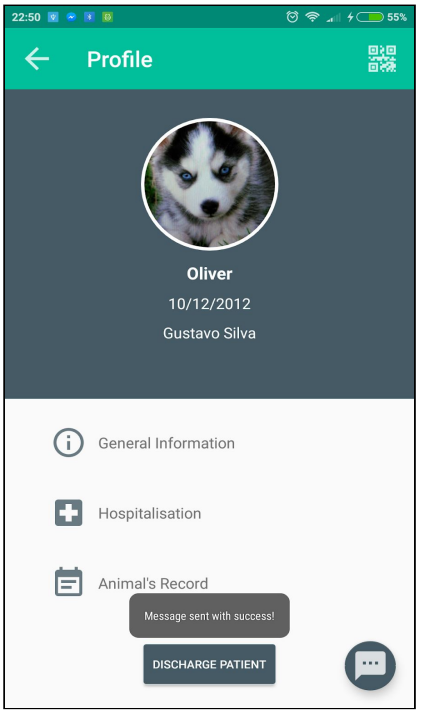

Fig. 37: Toast

Relativamente ao segundo botão que já foi referido e que contém o texto “Discharge Patient”, este botão servirá para dar alta a um animal. Ora portanto, assim que este botão é premido, o utilizador irá ser reencaminhado para um fragmento onde deverá preencher algumas informações: o peso com que o animal saiu da clínica (pois quando os animais estão internados e se encontram a tomar medicação e a realizarem procedimentos, existe tendência para uma mudança significativa no peso), o nome do veterinário que deu a alta e possíveis observações. Assim que neste fragmento for premido o botão “Generate PDF” que se encontra no fim da UI, será gerado um documento PDF com o nome do animal, as informações preenchidas no fragmento e os medicamentos administrados e procedimentos realizados durante o internamento. Este documento será armazenado tanto localmente como na Drive. Após o botão “Generate PDF” ser premido, ainda vai aparecer uma caixa ao utilizador em que este poderá alterar o nome do ficheiro PDF ou ainda alterar o local onde o quer guardar na Drive.

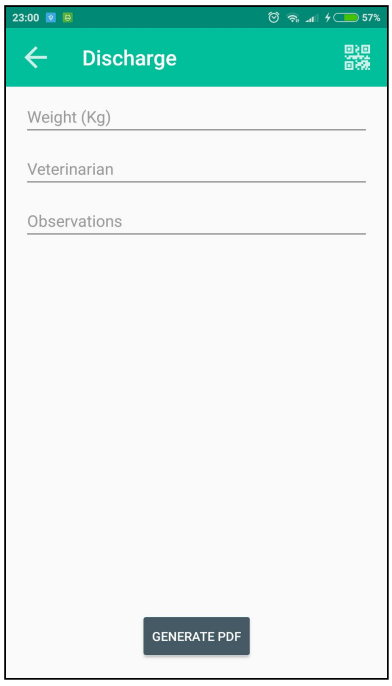

Fig. 38: Informação para o PDF

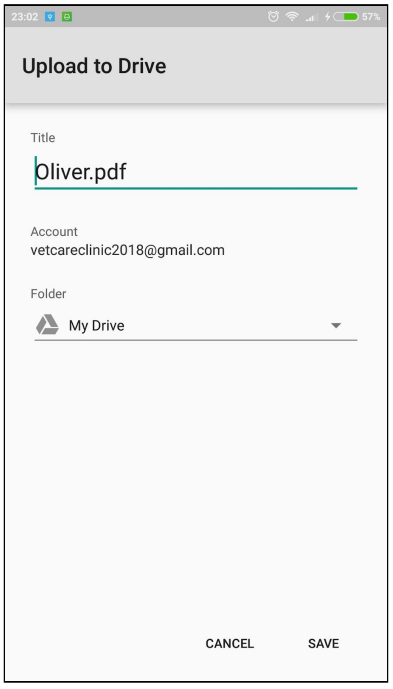

Fig. 39: Armazenamento na Drive

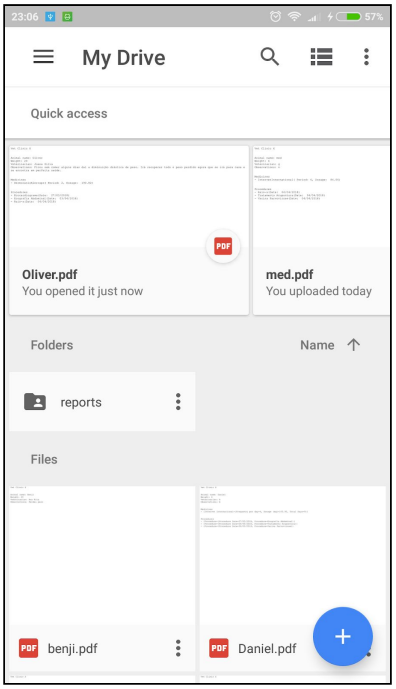

Fig. 40: Google Drive

Para além de todas as funcionalidades ligadas aos animais individualmente, na lista inicial onde estão dispostos todos os animais internados, é possível observar-se um *Floating Action Button* com o ícone de "+" cuja funcionalidade do mesmo é adicionar um novo animal. Quando se efetua um clique neste botão, o utilizador é reencaminhado para um fragmento com vários campos *EditText* que terão que ser preenchidos de modo a um novo animal ser adicionado à lista de animais internados. Para além do preenchimento dos campos, neste primeiro fragmento, é também necessário tirar-se uma foto ao animal, foto esta que será disposta na lista de animais e no perfil de cada animal. Após o preenchimento dos campos e a captura da foto, o utilizador tem um *Floating Action Button* no canto inferior direito da UI com a forma de uma seta (com o propósito de ser um "next") e este botão ao ser premido levará o utilizador para um novo fragmento que é o último no que diz respeito à adição de informação em relação ao animal. Este fragmento contém vários campos *EditText* que têm que ser preenchidos, nomeadamente: o motivo pelo qual o animal foi internado, que médico o internou e observações do animal como por exemplo se pode morder ou alguma alergia, ou outra observação importante que deve ser estritamente necessário os funcionários da clínica saberem.

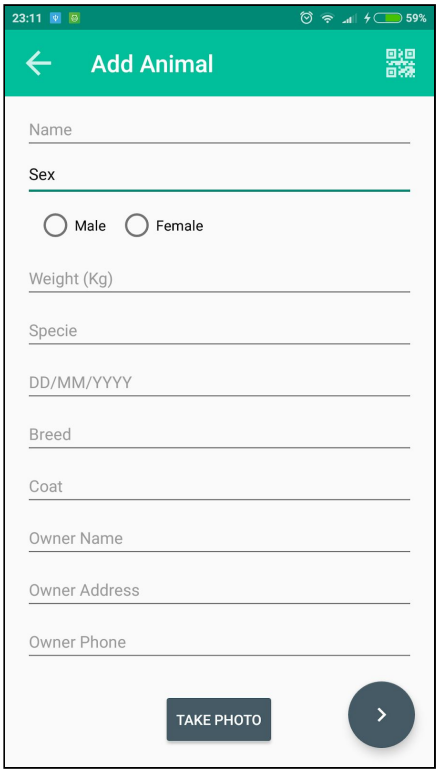

Fig. 41: Informação do Animal

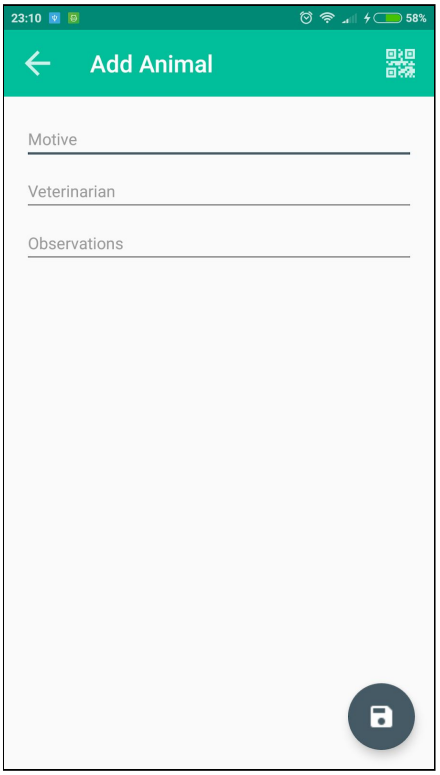

Fig. 42: Informação do Internamento

Voltando ao *Navigation Drawer*, irá ser agora abordado um novo Fragmento: *Calendar*. O utilizador pode navegar para este fragmento carregando no item "Calendar" no *Navigation Drawer*. Este fragmento é composto por um calendário com indicação do mês e ano por cima do mesmo e por baixo do calendário é possível observar-se uma *RecyclerView* com as tarefas a realizar no dia em que o utilizador clicar. Neste calendário é possível observar-se as tarefas do presente dia, tarefas de dias anteriores e futuras tarefas. O calendário disposto encontra-se ainda sincronizado com o Google Calendar sendo que quem possuir os dados da conta associada à clínica veterinária, consegue consultar em qualquer local e em qualquer altura (desde que tenha acesso à Internet), todas as tarefas associadas à clínica seja pela aplicação ou pelo Google Calendar.

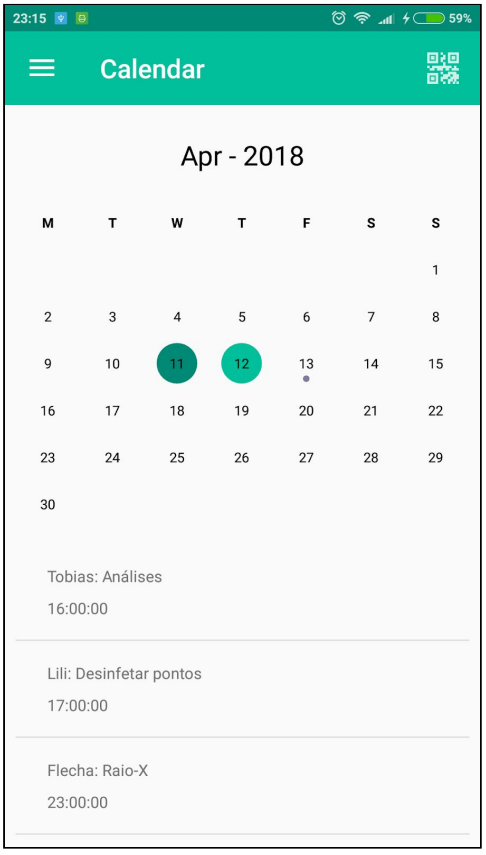

Fig. 43: Calendário

No *Navigation Drawer*, existe um item denominado “Documents” e com o ícone da Google Drive do seu lado esquerdo. Este item tem como objetivo que, ao ser pressionado, o utilizador seja reencaminhado para a Google Drive onde estão guardados os mais variados documentos acerca dos animais que atualmente se encontram internados ou já estiveram.

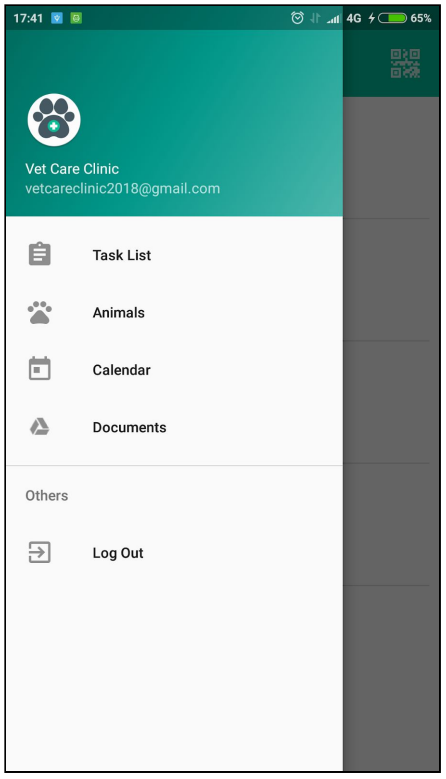

Fig. 44: Navigation Drawer

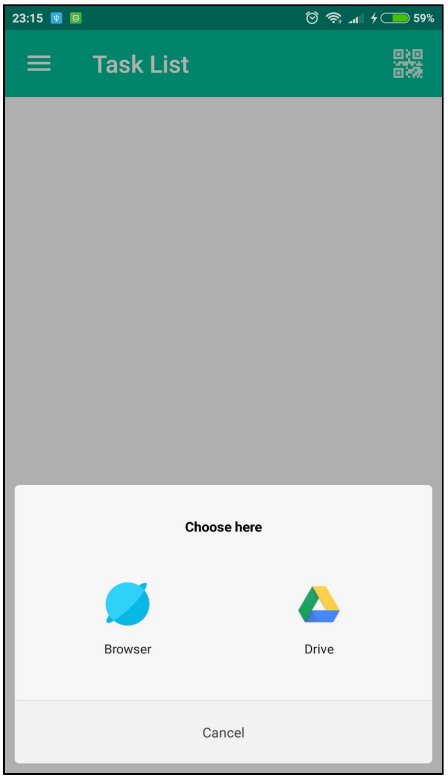

Fig. 45: Acesso à Drive

No *Navigation Drawer* existe ainda a possibilidade de fazer log out (como é possível verificar na primeira imagem apresentada acima) levando o utilizador a ser reencaminhado para a Login Activity.

Para finalizar, uma outra funcionalidade que foi implementada foi a hipótese de a aplicação se encontrar com o texto em Português ou Inglês. Através do diretório *values*, do diretório *values-pt* e do ficheiro *strings.xml* (onde o ficheiro *strings.xml* contém strings em inglês no caso do diretório *values* e se for o ficheiro *strings.xml* da pasta *values-pt*, as strings se encontram em português), foram definidas as strings para português ou inglês. Quando a aplicação é inicializada, o telemóvel detecta a localização do utilizador e caso o mesmo se encontre em Portugal, todas as strings da aplicação se encontrarão em português. Caso contrário, as strings estarão em inglês pois é a linguagem *default*. De referir ainda que todas as strings se encontram definidas no ficheiro *strings.xml* em vez de ser *hardcoded* dada que esta é uma boa prática de programação em Android.

Uso de Bibliotecas

Foram usadas diversas bibliotecas durante o desenvolvimento do projeto. As bibliotecas usadas foram:

- Zxing: usada para a leitura do QR code;

- Alerter: usada para lançar um Alerter que aparece no topo da UI (componente mais apelativo ao olho humano que Toasts e Snackbars);
- CircleImageView: usada para tornar as imagens do perfil circulares;
- CompactCalendarView: usada para criar o calendário presente no fragmento Calendar;
- iText: usada para a geração dos PDF's com pedaços de texto;
- MPAndroidChart: usada para a geração dos gráficos que contém os valores dos sinais vitais os animais.

O link para qualquer uma destas bibliotecas encontra-se no capítulo das Referências.

Testes

Em termos de testes, não foram realizados testes unitários mas a aplicação foi testada com o Monkey (500 toques). Inicialmente obteve-se alguns erros com rotações e cliques mas estes erros foram sendo corrigidos sendo que no final, a aplicação se encontrava a funcionar com sucesso.

Foi feita ainda a análise do código pelo Lint disponibilizado pelo Android Studio. O uso desta ferramenta foi importante no despiste de erros prévio à execução da aplicação. Permitiu que o projeto ficasse mais "limpo" pois foram corrigidos warnings, naming conventions, ficheiros e imports que não se encontravam em uso, entre outros.

6 Conclusão

Os objetivos do projeto foram concluídos com sucesso. Todas as funcionalidades pensadas inicialmente e também outras que foram sugeridas pelo professor posteriormente, ficaram implementadas.

Foi um trabalho que envolveu bastante pesquisa e bastante aprendizagem de novos conhecimentos num curto espaço de tempo mas que no fim revelou ser muito gratificante para ambos os elementos do grupo de trabalho pois concluímos que aprendemos bastante. Alguns dos maiores problemas foi a inclusão da API do Google Calendar e da Google Drive pois muita da documentação está desatualizada ou não se encontra muito explícita e esta documentação não ajudou muito levando a um elevado número de horas necessário para se conseguir implementar algo com estas API pois não se encontrava muita informação atual sobre as mesmas. Outro dos grandes problemas foi a geração dos documentos PDF pois conseguir guardar localmente foi fácil mas conseguir ter o acesso à Google Drive e conseguir armazenar os mesmos lá não foi de todo uma tarefa fácil. Outro problema sentido foi através do QR Code conseguir dar uma tarefa por concluída e conseguir transmitir para o Google Calendar a sincronização em como determinada tarefa já tinha sido concluído. Por último, o uso do Firebase e Firestore foi também uma dificuldade algo elevada mas ultrapassada com sucesso. Como esta é uma ferramenta que não foi abordada nas aulas e é uma ferramenta com bastantes detalhes e não de muito fácil implementação, foram precisas bastantes horas de trabalho inicialmente para perceber como lidar e trabalhar com o Firebase e depois adequar ao projeto desenvolvido. De um modo geral, foi um projeto trabalhoso mas muito gratificante no final e que levou a uma grande aprendizagem de várias tecnologias.

A unidade curricular de Computação Móvel na componente de Android tem uma boa organização, os conteúdos são bem leccionados e o grau de satisfação dos elementos do grupo com a unidade curricular é bastante elevada.

7 Referências e Recursos

Comunicação de Fragmentos
Estilos e Temas
AsyncTasks
MPAndroidChart
Cards
Cores
Ícones
Acesso à Câmara
QR Code Scanner
Firebase
CompactCalendarView
Google Calendar API
Shared Preferences
NFC
Organização de Código
Lint

Summary of project resources:

Project resources for the Android module:

- Code repository: http://code.ua.pt//projects/cm2018_vetcare/repository
- Ready-to-deploy APK: https://drive.google.com/drive/folders/1nsSpWroQjWpHo4hH2gBHe-eoAKKG36jt?usp=sharing