

Universidade do Minho

Escola de Engenharia

Departamento de Informática

A83631 – Filipa Santos

A85006 – Hugo Cardoso

A84775 – João Costa

A67656 – Rui Santos

A84464 – Válder Carvalho

SOS Animal



Universidade do Minho

Escola de Engenharia

Departamento de Informática

A83631 – Filipa Santos

A85006 – Hugo Cardoso

A84775 – João Costa

A67656 – Rui Santos

A84464 – Válder Carvalho

SOS Animal

Laboratórios de Informática IV

Mestrado Integrado em Engenharia Informática

Trabalho efetuado sob orientação de

José Manuel Ferreira Machado

Julho 2020

RESUMO

Trabalho prático feito no âmbito da unidade curricular Laboratórios de Informática IV do 3º ano do Mestrado Integrado de Engenharia Informática da Universidade do Minho, no ano letivo 2019/2020. Foi solicitado uma interface web ou mobile, com uma base de dados ativa, recorrendo a API e WebServices e com um backoffice implementado. O grupo decidiu criar uma aplicação Android, utilizando tecnologias como C, MySQL, Google Maps e Xamarin Forms.

CONTEÚDO

1	INTRODUÇÃO	1
1.1	Contexto	1
1.2	Motivação	1
1.3	Objetivos	2
1.4	Estrutura	2
2	PESQUISA	3
2.1	Competição	3
2.2	Financiamento	3
3	PROPOSTA	4
3.1	Objetivos	4
3.2	Requerimentos	5
3.3	Cronograma	5
3.4	Mockups	7
3.5	Diagramas	11
4	DESENVOLVIMENTO	13
4.1	Decisões	13
4.2	Implementação	14
4.2.1	Backend	14
4.2.2	Frontend	16
4.2.3	Funcionalidades da Aplicação	17
4.2.4	BackOffice	32
5	CONCLUSÃO	36
5.1	Conclusões	36
5.2	Trabalho Futuro	36

LISTA DE FIGURAS

Figura 1	Cronograma do Projeto	6
Figura 2	Página de Registo	7
Figura 3	Página de Log-in	7
Figura 4	Menu de Entrada	8
Figura 5	Página de Perfil	8
Figura 6	Agendar uma Visita	9
Figura 7	Lista de Animais Abrigados	9
Figura 8	Histórico do Utilizador	10
Figura 9	Menu de Sinalização	10
Figura 10	Diagrama de Classes do Projeto	11
Figura 11	Diagrama de Use Cases do Projeto	12
Figura 12	Api	14
Figura 13	Esquema da base de dados	15
Figura 14	Função Login	15
Figura 15	Função Login	16
Figura 16	Menu de Login	17
Figura 17	Avisos de Login	18
Figura 18	Menu Principal	18
Figura 19	Menu de Registo	19
Figura 20	Avisos de Registo	19
Figura 21	Página de Sinalizar	20
Figura 22	Páginas para inserir localização e foto na sinalização	21
Figura 23	Avisos de Sinalização	21
Figura 24	Histórico de sinalizações	22
Figura 25	Detalhe de uma sinalização	23
Figura 26	Perfil de um utilizador	24
Figura 27	Avisos ao editar o nome	25
Figura 28	Avisos ao editar a password	25
Figura 29	Página para agendar visita	26
Figura 30	Páginas para inserir data e hora da visita	27
Figura 31	Avisos no agendamento de uma visita	27
Figura 32	Página das visitas	28
Figura 33	Detalhes de uma visita	29

Figura 34	Página dos animais abrigados	30
Figura 35	Detalhes de um animal abrigado	31
Figura 36	Tabela de Utilizadores	33
Figura 37	Secção de login do website	34
Figura 38	Panorama Geral do Website	35

INTRODUÇÃO

Nesta primeira secção do relatório, iremos explicar o contexto em que este trabalho foi feito, a motivação por detrás da nossa ideia, os objetivos estabelecidos e, por fim, a estrutura do resto do documento.

1.1 CONTEXTO

Este projeto foi feito no contexto da unidade curricular Laboratórios de Informática IV. É um trabalho realizado por um grupo de 5 alunos do Mestrado Integrado de Engenharia Informática, da Universidade do Minho. Foi desenvolvido no 2º semestre do ano letivo 2019/2020, orientado pela equipa docente através de apresentações quinzenais. Devido às circunstâncias únicas deste semestre, todas estas apresentações intermédias foram feitas através de chamadas online. O conceito da aplicação pedida foi decidido pelo grupo, embora com a aprovação do professor responsável, e seguiu certos requisitos impostos pelos docentes.

1.2 MOTIVAÇÃO

Decidimos criar a aplicação Android "SOS Animal": uma plataforma para sinalizar animais perdidos, abandonados e até em situações críticas de saúde. A motivação por detrás da tecnologia escolhida foi o interesse por aprender algo útil e atual hoje em dia e também consideramos que uma app mobile Android seria eficaz para implementarmos a nossa ideia e simples o suficiente para conseguirmos completar tudo o desejado, dentro do prazo. Já sobre o tema específico escolhido, este foi determinado por experiências pessoais dos membros do grupo e porque também consideramos que é um serviço ainda pouco explorado em Portugal, o que vai ser explicado abaixo em mais detalhadamente.

1.3 OBJETIVOS

Como todas as disciplinas de Laboratórios de Informática, os principais objetivos deste tipo de projetos é desenvolver a autonomia de cada grupo ao criar um programa inteiro e encorajar o trabalho conjunto entre os membros. Neste caso específico, os objetivos indicados foram os seguintes:

- Delineamento de uma estratégia holística e flexível de desenvolvimento;
- Desenvolvimento iterativo e incremental ao longo do projeto;
- Desenvolvimento ágil, progressivo e melhorado do produto;
- Coesão e melhoria progressiva da dinâmica da equipa e da sua comunicação interna diária;
- Adoção de uma metodologia própria de gestão do projeto para discussão quinzenal com o docente responsável.

Resumindo estes pontos, podemos concluir que neste trabalho era solicitado que os alunos criassem um plano de desenvolvimento do projeto de modo a ser ajustável, progressivo e adequado para as apresentações agendadas e que, simultaneamente, fossem melhorando habilidade de trabalhar em conjunto, de forma coesa.

1.4 ESTRUTURA

O relatório terá a seguinte estrutura:

- Pesquisa: descrição da pesquisa feita antes da execução do programa, como aplicações semelhantes já existentes e possíveis meios de financiamento, no cenário hipotético de se tornar real.
- Proposta: ideia inicial do projeto, com os requisitos e objetivos pensados no começo do trabalho. Também é apresentado o cronograma seguido, os diagramas e os mockups feitos.
- Desenvolvimento: descrição das decisões tomadas, da backend, frontend, backoffice e de todas as funcionalidades da aplicação.
- Conclusão: conclusões que retirámos do trabalho, melhorias possíveis a fazer e pensamentos sobre trabalhos futuros.

PESQUISA

Esta parte do relatório tem como objetivo expor a pesquisa feita para verificar a viabilidade do conceito proposto, tanto em termos da sua necessidade como em termos financeiros.

2.1 COMPETIÇÃO

A disponibilização de ambulâncias animais é uma realidade muito recente em Portugal, sendo que ainda só estão disponíveis em poucas cidades. Exemplos destas ambulâncias podem-se encontrar em Lisboa, fornecida pela *Casa dos Animais de Lisboa (CAL)*, e em Braga, pela *Agere*. Porém, a maior parte dos serviços existentes apenas socorre animais acidentados na via pública, não oferecendo assistência em casos de emergências domiciliares. Como tal, é uma área que ainda tem muito por explorar, especialmente no que toca à transição para tecnologias mais modernas, nomeadamente aplicações de telemóvel de uso facilitado, rápido e eficiente. Surgiu assim a ideia de criar este serviço, de modo a satisfazer esta escassez de serviços de urgência para animais.

2.2 FINANCIAMENTO

O projeto funcionará bem se for adotada por alguma empresa com centros de adoção e/ou clínicas de tratamento e que tivesse a disponibilidade de usufruir dos serviços propostos por esta aplicação, assim como assumir custos de posse e de instalações.

A aplicação funciona como **serviço** e não como lançamento de um centro clínico, apenas são disponibilizadas as ferramentas para o cliente lidar com situações de adoção, pedidos, registo de pedidos e sinalizações de emergências que trarão, eventualmente, uma maior taxa de uso às instalações da empresa.

Como preço, o grupo prevê que seja uma percentagem do pagamento mensal baseado nos custos de manutenção mensais do serviço, assim como um valor extra a decidir na altura do contrato, para conseguir pagar aos empregados.

Assim, consegue-se abstrair à empresa todo este funcionamento por apenas um custo mensal obtido na altura do contrato, facilitando a gestão e atraindo mais clientela no futuro.

PROPOSTA

Antes de começarmos a programação em si, foi feita uma proposta do que seria o resultado final da aplicação. Nesta secção irá ser abordado o que foi pensado inicialmente, em termos de objetivos e requerimentos, organização temporal do projeto, mockups e até diagramas.

De notar que o projeto proposto aqui não corresponderá totalmente ao trabalho efetivamente realizada, pois no desenvolvimento da aplicação foram feitas alterações ao conceito inicial.

3.1 OBJETIVOS

Desenvolver um sistema de acolhimento de animais vadios/abandonados e de socorro que providencie os seguintes serviços:

- deslocação de especialistas ao local onde se encontram os animais, de possível emergência;
- transporte do animal para o centro e/ou para uma clínica veterinária;
- alojamento dos animais;
- agendamento de visitas ao centro;
- listagem dos animais no centro (nome, raça, ficha médica, etc), que ficam disponíveis para adoção;
- listagem de voluntários do centro (pessoas que podem sinalizar os animais na app e adotar);
- histórico de sinalizações e adoções;
- sinalização que permite anexar informação sobre o animal (descrição física, comportamental do animal, etc);
- listagem dos empregados do centro e dos veículos (para controlar se existem veículos livres, etc);

- sistema de sinalização com recurso a coordenadas (API externa – Google Maps);
- notificação do cliente sobre o resultado do resgate;

3.2 REQUERIMENTOS

Funcionais na app:

- sinalização de um animal, indicando coordenadas GPS, uma descrição do estado do animal e anexar fotografias do mesmo;
- marcação uma visita ao centro;
- consulta da lista de animais no centro;
- consulta / alteração dos dados pessoais (username, pass etc) e consulta das estatísticas (n resgates bem-sucedidos sinalizados, n sinalizações, n adoções, etc);
- opções de login, logout e registar;

Funcionais no backoffice:

- criação de fichas técnicas com as informações dos animais (nome, raça, marca ração, etc);
- criação de fichas técnicas de empregados (dados pessoais, profissão, etc);
- criação de fichas técnicas de adoções (dados sobre o animal, dados sobre o seu adotante, data de adoção, etc);
- consulta e alteração das listas de animais, empregados e adoções;
- adoção de um animal;

3.3 CRONOGRAMA

Numa fase inicial, o grupo decidiu dividir tarefas entre todos os seus elementos, garantindo o paralelismo entre as várias etapas do projeto. Para tal, foi utilizado o *website* **TeamGantt** para formar um diagrama de *Gantt*, que engloba todas as fases do projeto (num ponto de vista mais superficial) e, portanto, garante ao grupo um maior controlo de fluxo de atenção faseada.

Assim, foi possível denominar elementos para estabelecer contacto com as diversas tarefas que eram totalmente independentes, como o *back-end* e *front-end*, por exemplo.

Na página seguinte encontra-se o diagrama final utilizado nas primeiras semanas de desenvolvimento do projeto.

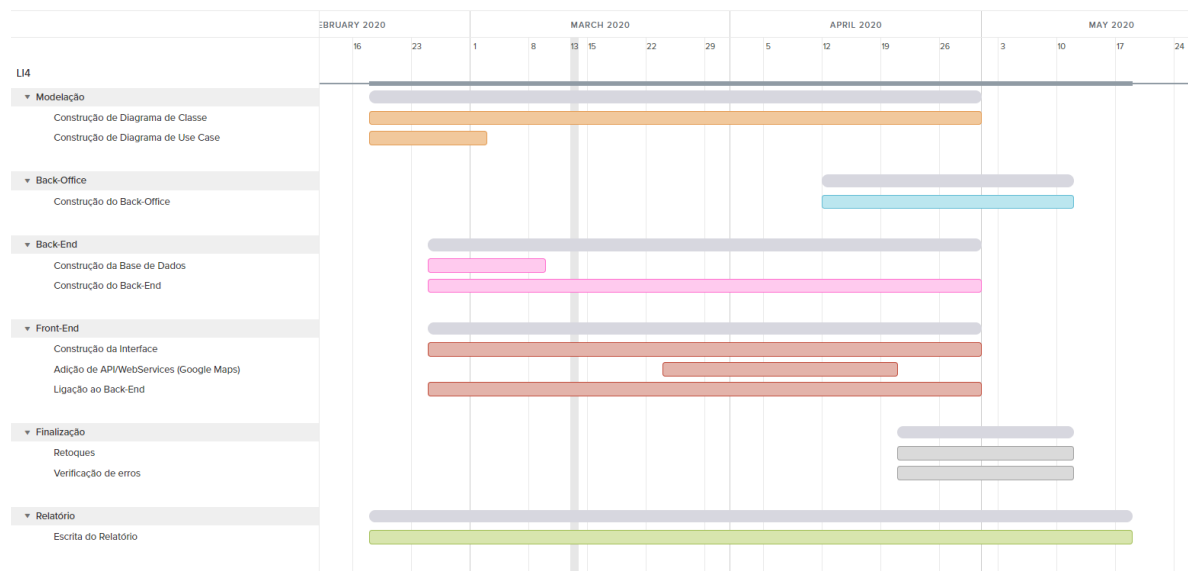


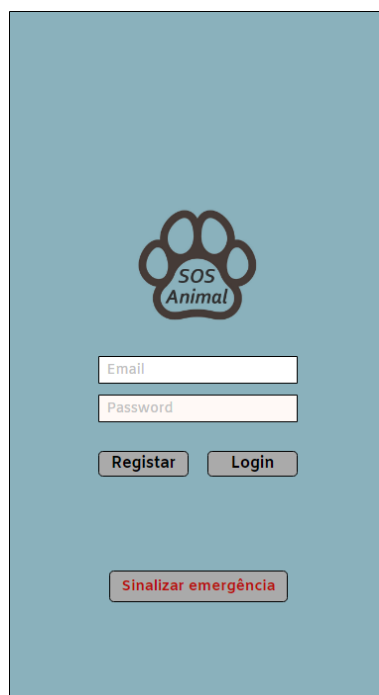
Figura 1: Cronograma do Projeto

3.4 MOCKUPS



The registration page features a light blue background. At the top center is the SOS Animal logo, which consists of a brown paw print with the text "SOS Animal" inside. Below the logo are four input fields stacked vertically: "Nome", "Email", "Password", and "Password Novamente". Each field has a thin border. Below the input fields is a single button labeled "Registrar" with a brown background and white text.

Figura 2: Página de Registro



The login page features a light blue background. At the top center is the SOS Animal logo, which consists of a brown paw print with the text "SOS Animal" inside. Below the logo are two input fields stacked vertically: "Email" and "Password". Each field has a thin border. Below the input fields are two buttons side-by-side: "Registrar" and "Login", both with brown backgrounds and white text. At the bottom center is a button labeled "Sinalizar emergência" with a brown background and white text.

Figura 3: Página de Log-in

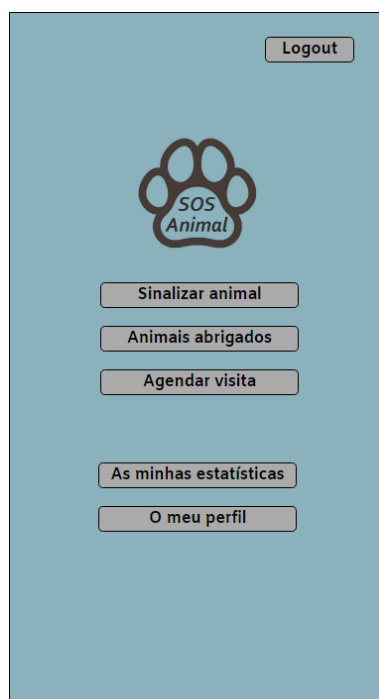


Figura 4: Menu de Entrada

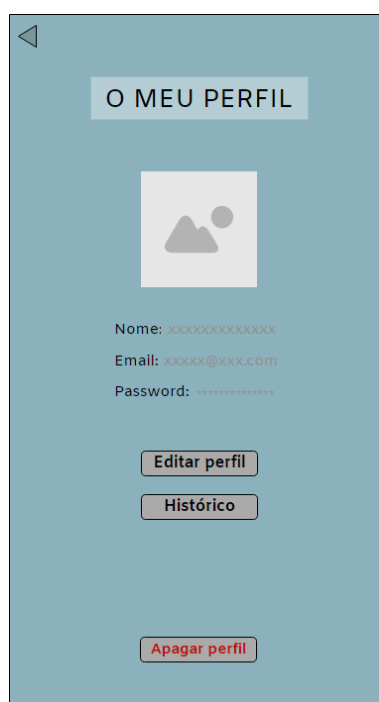


Figura 5: Página de Perfil

AGENDAR VISITA

Data da visita:

Hora 01/01/2021

Motivo da visita:

insira texto aqui...

Confirmar

Figura 6: Agendar uma Visita

ANIMAIS ABRIGADOS

Nome: xxxxxx
Raça: xxxxxx
Idade: xxxxxx

Nome: xxxxxx
Raça: xxxxxx
Idade: xxxxxx

Nome: xxxxxx
Raça: xxxxxx
Idade: xxxxxx

Nome: xxxxxx
Raça: xxxxxx
Idade: xxxxxx

Figura 7: Lista de Animais Abrigados

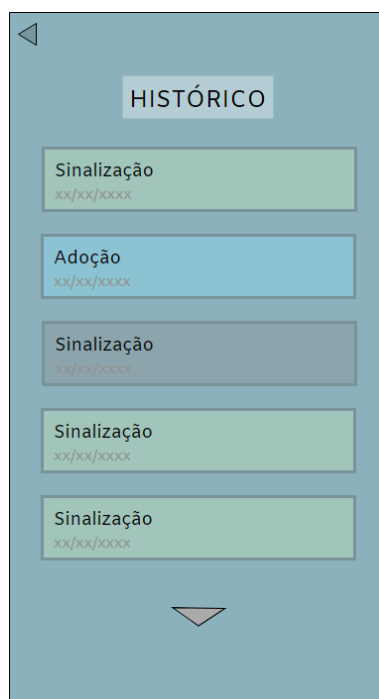


Figura 8: Histórico do Utilizador



Figura 9: Menu de Sinalização

3.5 DIAGRAMAS

Numa fase inicial do projeto, o grupo decidiu criar pequenos diagramas para tornar o desenvolvimento mais eficaz e, para além disso, garantir que todas as especificidades do projeto eram cobertas logo desde o início, para evitar redundâncias no projeto parcial e até mesmo no final.

Para tal foram utilizados os seguintes diagramas:

- Diagrama de Classe
- Diagrama de Use Cases

Foram limitados apenas os enunciados acima devido a serem desnecessários diagramas mais específicos, como se tratava duma linguagem nova, que não está relacionada com objetos, não se tiraria nenhum partido de utilizar diagramas mais complexas em UML.

De seguida estão anexados os diagramas utilizados numa fase inicial de desenvolvimento do projeto.

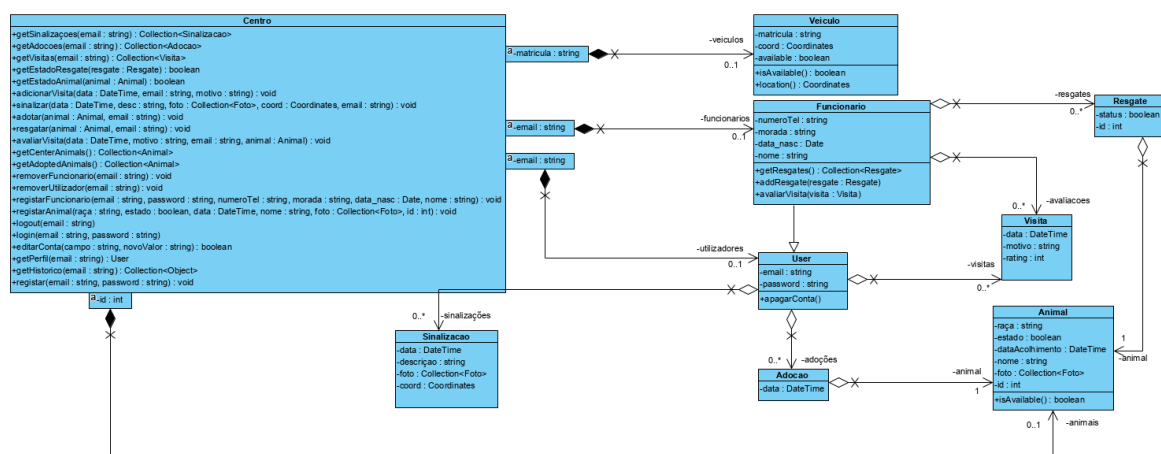


Figura 10: Diagrama de Classes do Projeto

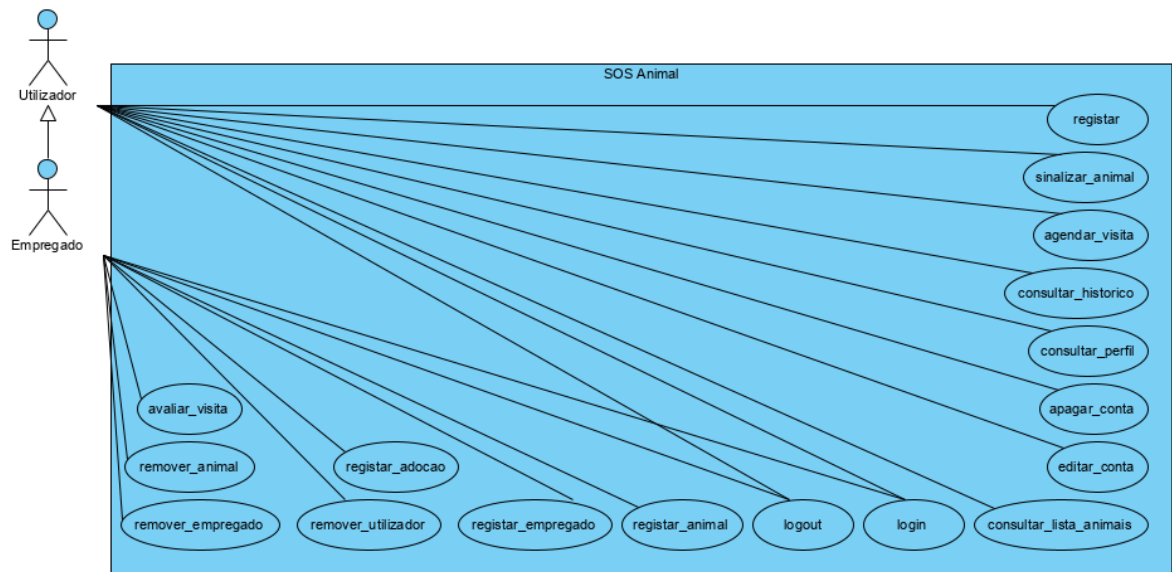


Figura 11: Diagrama de Use Cases do Projeto

DESENVOLVIMENTO

Nesta parte de desenvolvimento, vai-se entrar com mais detalhe sobre o projeto criado, começando por uma breve descrição sobre decisões tomadas, tanto em termo de tecnologias como nas divisões de trabalho e, de seguida, é descrita a implementação feita tanto da aplicação android (frontend, backend e funcionalidades) como do backoffice.

4.1 DECISÕES

Como foi explicado mais acima no relatório, a nossa aplicação “SOS Animal” consiste num serviço de urgência para animais. Tal como apresentado na proposta de projeto, desenvolvemos uma aplicação utilizando as seguintes tecnologias para cada parte do projeto:

- Backend: **SQL** para base de dados e **asp.net com C#** para a api
- Frontend: **Xamarin Forms** e **C# (MVVM)**
- BackOffice: **asp.net (MVC)**

Tanto a app como o backoffice estão alojados pelo **Azure**.

Além das decisões relativas às tecnologias usadas, o projeto também foi dividido entre os membros de modo a otimizar o desempenho: 3 membros focaram-se mais na backend e backoffice enquanto os restantes trabalharam mais na frontend, relatório e apresentações. Obviamente que mesmo assim, o grupo ajudou-se mutuamente, independentemente dessas divisões, quando surgiam problemas ou era necessário ajuda em alguma parte.

Por fim, durante o trabalho também foram feitas decisões no que respeita às funcionalidades a implementar. Por exemplo, decidiu-se abandonar a parte relativa aos veículos e concentrarmo-nos apenas nas funcionalidades diretamente relacionadas com o utilizador ou com os funcionários do centro.

4.2 IMPLEMENTAÇÃO

Este capítulo aborda todo o desenvolvimento da aplicação, abordando inicialmente a backend e frontend, de forma geral. Depois, todas as funcionalidades da aplicação Android são explicadas, seguidas da descrição da implementação do backoffice.

4.2.1 Backend

Para começarmos a lidar com a backend, e consequentemente a base de dados, criamos um serviço Azure, serviço tal que engloba estas duas componentes do projeto. A base de dados está alojada num servidor deles assim como o resto da backend.

Com a api criada, foi só necessário conectá-la à base de dados e, assim, a ponte entre a backend (e como consequência a base de dados) e a frontend ficou estabelecida.



Figura 12: Api

Com a tecnologia percebida, criamos e completamos as tabelas da base de dados. Como era de esperar, a nossa base de dados sofreu alterações à medida que era necessário guardar mais informação pertinente ou retirar parâmetros não utilizados. O resultado final é o mostrado na figura seguinte cujo esquema foi obtido no Microsoft SQL Server Management Studio:

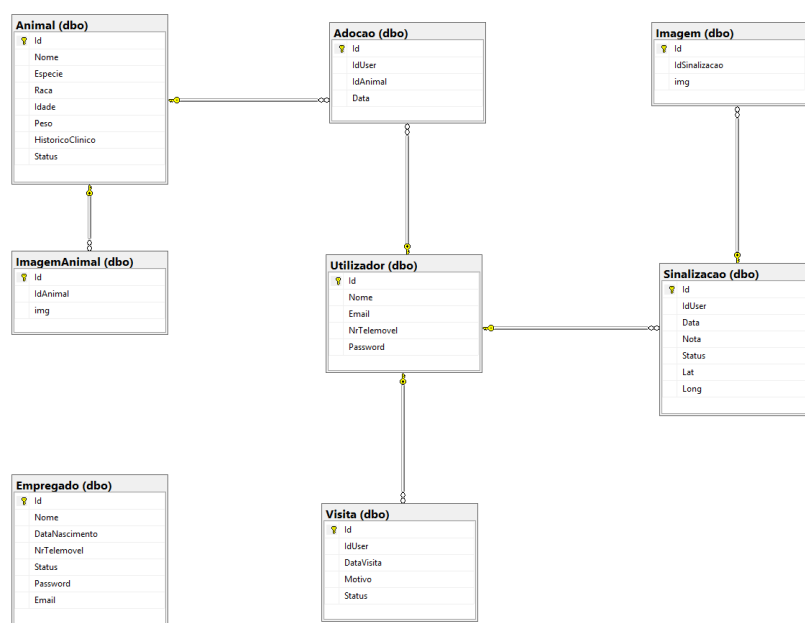


Figura 13: Esquema da base de dados

De seguida, inserimos este diagrama no nosso projeto de back-end, utilizando o Visual Studio, que cria os models correspondentes, utilizando a ferramenta entity framework, e os respetivos controllers (que irão conter todas as funções necessárias para aceder e/ou modificar a base de dados), na sua forma mais básica. Durante os resto do projeto, acrescentamos as funções necessárias no controller de modo a implementar as funcionalidades pretendidas. Na figura seguinte temos um exemplo de uma função no controller correspondente aos utilizadores:

```

[[ResponseType(typeof(Utilizador))]
[Route("api/Utilizador/Login/{email}/{password}/")]
[HttpGet]
0 references
public IActionResult Login(string email, string password)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    if (!UtilizadorExistsE(email))
    {
        return NotFound();
    }

    //string query = "SELECT * FROM Utilizador WHERE Email = @p0 and Password= @p1";
    //Utilizador ut = db.Utilizador.SqlQuery(query,email,password).Single();
    if (db.Utilizador.Count(e => e.Email.Equals(email) && e.Password.Equals(password)) > 0) {
        return StatusCode(HttpStatusCode.OK);
    }
    else
    {
        return StatusCode(HttpStatusCode.Conflict);
    }
}

```

Figura 14: Função Login

4.2.2 Frontend

Para a nossa frontend, guiamo-nos pelos mockups exemplificados acima. Para obter tal resultado, aplicamos diversas estruturas disponíveis pela plataforma que estamos a utilizar, **Xamarin.Forms**, de modo a alcançar os resultados pretendidos. Também é de notar que foi utilizada uma estrutura de **MVVM** (model-view-view-model).

Aqui segue-se uma imagem do ambiente de trabalho utilizado na frontend da aplicação:

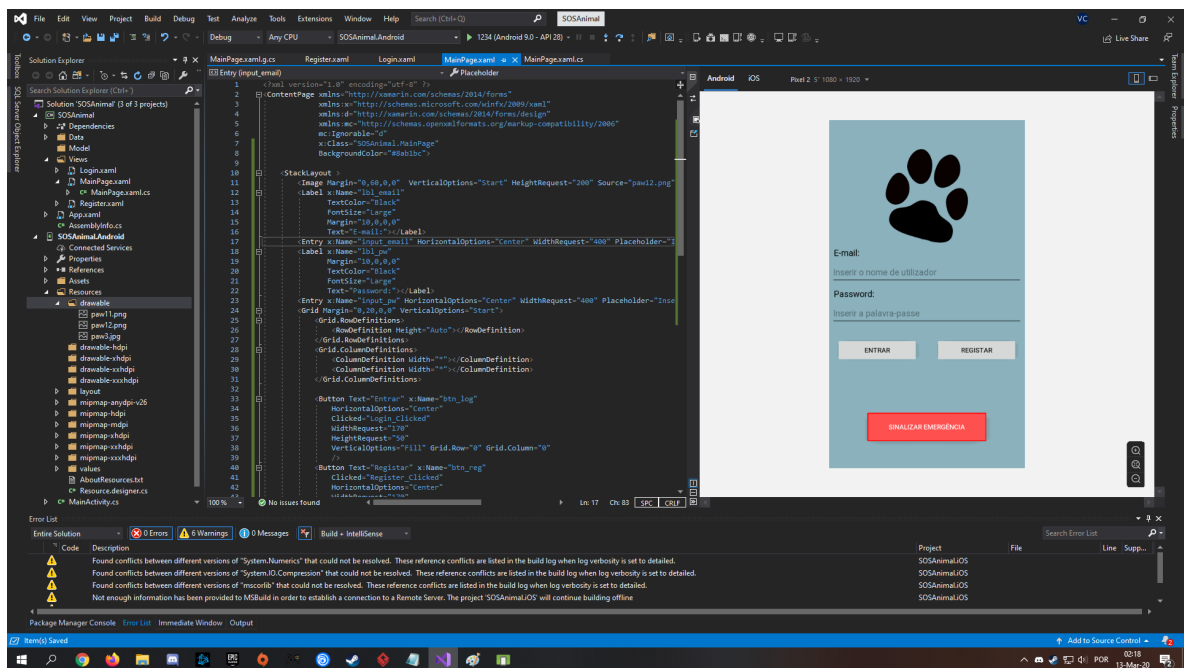


Figura 15: Função Login

4.2.3 Funcionalidades da Aplicação

Nesta subsecção, serão descritas todas as funcionalidades da aplicação android, de modo detalhado. Também são apresentados prints correspondentes a cada uma destas de modo a estabelecer uma ligação imediata com a aplicação em si.

Login

A figura seguinte (Figura 16) será o primeiro ecrã que o utilizador observa ao entrar na aplicação. Para ter acesso ao resto, é necessário inserir os dados corretos, isto é, uma combinação de email e password correta, previamente registada.

Caso os dados inseridos não correspondam a nenhuma entrada na base de dados, um aviso é mostrado ao utilizador (Figura 17) e, caso sejam dados corretos, o user é logado com sucesso e pode aceder ao resto das funcionalidades da aplicação, como podemos observar no menu representado na Figura 18.

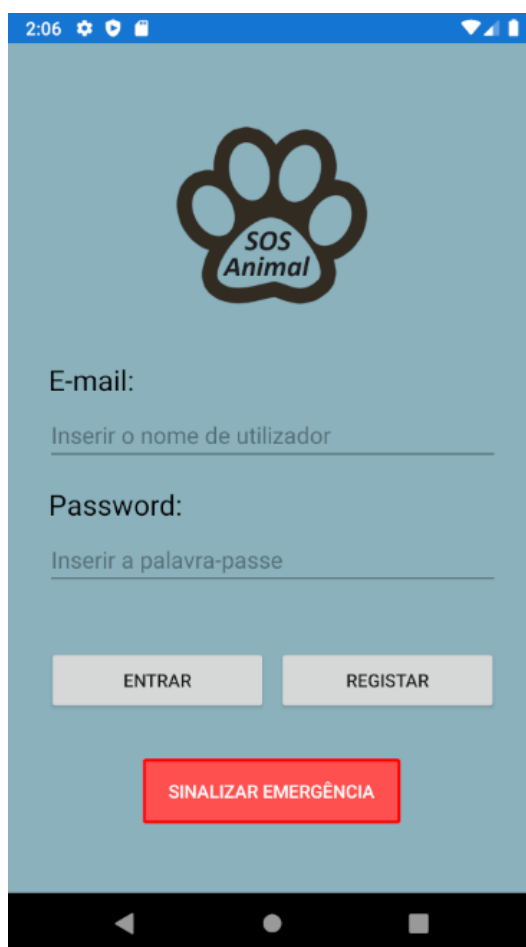


Figura 16: Menu de Login

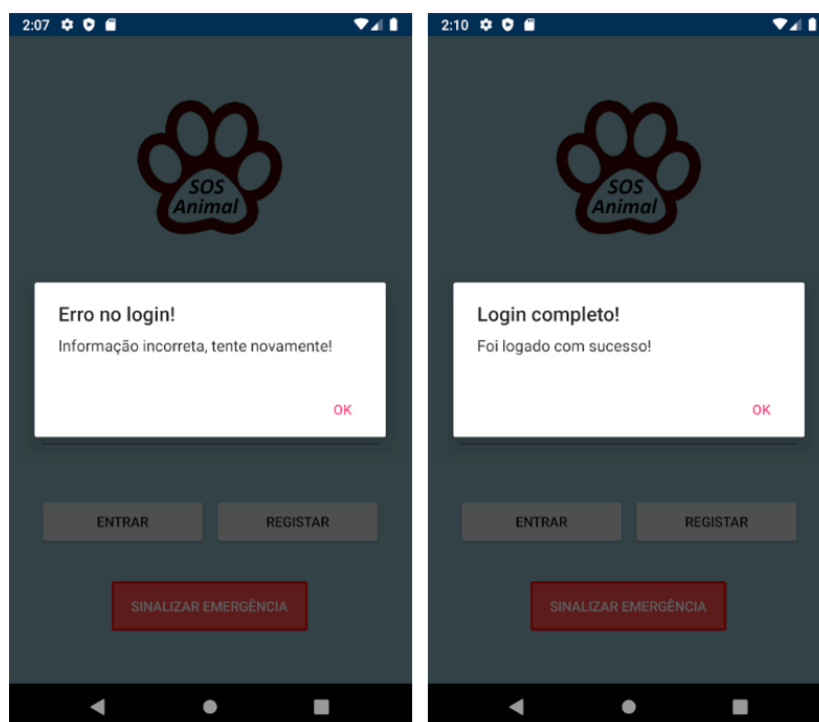


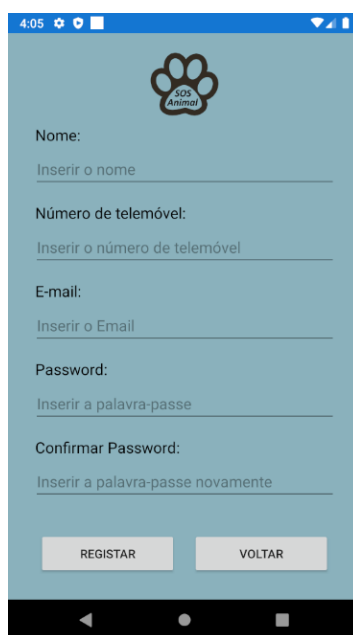
Figura 17: Avisos de Login



Figura 18: Menu Principal

Registar

Para fazer se registar, o user tem que criar uma conta na aplicação, fornecendo nome, número de telemóvel, e-mail e password (Figura 19). Caso já exista um utilizador na base de dados com o e-mail inserido, é mostrado um aviso indicando tal e, caso contrário, o utilizador é registado com sucesso (Figura 20).



The screenshot shows the registration form of the SOS Animal app. At the top, there is a blue header bar with the time 4:05 and status icons. Below the header is a light blue background with a paw print logo containing the text 'SOS Animal'. The form consists of several input fields with labels: 'Nome:' (placeholder: 'Inserir o nome'), 'Número de telemóvel:' (placeholder: 'Inserir o número de telemóvel'), 'E-mail:' (placeholder: 'Inserir o Email'), 'Password:' (placeholder: 'Inserir a palavra-passe'), and 'Confirmar Password:' (placeholder: 'Inserir a palavra-passe novamente'). At the bottom of the form are two buttons: 'REGISTAR' and 'VOLTAR'.

Figura 19: Menu de Registo

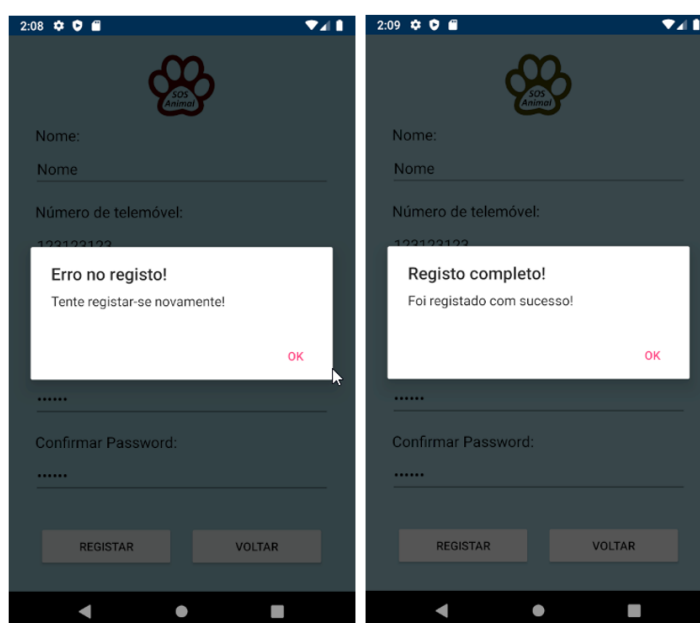


Figura 20: Avisos de Registo

Sinalizações

A funcionalidade em que se baseou o conceito da aplicação foi a de sinalizar animais em situações de emergência, tanto animais perdidos e/ou abandonados como domésticos. Para tal, o utilizador não precisa de estar registado no sistema, pode sinalizar anonimamente pelo menu de login inicial e, claro, também o pode fazer através do menu principal, uma vez logado.

Para sinalizar um animal tem que primeiro escrever uma breve descrição, se não o fizer um aviso será mostrado, como se observa da Figura 23. Para além disso, uma localização também é obrigatoriamente um dado pedido, ou outro aviso será apresentado. Para tal localização, é utilizada a API Google Maps para gravar as coordenadas no sítio atual e/ou do sítio onde o animal foi observado. Também é possível anexar uma fotografia do animal, apesar de não ser obrigatório.

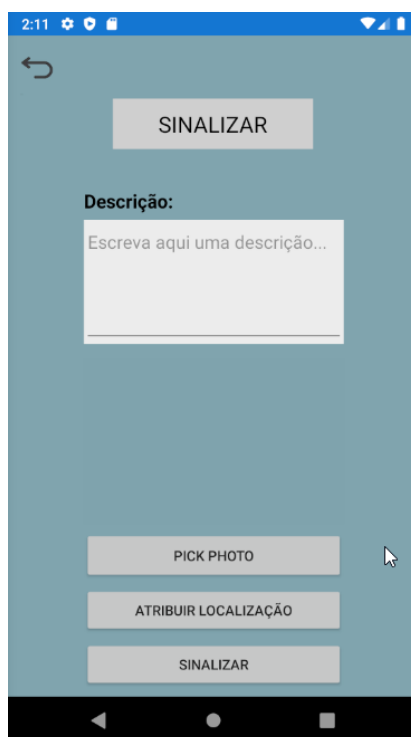


Figura 21: Página de Sinalizar

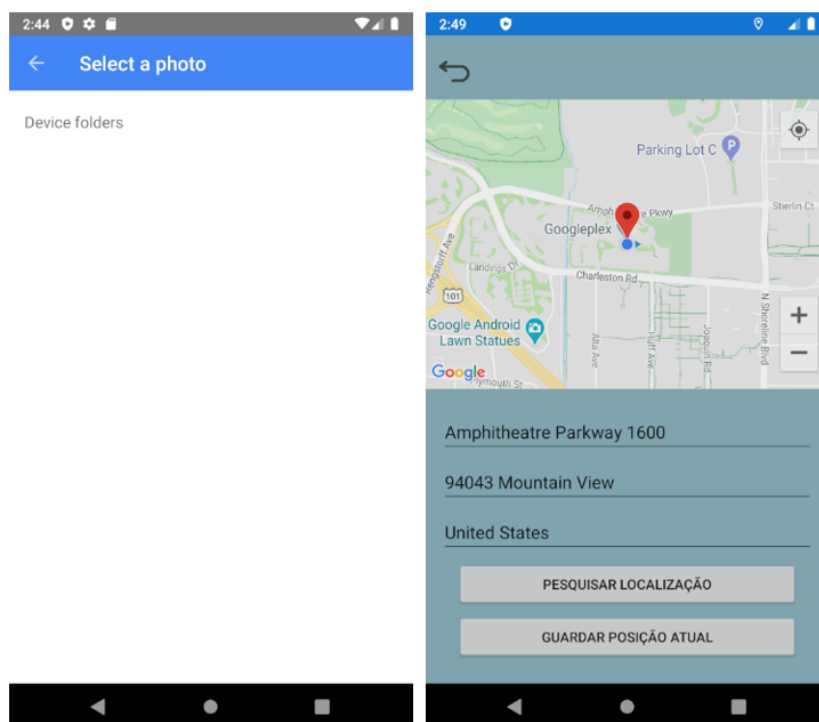


Figura 22: Páginas para inserir localização e foto na sinalização

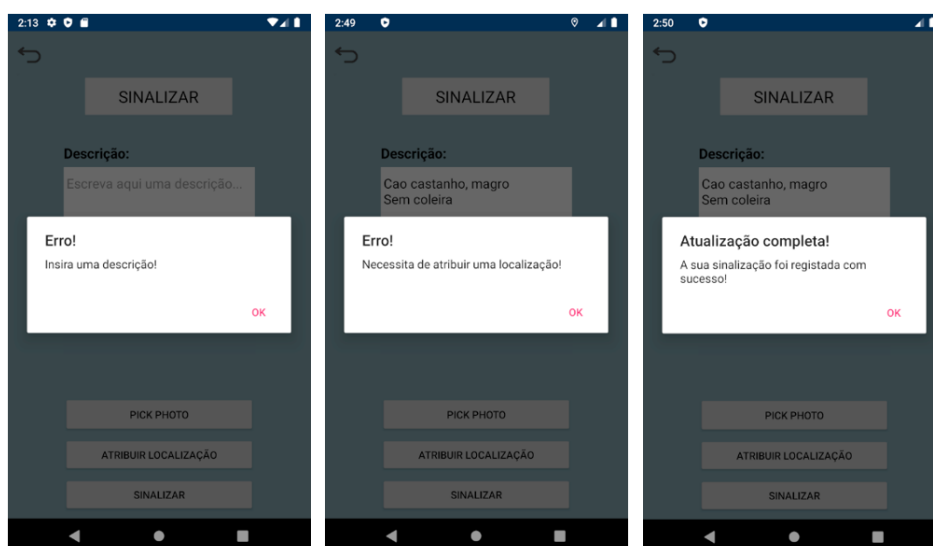


Figura 23: Avisos de Sinalização

Histórico de Sinalizações

No histórico de sinalizações, o utilizador consegue ver todas as sinalizações que efetuou e detalhes sobre cada uma delas. Quando se depara com a página principal (Figura 24), tem uma 'scroll list' em que consegue ver algumas das informações bem como o estado na sinalização pela cor: verde significa que o animal foi resgatado e cinzento o contrário.

Ao clicar em cada uma das sinalizações, é possível ver toda a informação relativa a essa sinalização: data, descrição, localização e fotografias, se existentes (Figura 25).



Figura 24: Histórico de sinalizações



Figura 25: Detalhe de uma sinalização

Perfil

Cada utilizador também consegue ver os detalhes sobre a sua conta clicando no botão "O Meu Perfil" no menu principal. Aí, irá ser direcionado para uma página (Figura 26) com o seu e-mail, username e password, embora censurada para a segurança do user em questão. A imagem é meramente simbólica, sendo que não vimos necessidade de implementar uma foto de perfil numa aplicação cujo objetivo é resgatar animais.

Na mesma página, o utilizador tem a opção de editar o seu username, bem como a password atual. Em ambas as situações, tem que confirmar com a password atual para confirmar a identidade de quem está a fazer tal alteração. Caso insira a password atual errada, aparece o aviso correspondente (Figuras 27 e 28) e caso contrário, as alterações são feitas e a base de dados é atualizada.

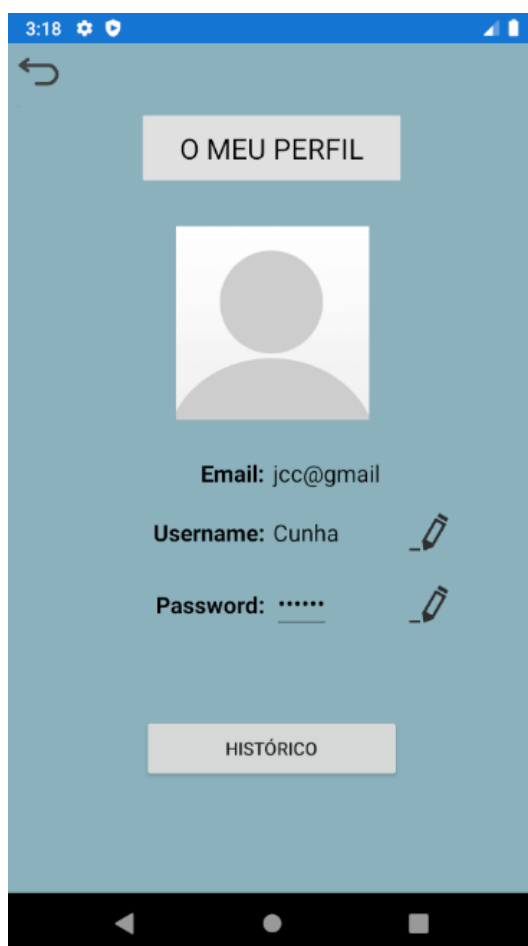


Figura 26: Perfil de um utilizador

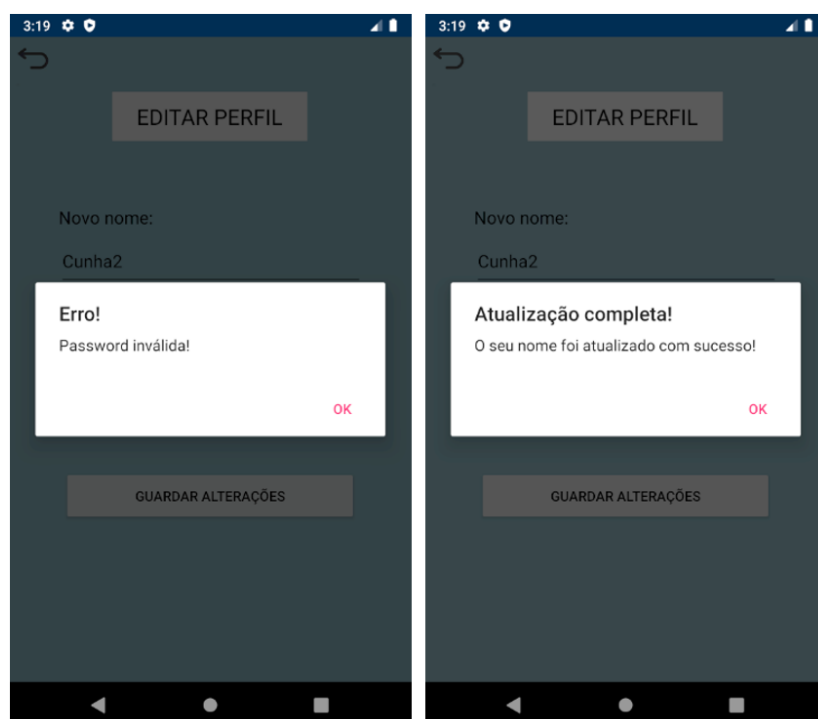


Figura 27: Avisos ao editar o nome

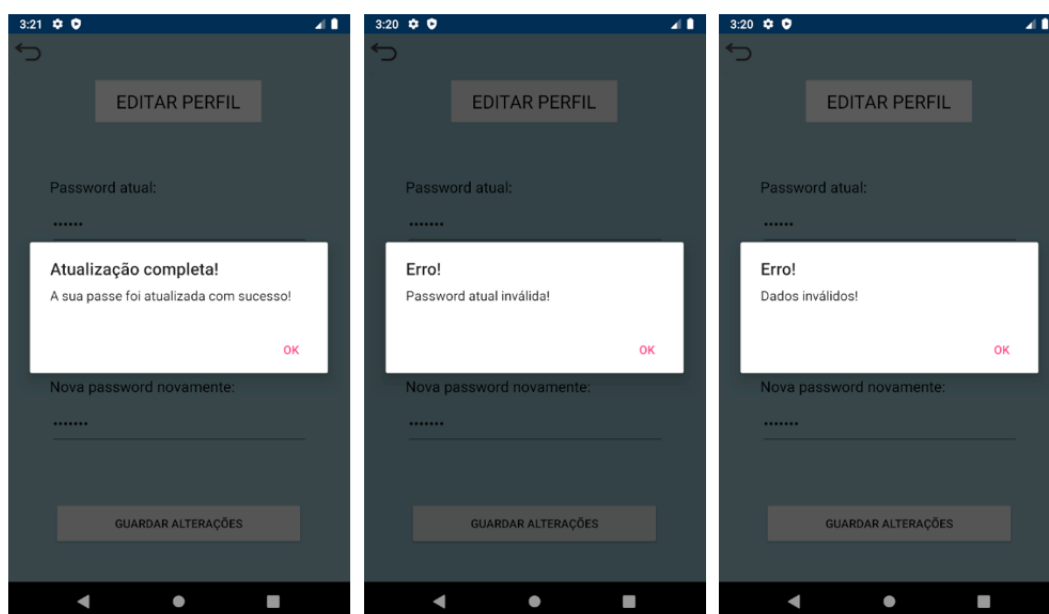


Figura 28: Avisos ao editar a password

Agendar visitas

Cada utilizador da aplicação também poderia (hipoteticamente) visitar os animais do centro. Para tal, tem de ir para o menu de agendar visita (Figura 29) e seleccionar o dia e hora da visita (Figura 30) e o motivo, sendo este último opcional.

Como é indicado, há restrições nas horas a visitar. Caso o utilizador selecione horas inválidas, tal aviso erro é indicado (Figura 31) enquanto a data já é restrita desde o dia presente até 1 mês à frente automaticamente.

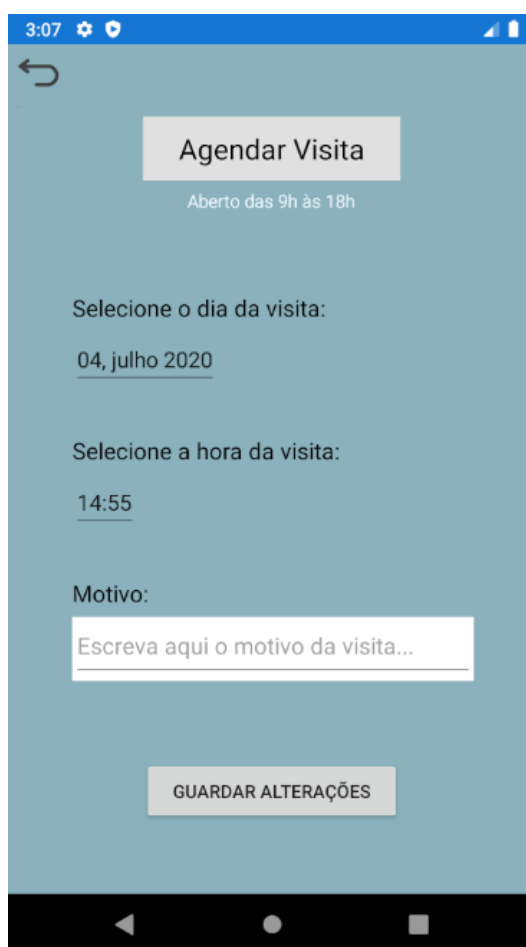


Figura 29: Página para agendar visita

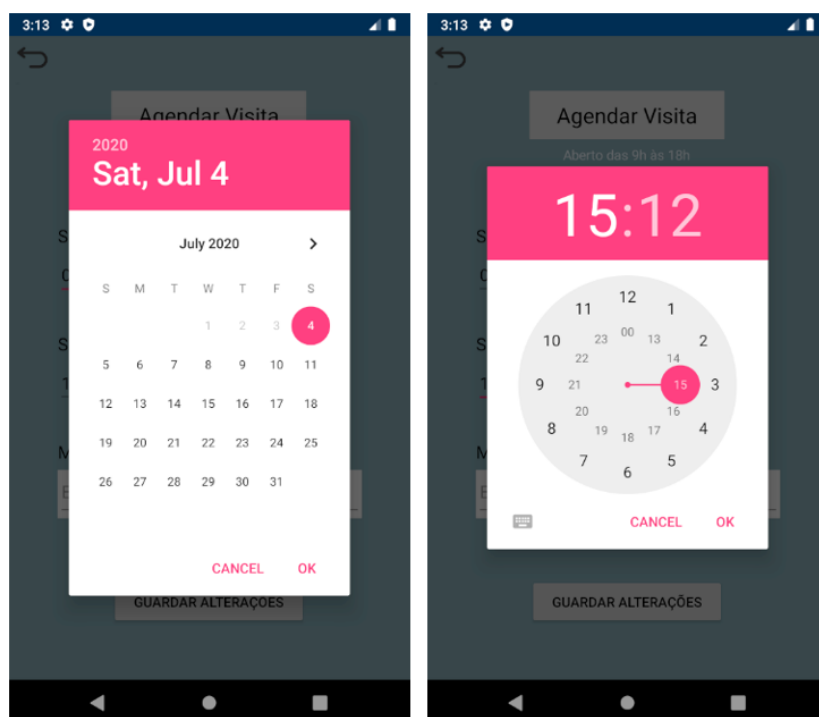


Figura 30: Páginas para inserir data e hora da visita

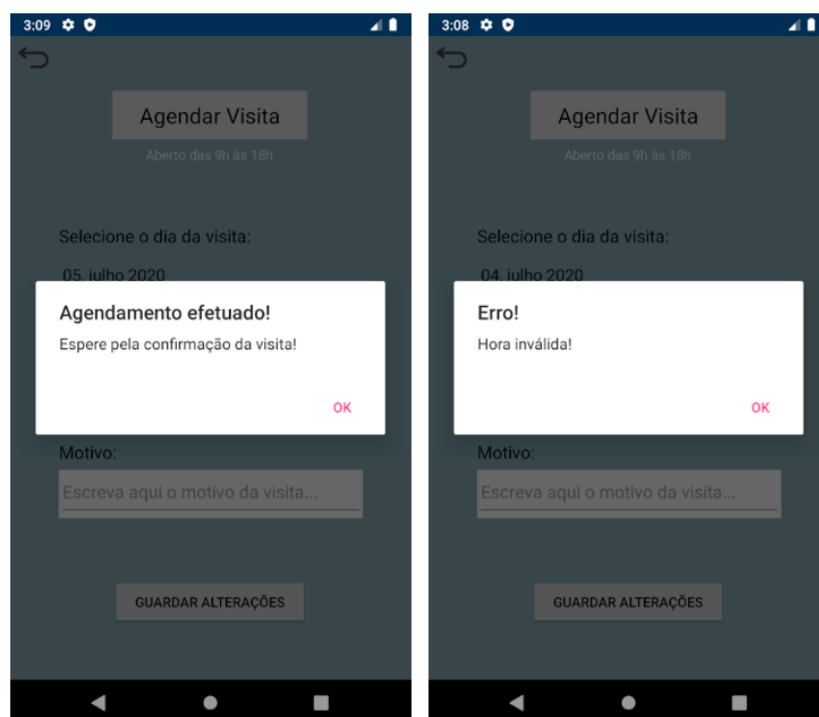


Figura 31: Avisos no agendamento de uma visita

Histórico de Visitas

O histórico de visitas funciona de maneira semelhante ao das sinalizações: é uma lista das visitas solicitadas pelo user (Figura 32) e, ao clicar em cada, é possível ver mais detalhes sobre cada um desses pedidos. As cores significam o estado de aprovação por parte dos funcionários do estabelecimento: verde quer dizer que foi aprovada, vermelho o contrário e amarelo que ainda está pendente.

Ao clicar em cada uma das visitas é possível ver os seus detalhes (Figura 33), como a data, o motivo e o estado.

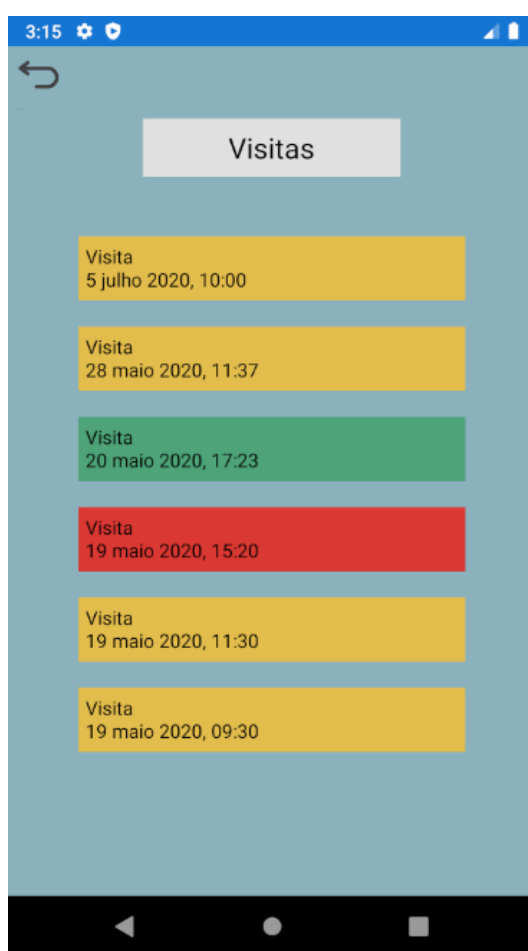


Figura 32: Página das visitas

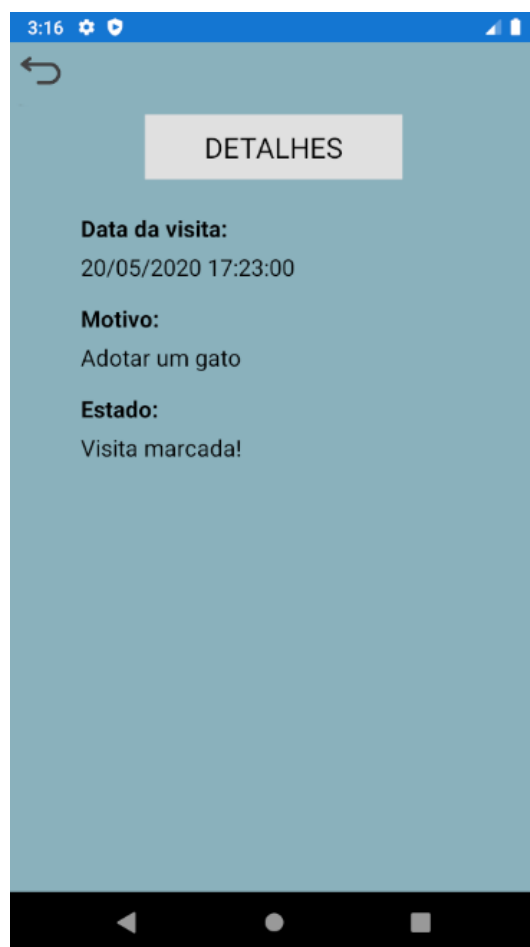


Figura 33: Detalhes de uma visita

Animais abrigados

De modo aos utilizadores conseguirem seguir o estado dos animais que sinalizaram ou até conseguirem ver animais para potenciais adoções, criou-se uma página em que se consegue ver todos os animais abrigados no estabelecimento (Figura 34). Na lista, as cores significam o estado do animal sendo que verde significa que já está em boas condições, no centro e amarelo indica que ainda está num estabelecimento exterior, como um hospital animal ou veterinário, a ser tratado.

Ao clicar em cada um, é possível ver os detalhes sobre o animal (Figura 35), como o nome, espécie, raça, peso, estado, histórico e até fotografias dele.



Figura 34: Página dos animais abrigados

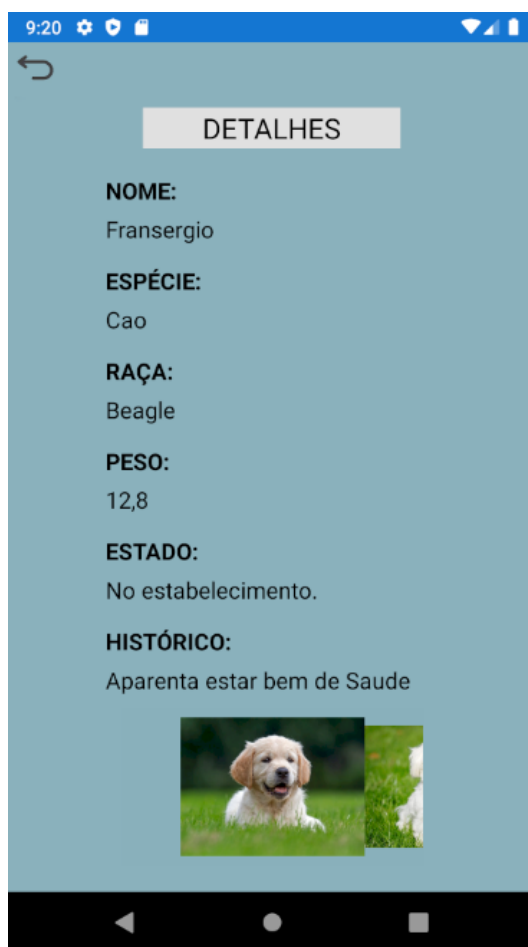


Figura 35: Detalhes de um animal abrigado

4.2.4 BackOffice

O *backoffice* é a ferramenta utilizada pela administração do serviço para uma gestão e monitorização do espaço desenvolvido na aplicação final.

É o local onde se efetuam mudanças nas bases de dados, como editar, eliminar e adicionar elementos manualmente, assim como verificar informação pertinente em tempo real sobre o sistema.

Para este projeto em concreto, o grupo decidiu utilizar um *website* visto que era a opção que faz mais sentido, uma vez que pode ser acedido em qualquer dispositivo desde que tenha acesso à Internet e permite uma maior flexibilidade de implementação, uma vez que toda a parte gráfica é implementada manualmente, assim como as funcionalidades associadas a cada uma das propriedades descritas.

Implementação

O grupo decidiu utilizar a ferramenta de *Web Development* do C#, que permite integração direta com os serviços *Azure* utilizados para o restante projeto. Assim, utiliza-se a mesma API externa para todos os serviços da aplicação, garantindo a consistência multiplataforma que se pretende desde o início, isto é, todo o *backend* criado anteriormente é utilizado neste *website* para alterações nas bases de dados.

O modelo escolhido para o desenvolvimento foi o **ASP.Net MVC**, que se baseia em código C# para implementação de comportamentos em páginas HTML, através da API realizada no *backend*.

Para cada entidade na base de dados, será disponibilizado ao administrador uma lista com todo o conteúdo relativo a ela, associando cinco ações para cada uma:

- **Visualizar:** modo por defeito, mostra todos os elementos da tabela ordenados por um ordem arbitrária, determinada pelos campos da entidade. Por exemplo, no utilizador poderia ser ordenado pelo seu nome, assim como o seu ID único, de forma crescente ou decrescente.
- **Editar:** Edita um dos campos da entidade e submete as alterações na base de dados.
- **Detalhes:** Mostra mais detalhes da entidade (campos) ao administrador.
- **Remover:** Remove a entidade da base de dados permanentemente.
- **Adicionar:** adiciona uma nova entrada daquela entidade à base de dados.

Utilizadores Adicionar utilizador

Show 10 entries Search:

Id	Nome	Email	Password	NrTelemóvel	Action
1	Cunha2	jcc@gmail	pwnova2	123123123	Editar Detalhes Remover
2	Guest	guest@gmail.com	guestpw	123456789	Editar Detalhes Remover
3	quim	jcc3@gmail.com	pw	2345345	Editar Detalhes Remover
4	zeca	jcc4@gmail.com	pw	2344643	Editar Detalhes Remover
5	Hugo	Hugo@gmail	pw	67563	Editar Detalhes Remover
6	Filipa	Filipa@gmail	pw	365357	Editar Detalhes Remover
7	Ruca	Ruca@gmail	pw	442432	Editar Detalhes Remover
11	Gestrudes	Gestrudes@gmail.com	pinguim	934511231	Editar Detalhes Remover
12	luisinho	luisinho@gmail.com	pingu	934523232	Editar Detalhes Remover
13	Dillaz	Dillaz@gmail.com	pingo	934233211	Editar Detalhes Remover
Id	Nome	Email	Password	NrTelemóvel	Action

Showing 1 to 10 of 23 entries Previous 1 2 3 Next

Figura 36: Tabela de Utilizadores

Para a tabela anterior, o código *cshtml* respetivo é:

```

1 @model IEnumerable<BackofficeSOSA.Models.Utilizador>
2
3 <p>
4     @Html.ActionLink("Create New", "Create")
5 </p>
6 <table class="table">
7     <tr>
8         <th>
9             @Html.DisplayNameFor(model => model.Nome)
10        </th>
11        <th>
12            @Html.DisplayNameFor(model => model.Email)
13        </th>
14        <th>
15            @Html.DisplayNameFor(model => model.Password)
16        </th>
17        <th>
18            @Html.DisplayNameFor(model => model.NrTelemovel)
19        </th>
20        <th></th>
21    </tr>
22
23    @foreach (var item in Model) {
24        <tr>
25            <td>
26                @Html.DisplayFor(modelItem => item.Nome)
27            </td>

```

```

29     <td>
        @Html.DisplayFor(modelItem => item.Email)
    </td>
31    <td>
        @Html.DisplayFor(modelItem => item.Password)
    </td>
33    <td>
        @Html.DisplayFor(modelItem => item.NrTelemovel)
    </td>
35    <td>
        @Html.ActionLink("Edit", "Edit", new { id=item.Id }) |
37        @Html.ActionLink("Details", "Details", new { id=item.Id }) |
39        @Html.ActionLink("Delete", "Delete", new { id=item.Id })
41    </td>
    </tr>
43 }
</table>

```

Para as restantes tabelas o processo foi igual, mudando apenas os campos respetivos, garantindo assim uma interface gráfica muito eficiente e que funciona corretamente para todo o processo de monitorização e alterações da base de dados.

Resultado Final

No final, com todo o HTML produzido para garantir a ligação entre os vários elementos, o resultado da página de login é o seguinte:

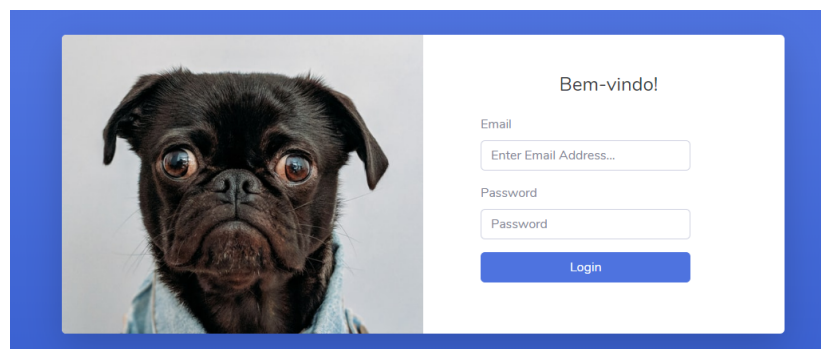
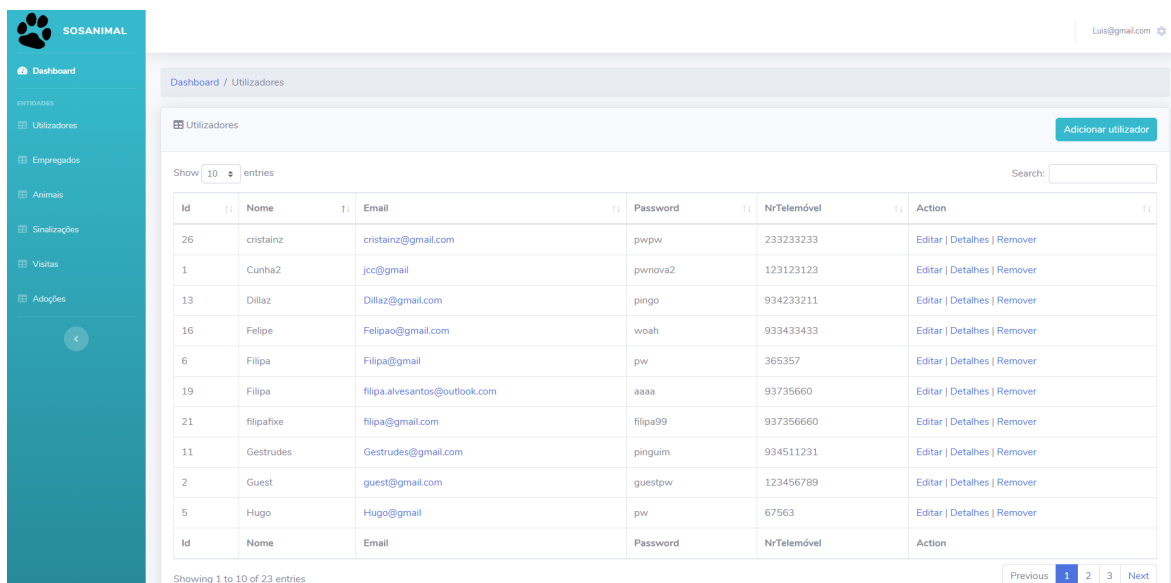


Figura 37: Secção de login do website

Que, após login, tem do lado esquerdo todas as ferramentas necessárias para os processos de manutenção da base de dados para as diversas entidades, isto é, os **Utilizadores**, **Empregados**, **Animais**, **Sinalizações**, **Visitas** e **Adoções**, como é possível ver na imagem seguinte.



SOSANIMAL

Dashboard / Utilizadores

Adicionar utilizador

Show 10 entries

Search:

Id	Nome	Email	Password	NrTelemóvel	Action
26	cristainz	cristainz@gmail.com	pwpw	233233233	Editar Detalhes Remover
1	Cunha2	jcc@gmail	pwnova2	123123123	Editar Detalhes Remover
13	Dillaz	Dillaz@gmail.com	pingo	934233211	Editar Detalhes Remover
16	Felipe	Felipao@gmail.com	woah	933433433	Editar Detalhes Remover
6	Filipa	Filipa@gmail	pw	365357	Editar Detalhes Remover
19	Filipa	filipa.alvesantos@outlook.com	aaaa	93735660	Editar Detalhes Remover
21	filipafixe	filipa@gmail.com	filipa99	937356660	Editar Detalhes Remover
11	Gestrudes	Gestrudes@gmail.com	pinguim	934511231	Editar Detalhes Remover
2	Guest	guest@gmail.com	guestpw	123456789	Editar Detalhes Remover
5	Hugo	Hugo@gmail	pw	67563	Editar Detalhes Remover
Id	Nome	Email	Password	NrTelemóvel	Action

Showing 1 to 10 of 23 entries

Previous 1 2 3 Next

Figura 38: Panorama Geral do Website

CONCLUSÃO

5.1 CONCLUSÕES

No final, o grupo achou que conseguiu um bom trabalho face aos desafios colocados, uma vez que tudo o que era obrigatório implementar foi implementado com sucesso, sendo ainda desenvolvidas mais capacidades ao programa ao longo do semestre.

Para além disso, em nota extra, o grupo determinou que utilizar *Xamarin Forms* garantiu uma enorme facilidade de aprendizagem de desenvolvimento *Mobile*, devido à sua implementação nativa para *Android*, *iOS* e *Windows*, apesar de nos focarmos apenas na primeira.

Quanto ao *back-office*, o grupo considerou que utilizar *HTML* juntamente com *C#* contribuiu para uma elevada taxa de sucesso, devido ao suporte imediato permitido pela Microsoft para aplicações Web, que ajudou não só a tornar um trabalho apelativo visualmente como funcional a nível técnico, assim como eficiente.

A fim a nota de agradecimento, o grupo gostaria de comentar que as apresentações quinzenárias ajudaram a obter um projeto final coeso, graças ao apoio constante dos docentes que se disponibilizam a comentar apresentações e a dar opiniões acerca do que melhorar, o que tornou o projeto bastante dinâmico e interessante.

5.2 TRABALHO FUTURO

Como trabalho futuro, o principal ponto de foco será a aplicação principal. Esta pode ser melhorada com mais algumas funcionalidades que permitam melhor qualidade de utilização ao consumidor, assim como lhe dar mais opções. Por exemplo:

1. Customização mais aprofundada do perfil;
2. *Blacklists* para pessoas que utilizem o sistema com propósitos avessos;
3. Autenticação por SMS (verificação de dois passos);
4. Disponibilizar a aplicação na *Play Store* aprumada e pronta a consumo público;

5. Modificar o design global da aplicação para algo mais eficiente, em termos energéticos, assim como mais apelativo, em termos visuais;
6. Análise estatística (por zona, por pessoa, etc) mais aprofundada.

Todos os pontos enumerados são basicamente funcionalidades da aplicação já desenvolvidos mas que podem ser aprofundados com maior detalhe, isto é, com mais um bocado de tempo de desenvolvimento.

