Projeto temático em Desenvolvimento de Aplicações
Relatório

Grupo 5
António Bento (97737)
Diogo Matos (98017)
Ivan Xavier (92441)
Maykol Santos (74079)
Ricardo Fernandes (49880)
Simão Julião (98045)

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

No âmbito do Projeto Temático em Desenvolvimento de Aplicações o Grupo 5 propôs-se desenvolver uma aplicação de gestão de uma clínica dentária, utilizando os conhecimentos adquiridos nas unidades curriculares de Engenharia de Software e Sistemas de Base de Dados incluídas no plano curricular da licenciatura em Tecnologias de Informação, ministrada na Escola Superior de Tecnologia e Gestão de Águeda – Universidade de Aveiro.

Os requisitos a cumprir no final do projeto foram identificados e desenvolvidos no decorrer de cinco reuniões entre a equipa e o representante do cliente, onde foi debatido a ideia geral do sistema.

Nestas reuniões foram abordados diversos temas que definem o funcionamento do projeto e irão ser a estrutura deste relatório nos capítulos de levantamento e desenvolvimento de requisitos do projeto.

Inicialmente, foi discutida a ideia geral do projeto onde surgiu a ideia do *software* para gestão de uma clínica, que mais tarde foi especializada para aplicações de medicina dentária. Após escolhido o tema principal do projeto foi escolhida a linguagem de programação que define o rumo do projeto. Esta originalmente estaria pré-definida por orientações dos Professores, sendo Java a língua escolhida por estar inserida no contexto académico do curso de Tecnologias de Informação, contudo foi concedida liberdade criativa ao grupo de escolher uma alternativa que satisfaça as condições de ser uma língua orientada a objetos.

Estando as formalidades iniciais completas, foram definidas as funcionalidades que o *software* deve conter e o que seria útil mas não prioritário. Dentro destas funcionalidades, foram levantados problemas naturais de restrições de acesso a funcionalidades ou a informações de doentes, estas têm de ser implementadas no código de forma a este estar em conformação com requisitos legais. Além destas, salientou-se a necessidade de manter a integridade da base de dados que foi ser implementada.

Dentro da parte mais técnica do pré desenvolvimento do projeto foi desenvolvido, durante várias reuniões com o orientador, os Diagramas de Atividades, os Diagramas de Classes e a Estrutura de Dados que juntos formam o esqueleto do projeto e o guia principal para orientador o desenvolvimento do código. Dado que este projeto consiste numa ferramenta de gestão para uma clínica, o sistema de *Login* é então um dos focos principais a desenvolver já que define a diferenciação de classes a ser implementadas e a forma como interagem entre si, assim como define a estrutura de como o interface irá ser desenvolvido.

Estes temas aqui referidos serão mais desenvolvidos no decorrer do relatório com capítulos inteiramente dedicados aos tópicos mais cruciais.

1.1 Visão geral do sistema

O projeto consiste na criação de *software* para gestão de uma clínica dentária. Este consiste numa interface desenvolvida para as necessidades dos diferentes cargos de funcionários da em-

presa ao mesmo tempo que é estabelecida uma comunicação com a base de dados, onde todos os dados serão guardados. Com o objetivo de desenvolver a mesma, foram levantados requisitos e necessidades que mais tarde definiram funcionalidades e estruturas do sistema.

O sistema irá registar a informação necessária dos profissionais da clínica e dos utentes. Vai também guardar a informação sobre consultas, possíveis exames e poderá imprimir receitas, relatórios de consulta e justificações. As funcionalidades da aplicação estarão divididas por profissão, sendo que existirão diferentes tipos de permissões para cada tipo de utilizador.

1.2 Cliente

Este projeto consta com a presença de um cliente hipotético sendo este um representante de uma clínica de medicina dentaria. Este tem o cargo de informar a equipa de desenvolvimento sobre os requisitos e funcionalidades necessárias a implementar no sistema, requisitos de segurança e de conformidade legal com proteção de dados sensíveis como também dos prazos de entrega e formalidades do mesmo. Foi assim possível definir objetivos concretos e realistas inerentes ao projeto a serem desenvolvidos durante o decorrer deste projeto.

1.3 Objetivos

Tendo em conta que a clínica dentária tem a necessidade de gerir e controlar todos os seus clientes e profissionais, bem como todos os dados relacionados com os mesmos (consultas, exames, ficha de cliente, ficha de profissional, entre outros) existe a necessidade de armazenar e gerir os dados necessários para permitir o correto funcionamento da clínica dentária. Considerando o que foi referido, é necessário criar uma aplicação capaz de responder aos requisitos previamente indicados, a equipa de desenvolvimento tem como objetivo fornecer coesão e estrutura para que seja possível desenvolver a aplicação em questão, de um modo eficaz e fluído.

O cliente pretende que o sistema apresente uma alternativa moderna e competente à solução anterior, para que tal se realize, a solução apresentada pela equipa de desenvolvimento tem a obrigação de ser apta para suportar todas as funcionalidades indispensáveis à clínica.

Com esta alteração de sistema existem benefícios não só de desempenho como de utilidade, no sentido em que as operações passam a ser instantâneas e é possível aceder à informação que é constantemente atualizada e validada, este é o mesmo conceito que possibilita a uniformidade de sistema (prevenção de falhas na integridade), permite uma validação e verificação ativa de dados aquando registados, permite desenvolvimento futuro, permite armazenamento de enormes quantidades de informação e retira todas as inconveniências de um sistema de registos tradicional.

1.4 Desafios de Crescimento Profissional

Por forma a desenvolver o nosso crescimento profissional fora das expectativas institucionalizadas e também de satisfazer conquistas pessoais, o grupo auto propôs-se ao desafio de usar a linguagem de programação Python em vez de Java como estava proposto na orientação da disciplina.

A linguagem Python tem vindo a crescer na sua adoção com o passar dos anos, tornando-se numa ferramenta versátil de programação a alto nível usada por várias instituições académicas e empresariais em diversas áreas, onde de momento se destacam os tópicos mais mediáticos como *machine learning*, cibersegurança[1] e aplicações científicas. A sua abordagem orientada para algoritmos permite desenvolvimento rápido de projetos que são fundamentais para prototipagem[2], o esforço das comunidades dedicadas resultaram em bibliotecas extensas de alta qualidade que permitem criação de código eficiente, capaz de ser mais rápido e consumindo menos memória que Java em determinadas aplicações[1].

Publicações recentes mostram que Python está a evoluir corretamente no que toca a mitigação de vulnerabilidades e desenvolvimento de funcionalidades, que permitem melhor segurança em aplicações online.[3].

Visto que Python não dispõe dos mesmos métodos de criação de interface que estão presentes num IDE de Java, como o JFrame, foi necessário escolher uma framework que tivesse compatibilidade com Python e que satisfaça as necessidades modernas de funcionamento com *desktop*, *mobile* e sistemas *embedded*. Tendo isto em consideração, foi escolhido a Qt *framework* que é de código livre e funciona com quaisquer sistema operativo moderno. Com Qt *framework* e o *software* de desenvolvimento de interface, **Qt Designer**, é possível desenvolver uma interface que funcione e se adapte para vários dispositivos diferentes com tamanhos de ecrã e resolução muito diferentes. Esta aplicação e as suas bibliotecas são compatíveis e compiladas por compiladores standard de C++. garantindo assim a sua compatibilidade com projetos que usem compiladores específicos de C++, e do lado do utilizador, um produto final de execução rápida e leve em recursos computacionais[4]. A interação entre as interfaces geradas e as bibliotecas de Qt com Python é feita com recurso à biblioteca **PyQt**.

Capítulo 2

Modelo de requisitos

2.1 Requisitos funcionais

Os requisitos funcionais do sistema foram debatidos em grupo de forma a definir a sua prioridade de implementação. Estes podem ser consultados na tabela seguinte:

Refª	**Requisito funcional**	**Prioridade**
RF.1	Guardar os dados dos utentes	Alta
RF.2	O sistema tem de registar profissionais	Alta
RF.3	Tem de existir um método de autenticação de funcionários	Alta
RF.4	Criação da ficha de cliente	Alta
RF.5	Criação do relatório médico	Alta
RF.6	Registar o estado do paciente no relatório	Alta
RF.7	Registar a prescrição do paciente no relatório	Alta
RF.8	Consultar a ficha do utente	Alta
RF.9	Consultar o relatório do utente	Alta
RF.10	Aceder a relatórios de consultas anteriores	Alta
RF.11	Alterar o relatório de utente	Alta
RF.12	Consultar prescrição médica	Alta
RF.13	Adição de exames ao relatório	Alta
RF.14	Administrador capaz de gerir e alterar todos os dados e configurações	Alta
RF.15	Agendar consultas	Alta
RF.16	Verificação de sobreposição de consultas	Média
RF.17	Facultar justificação médica	Baixa
RF.18	Possibilidade de suspender o processo de utente	Baixa
RF.19	Editar ficha de utente	Alta

2.2 Restrições e requisitos não funcionais

2.2.1 Requisitos de interface e facilidade de uso

Para garantir padrões de satisfação e facilidade de uso, foram tidos em conta os seguintes pontos:

Refª	**Requisito de interface e usabilidade**	**Req. funcionais relacionados**
RInt.1	O sistema deve ter uma interface simples e acessível	
RInt.2	As cores do sistema devem ter em conta daltonismo	
RInt.3	O tamanho dos objetos e das letras deve ser equilibrado	

2.2.2 Requisitos de desempenho

A equipa de desenvolvimento teve em consideração o desempenho e performance do sistema, com isto, foram definidas algumas restrições com o objetivo de garantir usabilidade.

Refª	**Requisito de desempenho**	**Req. funcionais relacionados**
RDes.1	As trocas de informação com a base de dados não devem exceder os 5000 milissegundos	
RDes.2	Existe a necessidade de armazenar valores introduzidos pelo utilizador aquando introduzidos	RF.19
RDes.3	Sincronização do sistema com o calendário e fuso horário local	
RDes.4	Uso de uma base de dados segura e capaz de registar todos os dados pretendidos	
RDes.5	Introdução de imagens de exames	RF.13

2.2.3 Requisitos de segurança e integridade dos dados

Atendendo às necessidades de cibersegurança e estabilidade da base de dados, foram estipuladas as seguintes normas:

Refª	**Requisito de segurança, privacidade e integridade de dados**	**Req. funcionais relacionados**
RSeg.1	A entrada no sistema deve ser autenticada	RF.3
RSeg.2	O utilizador só pode realizar operações que lhe são permitidas	RF.3
RSeg.3	Cada operação deve ficar registada	
RSeg.4	A base de dados deve manter os dados	RF.1
RSeg.5	A base de dados deve assegurar a integridade dos dados	RF.1

2.2.4 Requisitos de interface com sistemas externos e com ambientes de execução

Foram estipulados os seguintes requisitos funcionais relacionados com a compatibilidade de sistemas operativos, linguagem de comunicação com base de dados e linguagem de ligação entre interface visual e base de dados:

Refª	**Requisito de interface e usabilidade**	**Req. funcionais relacionados**
RIntE.1	O sistema pode ser executado em sistemas Windows e Linux	
RIntE.2	O sistema comunica com uma base de dados MySQL	RF.1
RIntE.3	O sistema vai ser interpretado e executado em Python	

2.2.5 Normas e regulamentação específicas aplicáveis

Foram tomadas em conta as seguintes normas regulamentais relacionadas com cibersegurança e rastreabilidade de dados:

Refª	**Requisito de interface e usabilidade**	**Req. funcionais relacionados**
RReg.1	Os dados emitidos devem ser mantidos para fins legais.	
RReg.2	Os dados emitidos pelos clientes devem ser atuais e autênticos.	
RReg.3	As palavras-chave devem estar encriptadas.	

2.2.6 Outros requisitos não funcionais

Para além dos requisitos estipulados acima, foram ainda definidos requisitos adicionais que complementam a integridade do programa, tais como:

Refª	Requisito de interface e usabilidade	Req. funcionais relacionados
RnF.1	Manutenção de erros inesperados do sistema.	

2.3 Requisitos de hardware

Os requisitos de hardware aqui apresentados são baseados nos sistemas de desenvolvimento disponíveis aos membros do grupo sendo que são exclusivamente limitados pelos meio disponiveis existentes, sendo estes:

Refª	Requisito de Hardware	Req. funcionais relacionados
RH.1	Necessidade de um computador por posto de trabalho, com uma distribuição dos três sistemas operativos principais (Windows e distribuições de Linux)	RF.16
RH.2	Necessidade de cada posto de trabalho ter acesso à Internet para ligação com a base de dados.	
RH.3	Necessidade de cada aplicação ser utilizada por um profissional	

Capítulo 3

Modelo de Casos de utilização

3.1 Visão geral

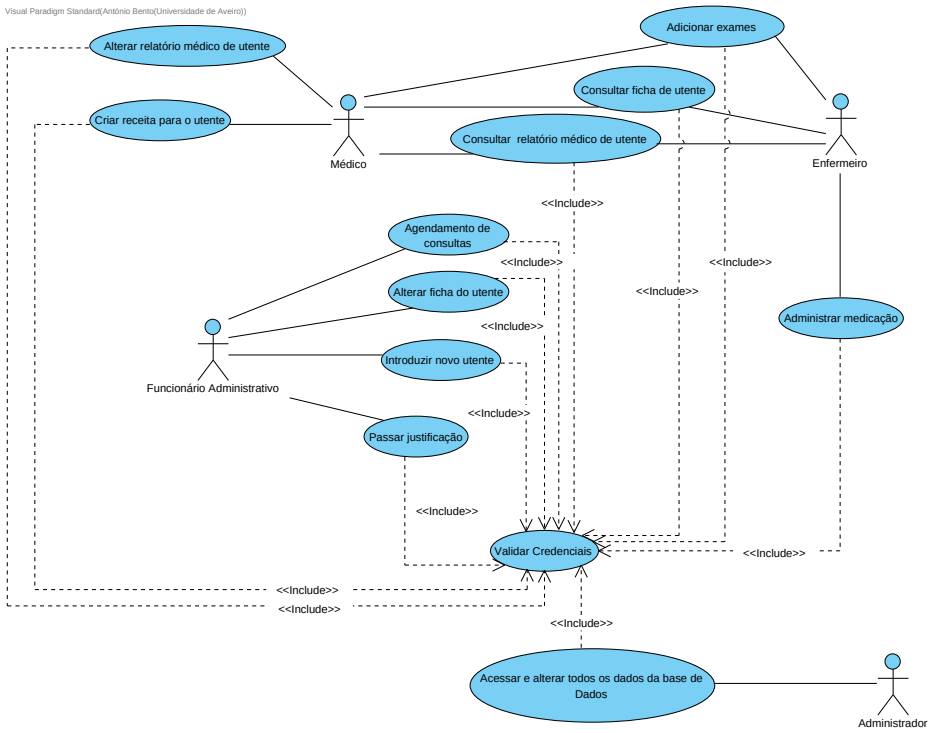

Figura 3.1: Diagrama de Casos de Uso.

Com a definição dos requisitos da clínica foi possível identificar funcionalidades e processos necessários para o correto funcionamento do sistema. Com isto, foi elaborado um diagrama de casos de uso que identifica e organiza as principais atividades que devem ser implementadas na aplicação. É possível verificar, na figura 3.1, que cada ator está relacionado a um caso de uso, mas por particularidade, o sistema em questão, antecedendo à execução de qualquer atividade, o utilizador tem de ser validado com as respetivas credenciais.

3.2 Atores

Ator	Descrição
Médico	Profissional de medicina dentária responsável por: - Realizar consulta; - Consultar ficha de utente; - Consultar e alterar relatório médico; - Adicionar exame.
Enfermeiro	Profissional de saúde que auxilia o médico e executa certas tarefas individualmente, tais como: - Consultar relatório medico e ficha de utente; - Administrar medicação; - Adicionar exames.
Funcionário Administrativo	Funcionário da clínica responsável pelas tarefas administrativas, tais como: - Introduzir novo utente; - Alterar a ficha do utente; - Conceder justificação médica.
Utente	Paciente que recorre à clínica para receber tratamento dentário
Administrador	Responsável pelo bom funcionamento do sistema da clínica bem como gestão e verificação do mesmo.

3.3 Descrição dos casos de utilização

3.3.1 Validar Credenciais

Um funcionário administrativo que pretende introduzir um novo utente, necessita de antes validar as suas credenciais introduzindo-as na interface do *login* e verificar que estas são validadas pelo sistema de *login*.

Nome:	**Validar Credenciais**
Atores:	Administrador, Médico, Enfermeiro e Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Validar o acesso ao sistema por parte de cada funcionário e isolar as suas permissões
Requisitos Funcionais:	RF.5
Pré-condições:	RF.18, RF.19, RF.20 e RF.21
Sumário:	Os funcionários da clínica quando abrem o software, são apresentados com uma interface com campos de texto para introduzir credenciais, que permitem identificar cada utilizador

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O profissional clica no executável	1.1- Apresenta uma interface para introduzir "Nome de utilizador"
2. O profissional introduz o nome de utilizador	1.2- Apresenta uma interface para introduzir a "Password"
3. O profissional introduz a password	4.1- Valida as credenciais na base de dados
4. O funcionário confirma o pedido de acesso	4.2- A base de dados confirma a validação
	4.3- A interface encaminha para a página associada ao profissional
Sequências alternativas	
4.1.1- A base de dados invalida as credenciais	
4.2.1- O sistema devolve mensagem de erro	
4.3.1- O sistema mantém-se na página de acesso permitindo nova tentativa	

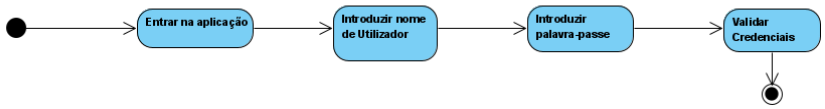

Figura 3.2: Diagrama de atividades - Validar credenciais.

3.3.2 Registar profissional

O administrador procede ao preenchimento do formulário referente à criação de um novo profissional e atribui também as permissões conforme a profissão que o ator vá realizar. Após este preenchimento, o formulário é submetido e são guardadas as alterações no sistema.

Nome:	**Registar profissional**
Atores:	Administrador
Prioridade (1/3):	Alta
Finalidade:	Registar funcionários no sistema e atribuir as devidas permissões
Requisitos Funcionais:	RF.2, RF.3 e RF.4
Pré-condições:	RF.18, RF.19, RF.20 e RF.21
Sumário:	O Administrador digita as credenciais no sistema, o sistema após a validação e seleciona a opção de criar um utilizador, é confrontado com um formulário onde pode associar profissões diferentes a cada utilizador.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. Introduz as credenciais no sistema	1.1- Apresenta a opção de "Criar um funcionário"
2. O administrador preenche o formulário e associa a profissão a cada utilizador	1.2- Apresenta um formulário para preencher sobre o funcionário
3. O administrador submete o formulário	3.1- A estrutura de dados do formulário é validada
4. O administrador clica no botão de saída	3.2- O sistema guarda os dados e credenciais na base de dados
	3.3- O sistema atribui as devidas permissões a cada conta de utilizador
	3.4- O sistema confirma operação
	3.5- O sistema mostra o botão de saída
Sequências alternativas	
1.1.1- Nega acesso e regista na base de dados o erro	
3.1.1- Caso exista algum dado inválido, realça o mesmo e pede para voltar a introduzir	
3.2.1- A comunicação com a base de dados não é possível	
3.4.1- O sistema falha no registo do profissional	

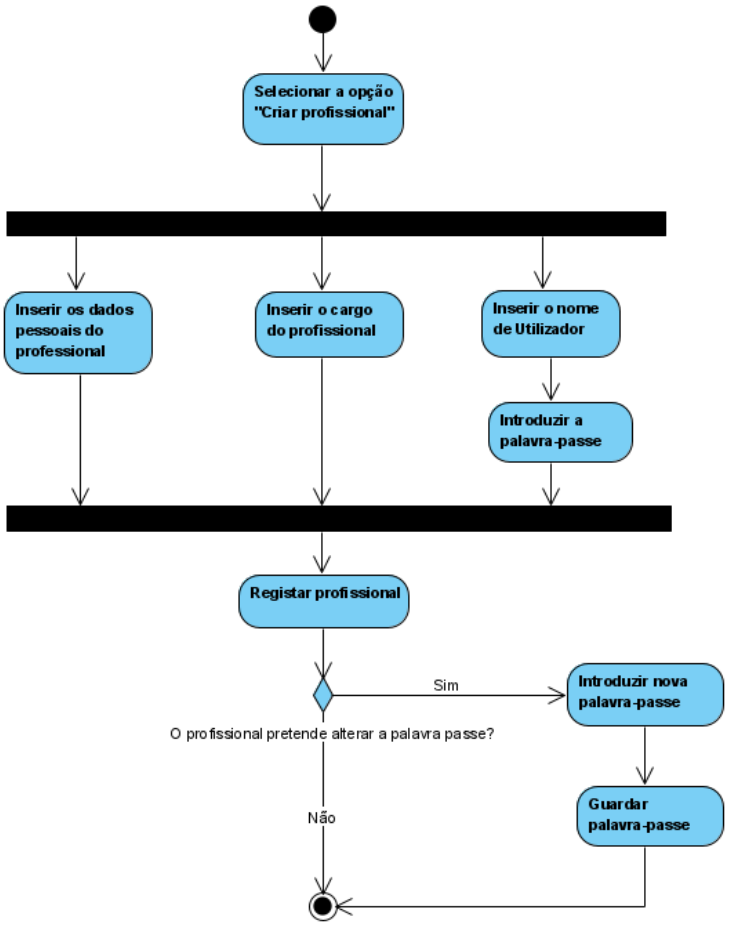

Figura 3.3: Diagrama de atividades - Registar profissional.

3.3.3 Introduzir novo utente

Entra um utente que nunca tinha estado na clínica anteriormente. Este utente tem de ser adicionado à base de dados do sistema, ao criar uma ficha para o mesmo. O funcionário administrativo é o responsável por essa adição, pedindo um conjunto de dados ao utente para preencher o formulário. Após isto, o formulário é submetido e guardado no sistema.

Nome:	**Introduzir novo utente**
Atores:	Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Introduzir novo Utente na Clínica
Requisitos Funcionais:	RF.5, RF.6, RF.10
Pré-condições:	O Funcionário Administrativo deverá ter credenciais para entrar no sistema
Sumário:	O funcionário deverá digitar as credenciais no sistema, para depois poder registar os dados o Utente no sistema e cria a ficha do Utente. Quando completo, o Utente fica registado no Sistema.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O funcionário regista os dados do Utente e cria a ficha do Utente	1.1- O sistema guarda esses dados na base de dados
2. Atribuir ao utente um médico	2.1- O sistema guarda essa atribuição
3. Agendar a primeira consulta	3.1-O sistema marca na agenda do médico essa consulta
	3.2 - O sistema regista o Utente com o sucesso
Sequências alternativas	

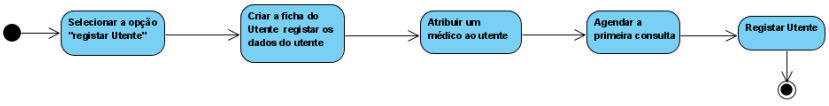

Figura 3.4: Diagrama de atividades - Introduzir novo utente.

3.3.4 Alterar ficha do utente

Um utente com ficha na clínica dirige-se à mesma com o intuito de marcar uma consulta mas, após uma verificação dos dados, repara que a sua morada não está atualizada e precisa de o fazer. Para isso, o funcionário administrativo acede à ficha do utente e procede então à alteração dos dados na ficha e guarda-as no sistema.

Nome:	**Alterar ficha do utente**
Atores:	Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Alterar e atualizar a ficha do utente com novos dados
Requisitos Funcionais:	RF.5, RF.6
Pré-condições:	O Funcionário Administrativo deverá ter credenciais para entrar no sistema
Sumário:	O funcionário deverá digitar as credenciais no sistema, para alterar os dados do Utente no sistema. Quando completo, o Utente fica com os seus dados atualizados na sua ficha de Utente.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O funcionário abre a ficha do utente	1.1- O sistema disponibiliza a ficha
2. O funcionário modifica os dados do Utente	2.1- O sistema guarda esses dados na base de dados 2.2 - O sistema altera os dados com sucesso
Sequências alternativas	

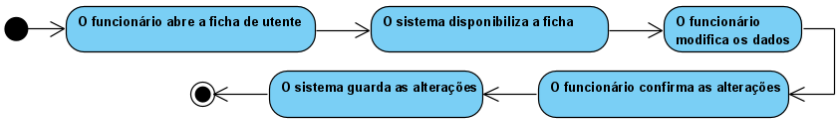

Figura 3.5: Diagrama de atividades - Alterar ficha do utente .

3.3.5 Agendamento de consultas

Um utente dirige-se à clínica com o intuito de marcar uma consulta. Este pede então ao funcionário administrativo, que irá abrir a agenda do médico de forma a verificar a disponibilidade, depois apresenta as opções ao utente que deve escolher a mais conveniente. Após a escolha, o funcionário de balcão confirma o agendamento da consulta em sistema e informa o seu correto agendamento.

Nome:	**Agendamento de consultas**
Atores:	Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Agendar uma consulta para o utente
Requisitos Funcionais:	RF.5, RF.23
Pré-condições:	O Funcionário Administrativo deverá ter credenciais para entrar no sistema
Sumário:	O funcionário deverá digitar as credenciais no sistema, para efetuar o agendamento da consulta na agenda do médico correspondente. Quando completo, o Utente fica com a próxima consulta marcada.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O funcionário abre a agenda do médico	1.1- O sistema disponibiliza a agenda
2. O funcionário marca a consulta na agenda do médico	2.1- O sistema verifica se o médico está disponível para essa consulta 2.2- O sistema marca a consulta com sucesso
Sequências alternativas	
O sistema poderá recusar a marcação da consulta devido a indisponibilidade do médico	

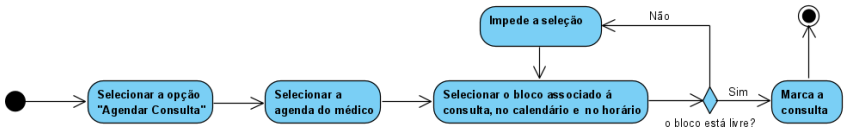

Figura 3.6: Diagrama de atividades - Agendamento de consultas.

3.3.6 Passar justificação

Após realizada uma consulta, um utente que esteja a faltar a um compromisso e que necessite de uma justificação, deve dirigir-se ao funcionário administrativo para a pedir. O funcionário preenche então o formulário da justificação e de seguida imprime a mesma, entregando-a ao utente.

Nome:	**Passar justificação**
Atores:	Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Facultar justificação ao utente no caso de estar a faltar a algum compromisso como trabalho ou escola
Requisitos Funcionais:	RF.5, RF.24
Pré-condições:	O Funcionário Administrativo deverá ter credenciais para entrar no sistema
Sumário:	O funcionário deverá digitar as credenciais no sistema, para depois preencher o formulário da justificação. Quando concluído, o funcionário imprime a justificação e entrega ao utente.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O funcionário abre o formulário da justificação	1.1- O sistema disponibiliza o formulário
2. O funcionário preenche os campos com os dados do utente	2.1- O sistema verifica que todos os campos estão preenchidos
3. O funcionário imprime a justificação	
4. É entregue a justificação ao utente	
Sequências alternativas	

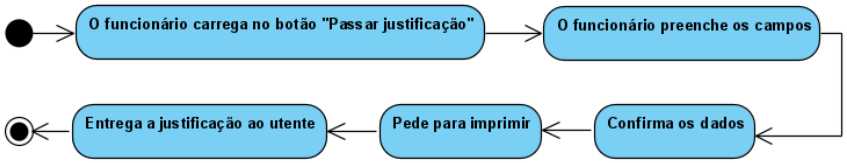

Figura 3.7: Diagrama de atividades - Passar justificação.

3.3.7 Criar receita para o utente

Durante uma consulta, um médico repara numa infeção presente na boca do utente que necessita de tratamento antibiótico. O médico seleciona então a opção de criar uma receita para o utente, preenche os campos do formulário com as devidas informações. E depois o funcionário Administrativo procede à sua impressão e entrega-a ao paciente no fim da consulta.

Nome:	**Criar receita para o utente**
Atores:	Médico, Funcionário Administrativo
Prioridade (1/3):	Alta
Finalidade:	Passar receita referente a algum medicamento ao utente
Requisitos Funcionais:	RF.3, RF.7
Pré-condições:	O médico deverá ter credenciais para entrar no sistema
Sumário:	O médico deverá digitar as credenciais no sistema, para selecionar e preencher o formulário das receitas. Quando finalizado, o médico imprime a recita e entrega ao utente.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O médico seleciona o formulário das receitas	1.1- O sistema apresenta o formulário
2. O médico preenche com os dados necessários	2.1- O sistema verifica que todos os campos estão preenchidos
3. O funcionário Administrativo imprime a receita	3.1- O sistema valida a receita e imprime
Sequências alternativas	

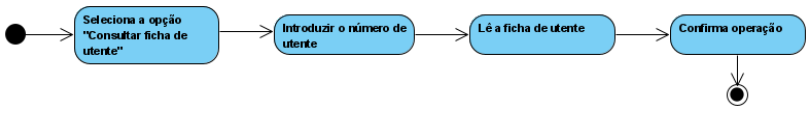

Figura 3.8: Diagrama de atividades - Criar receita para utente.

3.3.8 Alterar relatório médico do utente

Aquando a realização de uma consulta, é da responsabilidade do médico registar no sistema o resultado da mesma incluindo todos os detalhes necessários para que mais tarde seja possível verificar essa informação.

Nome:	**Alterar relatório médico do utente**
Atores:	Médico
Prioridade (1/3):	Alta
Finalidade:	Atualizar o relatório do utente
Requisitos Funcionais:	RF.3, RF.5, RF.11
Pré-condições:	O médico deverá ter credenciais para aceder ao sistema
Sumário:	O médico deverá digitar as credenciais no sistema, para selecionar o utente e poder alterar o relatório médico. Quando terminado, o relatório fica atualizado.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O médico pesquisa o utente em questão	1.1- O sistema apresenta a lista de utentes
2. O médico abre o relatório	2.1- O sistema apresenta a informação referente ao utente selecionado
3. O médico altera o relatório conforme necessário	3.1- O sistema guarda as alterações feitas no relatório
4. O médico fecha o relatório	
Sequências alternativas	

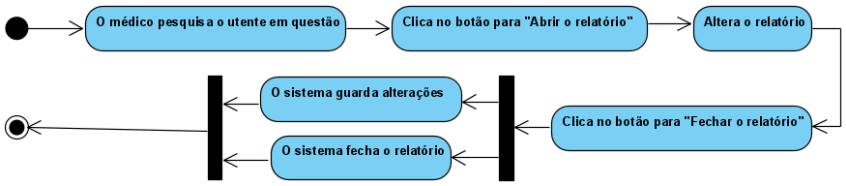

Figura 3.9: Diagrama de atividades - Alterar relatório médico do utente.

3.3.9 Consultar relatório médico do utente

Um cliente chega à consulta com o médico. O médico necessita então de rever o relatório médico do utente, para ter o ponto de situação sobre a consulta que vai realizar.

Nome:	**Consultar relatório médico do utente**
Atores:	Enfermeiro e médico
Prioridade (1/3):	Alta
Finalidade:	Consultar informação presente no relatório médico do paciente
Requisitos Funcionais:	RF.3, RF.5, RF.9
Pré-condições:	O enfermeiro e/ou o médico deverão ter credenciais para aceder ao sistema
Sumário:	O médico e o enfermeiro deverão introduzir as credencias, para depois poderem pesquisar o utente e de seguida consultar o relatório médico.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O médico e/ou o enfermeiro pesquisa o utente em questão	1.1- O sistema apresenta a lista de utentes
2. O médico e/ou enfermeiro consulta o relatório	2.1- O sistema apresenta o relatório referente ao utente selecionado
3. O médico e/ou o enfermeiro fecha o relatório	
Sequências alternativas	

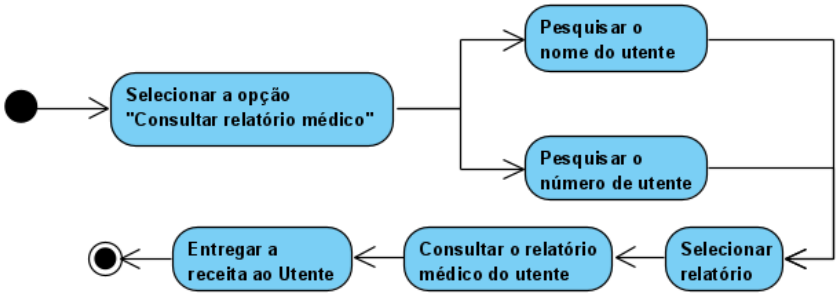

Figura 3.10: Diagrama de atividades - Consultar relatório médico do utente.

3.3.10 Consultar ficha do utente

Após consulta, o utente solicita que lhe seja entregue uma justificação. O funcionário administrativo pesquisa pelo utente em questão, elaborando a justificação e entregando a mesma ao utente.

Nome:	**Consultar relatório médico do utente**
Atores:	Médico e enfermeiro
Prioridade (1/3):	Alta
Finalidade:	Consultar informação presente na ficha de utente
Requisitos Funcionais:	RF.3, RF.4, RF.8
Pré-condições:	Médico e/ou enfermeiro devem ter credenciais para aceder ao sistema
Sumário:	O médico e o enfermeiro deverão introduzir as credencias, para depois poderem pesquisarem o utente e de seguida consultar a ficha de utente.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O Profissional pesquisa o utente em questão	1.1 O sistema apresenta a lista de utentes
2. O Profissional consulta a ficha de utente	2.1- O sistema apresenta a ficha referente ao utente selecionado
3. O Profissional fecha a ficha de utente	
Sequências alternativas	

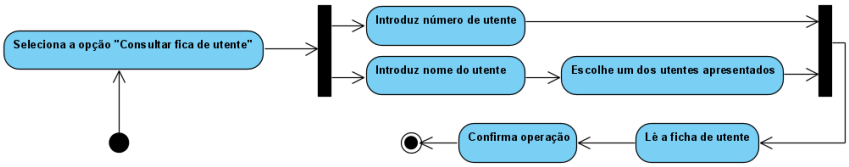

Figura 3.11: Diagrama de atividades - Consultar ficha de utente.

3.3.11 Adicionar exames

Em qualquer consulta, o médico pode requisitar a realização de exames. Tendo em consideração esta informação, o cliente, após a realização do exame, entra na consulta e entrega o exame ao médico ou enfermeiro que, de seguida, o introduzirá no sistema, dando a informação que o mesmo foi introduzido com sucesso.

Nome:	**Adicionar exames**
Atores:	Médico, Enfermeiro
Prioridade (1/3):	Alta
Finalidade:	Adicionar uma descrição de um exame médico
Requisitos Funcionais:	RF.3, RF.13
Pré-condições:	Médico e/ou enfermeiro devem ter credenciais para aceder ao sistema
Sumário:	Adicionar ao relatório uma descrição de um exame médico

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O médico seleciona a opção para adicionar o exame	1.1- A interface apresenta o botão para adicionar exame médico
2. O médico seleciona o ficheiro a importar	2.1- A interface abre o explorador de ficheiros do sistema
3. Utilizador carrega no botão de saída	3.1- Importa o ficheiro para o sistema 3.2- Envia o ficheiro para a base de dados 3.3- Associa o ficheiro ao relatório da consulta 3.4- Devolve mensagem de exportação concluída 3.5- Interface mostra botão de saída

Sequências alternativas
1.1- Os médicos não têm a sessão iniciada no sistema e precisa de entrar na sua conta
3.1.1- O programa devolve mensagem de ficheiro inválido a ser importado
3.1.2- A importação falha ou é interrompida a meio
3.2.1- Comunicação com a base de dados não ser possível

Figura 3.12: Diagrama de atividades - Adicionar exames.

3.3.12 Administrar medicação

Depois da entrada do utente na consulta, foi necessário administrar medicação ao cliente. Para tal, o enfermeiro verifica a medicação a administrar e após a executar, registou em sistema o que foi administrado e por quem foi administrado.

Nome:	**Administrar medicação**
Atores:	Enfermeiro
Prioridade (1/3):	Alta
Finalidade:	Facilitar o processo de administração de medicação por parte do enfermeiro
Requisitos Funcionais:	RF.3, RF.12
Pré-condições:	Enfermeiro deve ter credenciais para aceder ao sistema
Sumário:	O enfermeiro dirige-se ao sistema com a sua sessão iniciada e consulta a ficha de utente. Lá terá uma opção para administrar medicação, que guardará essa informação.

Sequência típica dos eventos	
Ações dos atores	**Respostas do sistema**
1. O enfermeiro consulta a ficha de utente	1.1 A interface apresenta a ficha de utente
2. O enfermeiro seleciona a opção "Administrar Medicação"	2.1- Interface apresenta um campo para inserção da medicação a administrar.
3. O enfermeiro preenche esse campo e guarda a informação	3.1- Dados guardados na base de dados
Sequências alternativas	

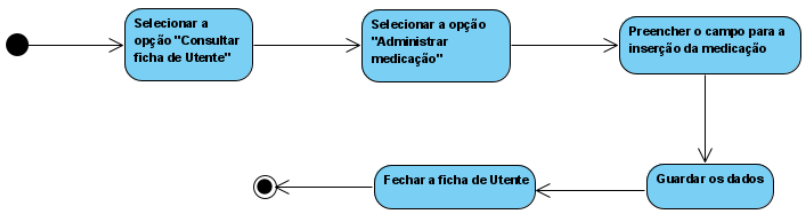

Figura 3.13: Diagrama de atividades - Administrar medicação.

3.4 Cobertura de requisitos

	RF.1	RF.2	RF.3	RF.4	RF.5	RF.6	RF.7	RF.8	RF.9	RF.10	RF.11	RF.12	RF.13	RF.14	RF.15	RF.16	RF.17	RF.18	RF.19
Validar credenciais		x	x											x					
Registar profissional		x	x											x					
Introduzir novo utente	x	X	X	x				x						x				x	
Alterar ficha de utente	x	X	X	x				x						x					x
Agendamento de consultas	x	x	x	x										X	x	x			
Passar justificação	x	x	x	x										X			x		
Criar receita para utente	x	x	x	x	x		x		x	x	x			X					
Alterar relatório médico do utente	x	x	x	x	x	x					x			X					
Consultar relatório médico do utente	x	x	x	x	x	x			x	x		x		X					
Consultar ficha do utente	x	x	x	x				x						X					
Adicionar exames	x	x	x	x	x	x	x				x		x	X					
Administrar medicação	x	x	x	x				x	x	x		x		x					

Figura 3.14: Tabela de funcionalidades em função de requisitos.

Capítulo 4

Modelo de conceitos do domínio

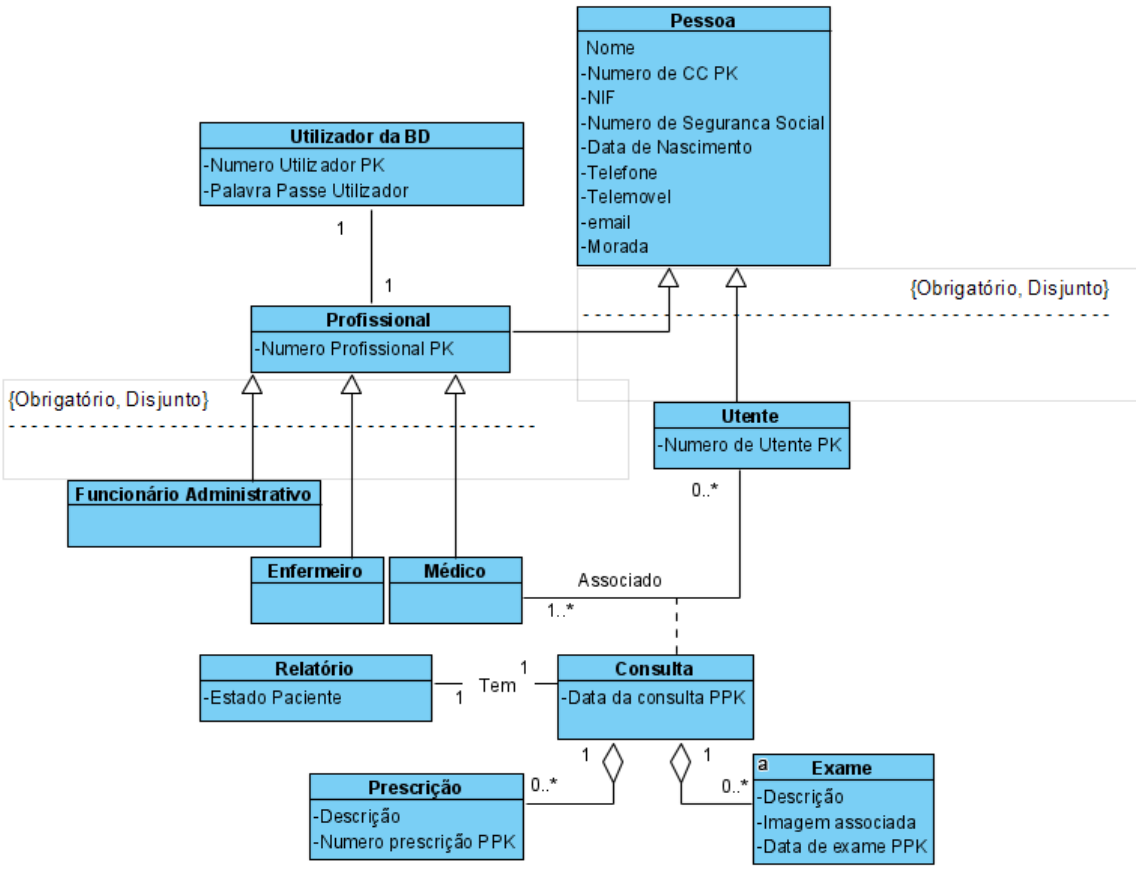

Figura 4.1: Diagrama de Classes

Após compreender os processos e intervenientes da clínica dentária, foi possível confirmar que tipologia terá o sistema no que toca à sua estrutura estática. Dito isto, foi elaborado um diagrama de classes que visa determinar todas as classes do sistema e os seus atributos, tal como se pode verificar na Figura 4.1.

Confirma-se a presença de relações na Figura 4.1. Em primeiro lugar, o diagrama acima indicado foi elaborado tendo em conta quem interage com a aplicação e que tipos de interações existem entre estes e os utentes, permitindo descrever que atributos vão ser usados e onde são alojados, como por exemplo a relação existente entre médico e profissional, sendo que médico é

uma especialização e profissional é uma generalização, também como, partindo do pressuposto, a consulta deriva da interação entre o médico e o cliente que por si tem um relatório e a qual cada cliente tem um e apenas um médico associado.

Capítulo 5

Especificação Lógica

O objetivo principal foi desenvolver uma aplicação capaz de suportar o sistema de gestão de uma clínica dentária. Tendo conhecimento da dimensão da equipa de desenvolvimento (6 elementos), ficou decidido que em cada semana todos os elementos se orientavam para um conjunto de requisitos específicos, permitindo organizar corretamente o trabalho. Foi também atribuído um *project owner* a cada ciclo (1 semana) que contacta o cliente e apresentava, a cada ciclo o que foi desenvolvido pela equipa, usando assim um modelo ágil que se assemelha ao *scrum*. Foram concluídas fases iniciais do projeto tais como a fase da análise e definição de requisitos e o planeamento do sistema de desenvolvimento.

Os requisitos da clínica dentária foram definidos em maior parte aquando da análise do sistema e requisitos, sendo assim possível transitar para a fase de modelação e planeamento com grande parte das funcionalidades em consideração. Em simultâneo com o planeamento, o cliente foi contactado sempre que necessário para qualquer orientação tendo sempre em atenção às necessidades do sistema.

No que toca à estrutura de desenvolvimento, o grupo seguiu as indicações do *project owner*, sendo que este elaborou o *sprint backlog* que contém as necessidades a ser implementadas para cada *sprint*. A equipa organizou-se de modo a que cada elemento recebesse tarefas com responsabilidade, carga horária e grau de dificuldade semelhantes. O objetivo predominante nas fases iniciais foi preparar o ambiente para cada elemento e foram tomadas decisões como o *software* de desenvolvimento e *software* de controlo de versões. Dito isto, são enumeradas as seguintes decisões:

- *Software* de controlo de versões: Git
- Servidor para controlo de versões: GitHub
- Sistema de gestão de base de dados: MySQL com MySQL WorkBench

Com o objetivo de obter uma boa estrutura de desenvolvimento tomou-se a decisão de usar o mesmo *software* em todas as máquinas de desenvolvimento da aplicação;

- IDE para desenvolvimento em Python: Atom, PyCharm, Sublime, Spyder, VSCode e CLI.
- *Framework* de desenvolvimento de interface gráfica: QtDesigner
- *Software* de desenvolvimento de documentação formal: LaTeX com TexStudio

Capítulo 6

Modelo dinâmico

6.1 Funções de sistema

Aqui vamos apresentar todas as funcionalidades relacionadas com o ator que as executa. Existem também certas funcionalidades que estão interligadas entre si.

6.1.1 Validação de Credenciais

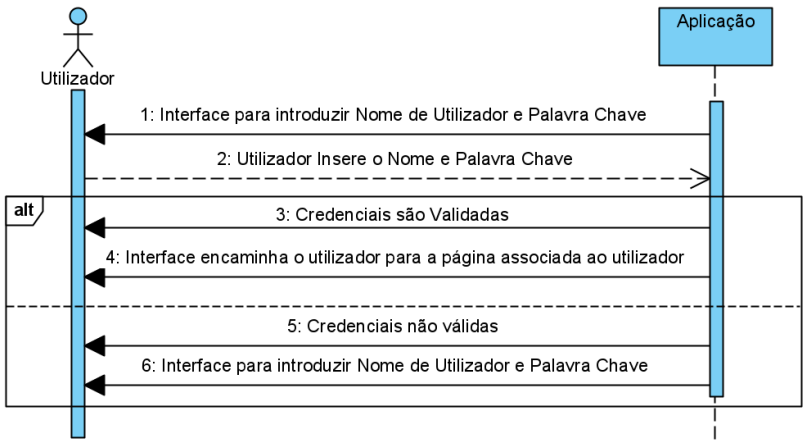

Figura 6.1: Diagrama de sequências de validação de credenciais.

A validação de credenciais dá-se para todos os utilizadores do sistema. É um caso importante pois permite e/ou delimita as funcionalidades que cada utilizador possa fazer.

6.1.2 Registar Profissional

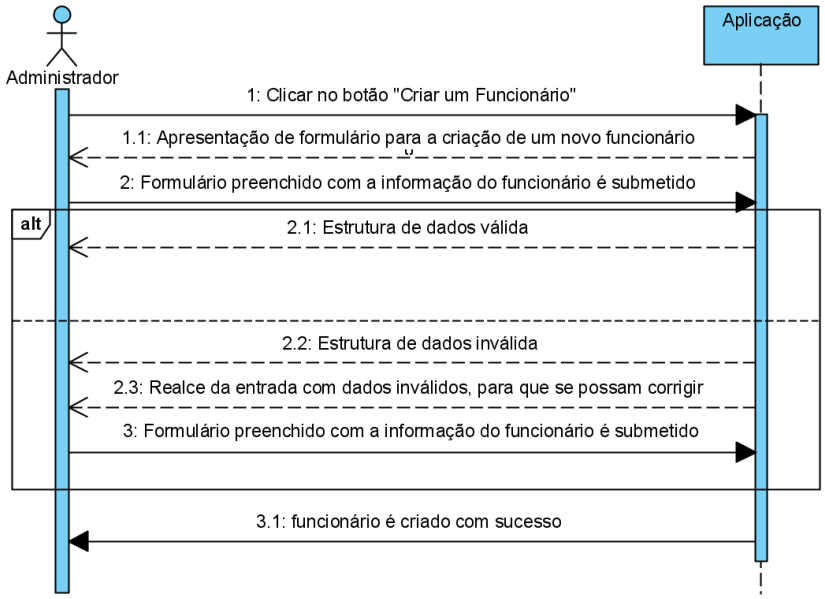

Figura 6.2: Diagrama de sequências de registar profissional.

Esta função é exclusiva para o administrador do sistema. Serve para adicionar profissionais ao sistema, como médicos, funcionários administrativos e enfermeiros, que fará uma atribuição de permissões para com a conta do utilizador.

6.1.3 Introduzir novo utente

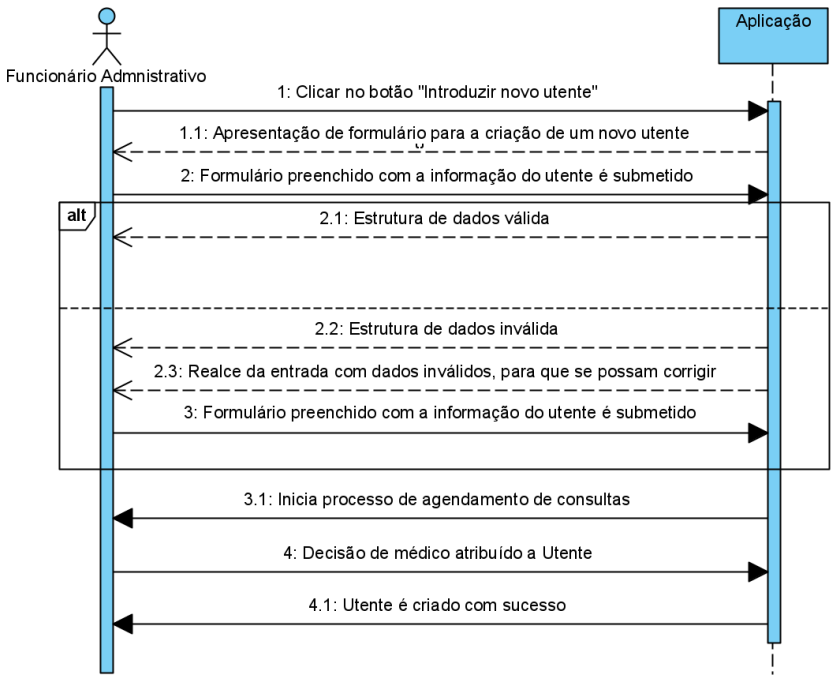

Figura 6.3: Diagrama de sequências de introduzir novo utente.

A introdução de um novo utente é feita pelo funcionário administrativo. Este processo serve para fazer a criação de um novo utente para que todos os seus dados fiquem guardados na base de dados e para que se possa criar a respetiva ficha.

6.1.4 Alterar Ficha de Utente

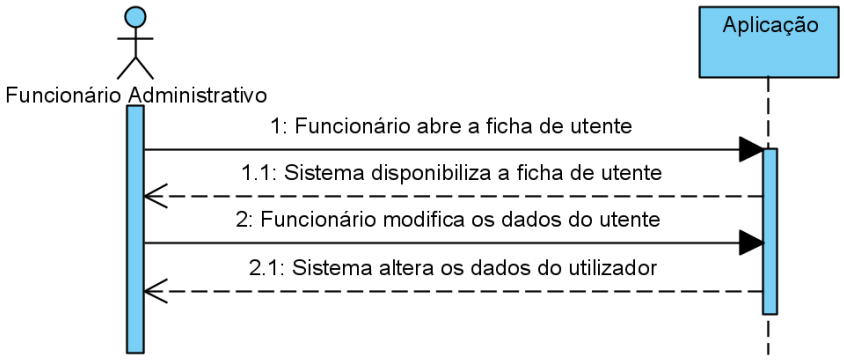

Figura 6.4: Diagrama de sequências de alterar ficha de utente.

Esta função é usada pelo funcionário administrativo e serve para a alteração de dados na ficha de utente, quando necessário.

6.1.5 Agendamento de Consultas

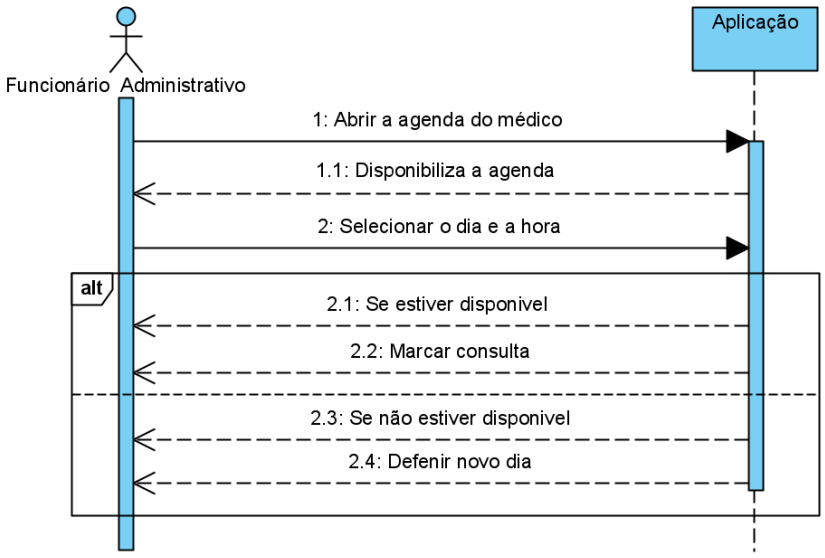

Figura 6.5: Diagrama de sequências de agendamento de consultas.

A funcionalidade poderá ser utilizada pelo funcionário administrativo. Tem como objetivo a marcação de uma consulta num determinado dia e hora.

6.1.6 Passar Justificação

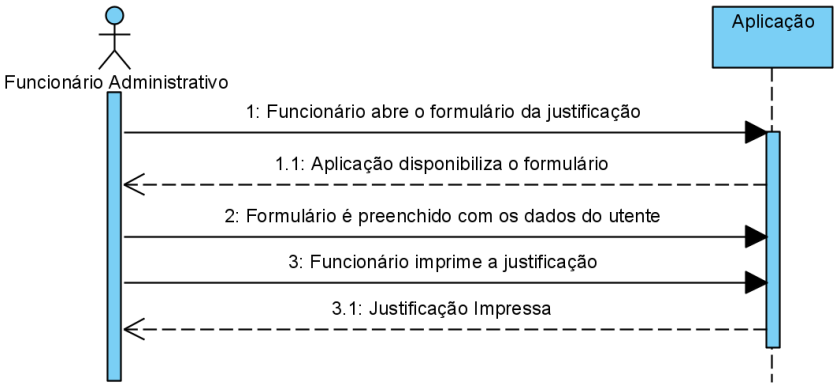

Figura 6.6: Diagrama de sequências de passar justificação.

Esta função será apenas utilizada pelo funcionário administrativo. Tem como objetivo criar um comprovativo de como o utente esteve de facto numa consulta.

6.1.7 Criar Receita

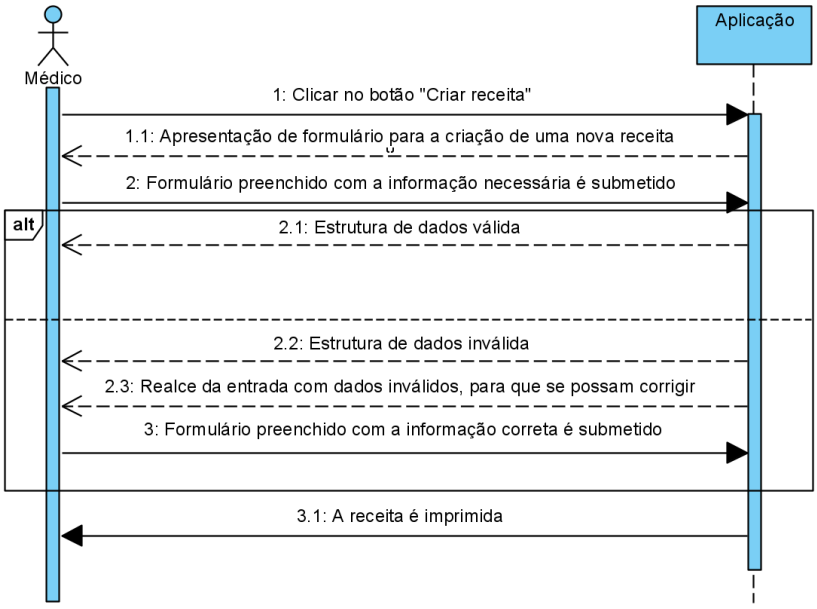

Figura 6.7: Diagrama de sequências de criar receita.

Esta função será utilizada pelo médico e o funcionário administrativo. Em que o médico com ela cria uma receita para o utente, para que este possa comprar o medicamento nela descrito numa farmácia. E o funcionário administrativo imprime a receita e entrega-a ao utente.

6.1.8 Alterar Relatório Médico do Utente

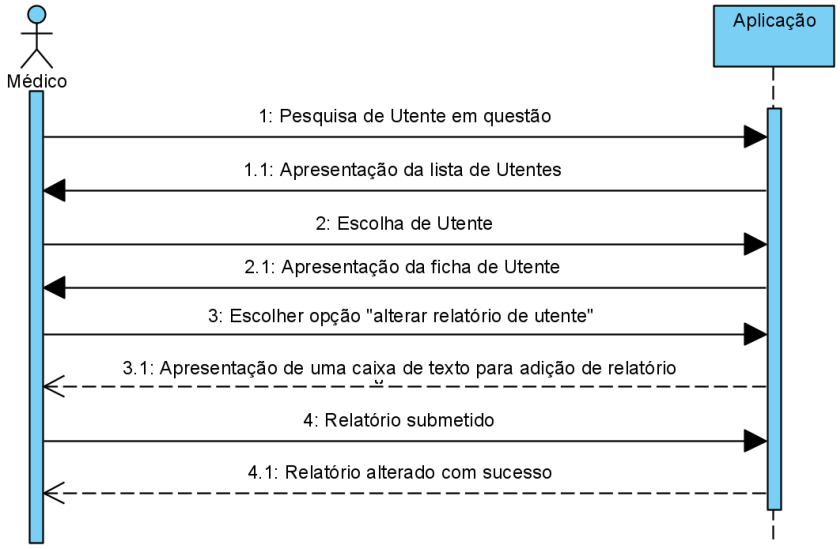

Figura 6.8: Diagrama de sequências de alterar relatório médico do utente.

A função mencionada apenas poderá ser utilizada pelo médico. Com esta, o médico altera ou acrescenta informação ao relatório médico em questão.

6.1.9 Consultar Relatório Médico do Utente

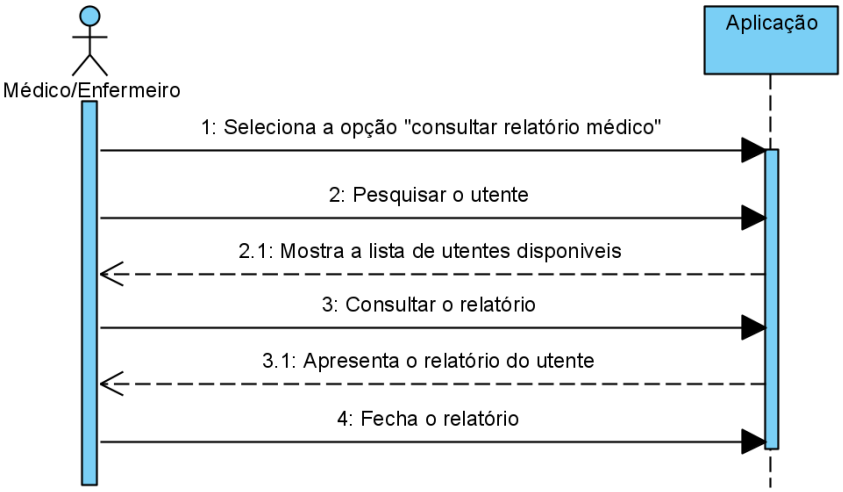

Figura 6.9: Diagrama de sequências de consultar relatório médico do utente.

A função presente pode ser utilizada pelo médico e pelo enfermeiro. Com esta, os atores podem consultar o relatório médico para consultar as suas informações.

6.1.10 Consultar Ficha de Utente

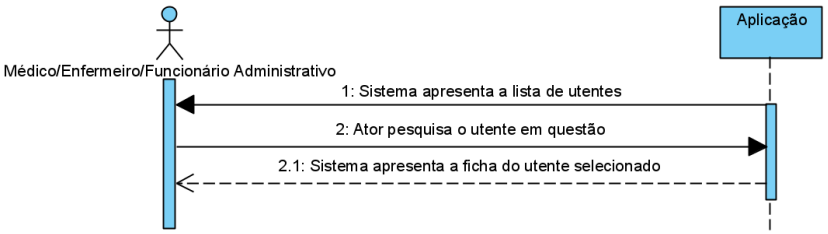

Figura 6.10: Diagrama de sequências de consultar ficha do utente.

A funcionalidade poderá ser utilizada pelo médico, enfermeiro e funcionário administrativo. Tem como objetivo consultar a ficha de utente, para aceder tanto aos dados pessoais como aos dados de consultas anteriores (que estão guardadas nos relatórios médicos).

6.1.11 Adicionar Exames

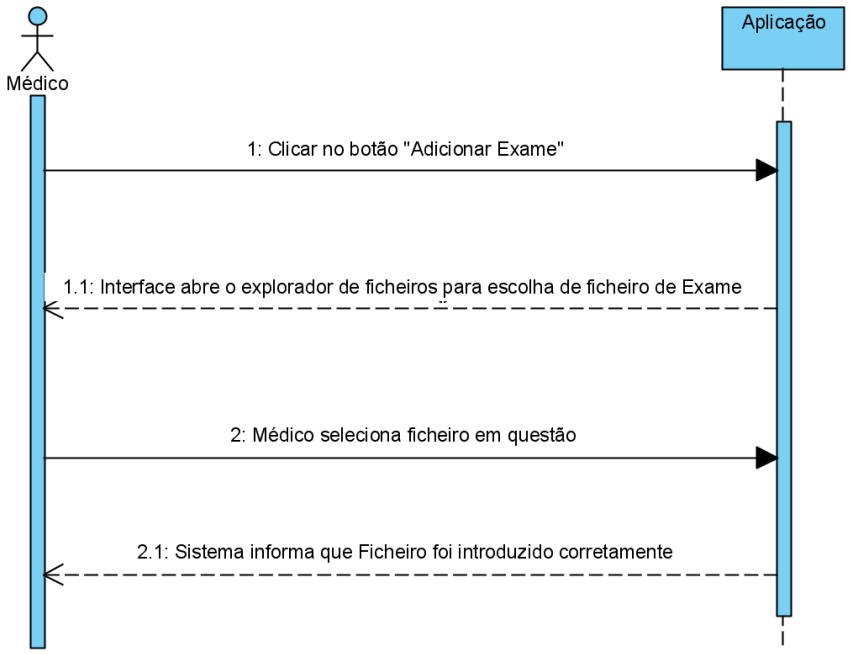

Figura 6.11: Diagrama de sequências de adicionar exames.

Esta função poderá ser utilizada pelo médico e pelo enfermeiro. Consiste em adicionar um exame ao relatório médico do utente.

6.1.12 Administrar Medicação

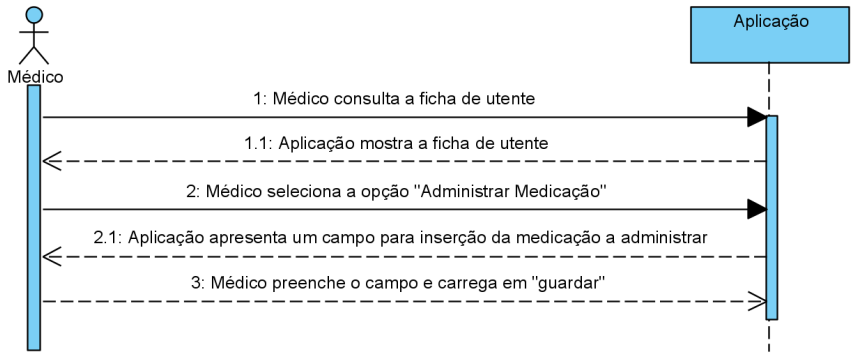

Figura 6.12: Diagrama de sequências de administrar medicação.

A funcionalidade aqui descrita é utilizada pelo enfermeiro e consiste em guardar no relatório médico o que foi administrado e por quem foi administrado durante a consulta.

6.2 Colaborações entre objetos

Existem certas funcionalidades que estão diretamente ligadas entre si. Entre estas, a mais importante é Validar Credenciais, que acontece assim que o utilizador faz o *Login* na máquina, que por sua vez dará ao utilizador todas as permissões que necessita para executar as respetivas tarefas.

Existem também vários casos de consulta que estão diretamente relacionados com casos de alteração de dados:

- Consultar e Alterar relatório médico.
- Consultar e Alterar ficha de utente.

Capítulo 7

Modelo de dados persistente

Dados os requisitos estabelecidos e a divisão lógica do modelo conceptual, foi elaborado o seguinte modelo de dados persistentes:

- utente (nome, nr_cartao_cidadao, nif, nr_seg_social, data_nascimento, telefone, telemovel, email, morada, numero_utente)
- id_funcionario (nome, nr_cartao_cidadao, nif, nr_seguranca_social, data_nascimento, telefone, telemovel, email, morada, funcionario_numero, profissao)
- funcionario (numero, utilizador_nome, palavra_passe, secret_key)
- consulta (data, utente_numero, relatorio_consulta, funcionario_numero, aviso_consulta, hora, medicacao_administrada, prescricao_consulta)
- exame (int, consulta_data, consulta_utente_numero, data_exame, imagem_exame)

Capítulo 8

Implementação Base de Dados

Derivado do modelo de dados persistente, foi elaborado um esquema de SQL, que pode ser verificado na Figura 8.1, referente aos dados em questão, de forma a que cada elemento da equipa de desenvolvimento fosse capaz de testar o código individualmente, a base de dados foi partilhada pelos elementos do grupo tornando-se local.

Na base de dados surgiu a necessidade identificar que tuplos estão ativos ou inativos, com isto foram definidos atributos adicionais, nomeadamente foram adicionados às relações "id_funiconário"e "utente"o atributo "ativo", este define se o funcionário/utente está presente na base de dados, é possível associar este atributo à função de eliminar (quando eliminamos um funcionário, o atributo ativo passa a ser 0 em vez de 1), tal como é possível verificar no modelo físico da base de dados representado pela figura Figura 8.1.

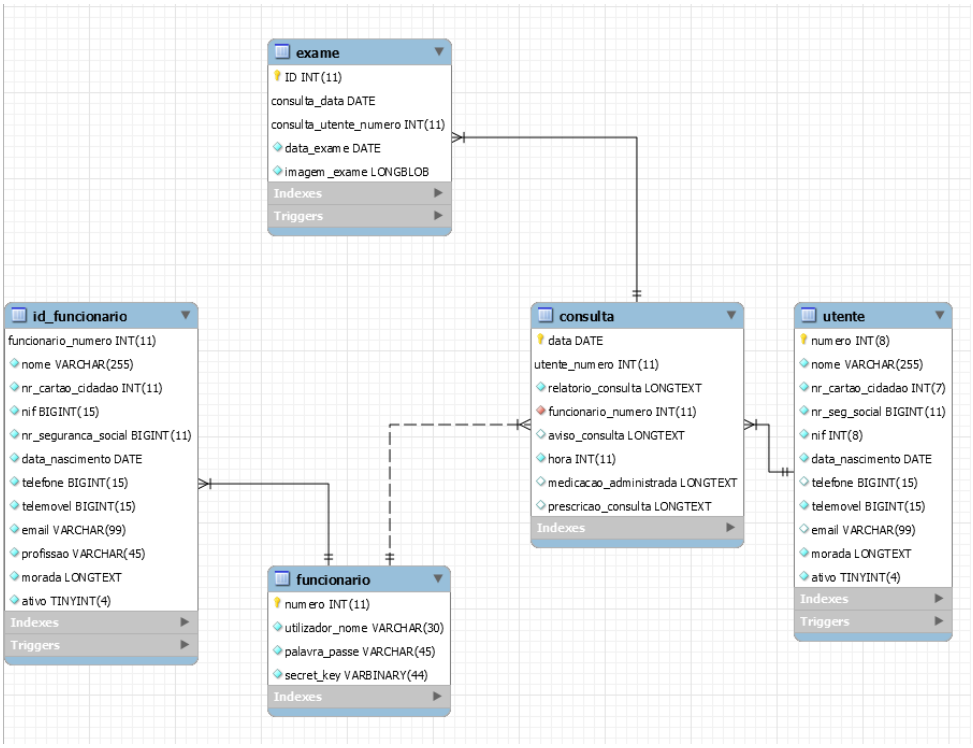

Figura 8.1: Modelo físico da Base de Dados SQL.

Inicialmente, a base de dados foi elaborada a partir do modelo de classes utilizando engenharia inversa, sendo que foi um método bastante prático para implementar as bases de dados sem qualquer informação prévia, partindo de um modelo desenhado. Com o evoluir de várias versões da base de dados, esta passou a ficar guardada em ficheiros SQL, de forma a permitir a existência

de um utilizador universal presente na primeira utilização (admin) e para simplicidade em termos de implementação do sistema. Foram também implementados constrangimentos necessários para que o sistema se comporte em conformidade com as restrições previamente mencionadas no Capítulo 2.

Devido à necessidade funcional de colocar a base de dados operacional na rede da Universidade de Aveiro, um esquema atualizado com base neste requisito foi migrado para a base de dados presente na rede da Universidade. Aquando da migração das bases de dados locais para a base de dados da rede UA, a utilização de *scripts* (.sql) foi uma mais valia pela fácil atualização dos mesmos para o *schema* e endereço da base de dados, juntamente com a necessidade de deixar um utilizador administrativo pronto para a utilização com a aplicação desenvolvida.

Por fim, tendo um esquema na base de dados online, este foi verificado e testado exaustivamente até a verificada a conformidade com os requisitos estabelecidos. Na última etapa foram implementados constrangimentos evitando assim que certas verificações se façam em código e que sejam apenas controladas pelo Sistema de Gestão de Bases de Dados, facilitando também o uso de outras aplicações para a base de dados elaborada.

Capítulo 9

Código e Design

Embora exista uma clara distinção entre o código e a interface do utilizador, ambos foram desenvolvidos em paralelo de modo a permitir um ambiente de desenvolvimento útil para *debugging*.

Dada a divisão de profissões dentro da clínica, a equipa de desenvolvimento decidiu dividir os ficheiros de sistema de acordo com a profissão em causa, para que fosse possível um significativo nível de controlo sobre as funcionalidades de cada utilizador, mesmo existindo esta divisão, existe ainda uma divisão lógica caracterizada pela distribuição de código associado à mesma tarefa por cada ficheiro, como por exemplo o "Connection.py"que alberga código orientado para alterar variáveis no âmbito da conexão à base de dados.

9.1 Programação

9.1.1 Planeamento

Em primeiro lugar, é necessário ter as seguintes ideias em mente, o projeto é dividido em vários ficheiros, nomeadamente organizados por funcionalidade ou por uma classe associada à profissão do utilizador, existe a necessidade de ter uma estrutura coesa que ligue a interface ao código funcional da base de dados, cada elemento do grupo é responsável por uma parte considerável do produto final, de modo a distribuir responsabilidade irmãmente pelos membros da equipa.

9.1.2 Preparação

Tendo em conta os conceitos descritos anteriormente, foi decidido utilizar um sistema de controlo de versões capaz de monitorizar e gerir o trabalho realizado, tendo em conta as dimensões do projeto e a utilidade da aplicação o git foi estabelecido como plataforma de controlo de versões, na qual existe uma ligação ao GitHub que permite um controlo eficaz e constante do desenvolvimento do projeto.

9.1.3 Desenvolvimento

Numa fase inicial foi estipulado elaborar um ficheiro de testes ("Index.py") para que sejam adicionadas e alteradas funcionalidades aquando necessário, associado a este foi elaborada uma interface de testes designada por "Index.ui", ambos permitiram suportar o desenvolvimento de novas funcionalidades, que mais tarde foram migradas para os respetivos ficheiros.

Pondo a relação dos ficheiros de um modo lógico, o ficheiro principal a ser executado denomina-se "main.py", este é responsável por validar o *login* de utilizadores e chamar corretamente a interface/código associado à profissão registada.

Dado que foram elaborados ficheiros para cada tipo de profissão, cada um destes inicia uma interface individualmente, possibilitando a correta separação de funcionalidades e ecrãs pelos diferentes tipos de utilizadores de sistema. Existem 4 interfaces distintas, estas sendo:

- Administrador
- Médico
- Enfermeiro
- Rececionista

As funcionalidades de cada interface apresentada correspondem diretamente com as indicadas na Figura 3.1.

Os contactos com a base de dados são uma componente fundamental para o correto funcionamento do sistema permitindo que sejam introduzidos, consultados, ou alterados dados da base de dados, estes contactos foram realizados estabelecendo uma ligação que é caraterizada, nomeadamente, pelos conteúdos do ficheiro "Connection.py".

Foi tomada em consideração a necessidade de terminar todas as chamadas com a base de dados assim que estas se tornem desnecessárias, resultando numa base de dados que pode ser acedida por várias instâncias do programa abertas.

9.2 Design Gráfico

Durante o desenvolvimento da interface é necessário ter em consideração que a esta tem como finalidade ser um instrumento de trabalho, sendo assim, esta deve ser simples de forma a que o utilizador identifique rapidamente a ferramenta que procura e dinâmica de maneira a que se adapte ao ecrã onde está a ser utilizada.

O design visual deverá usar cores claras como fundo e letra preta de forma ao contraste ser mais confortável ao olhos e assim mais visível e identificável[5], tal como pode ser visto na Figura 9.1. Contudo o esquema de cores fica também dependente das cores em uso pelo sistema operativo, pelo que um utilizador que prefira um tema de cores escuras também o irá usufruir do mesmo esquema com a nossa interface se assim o desejar.

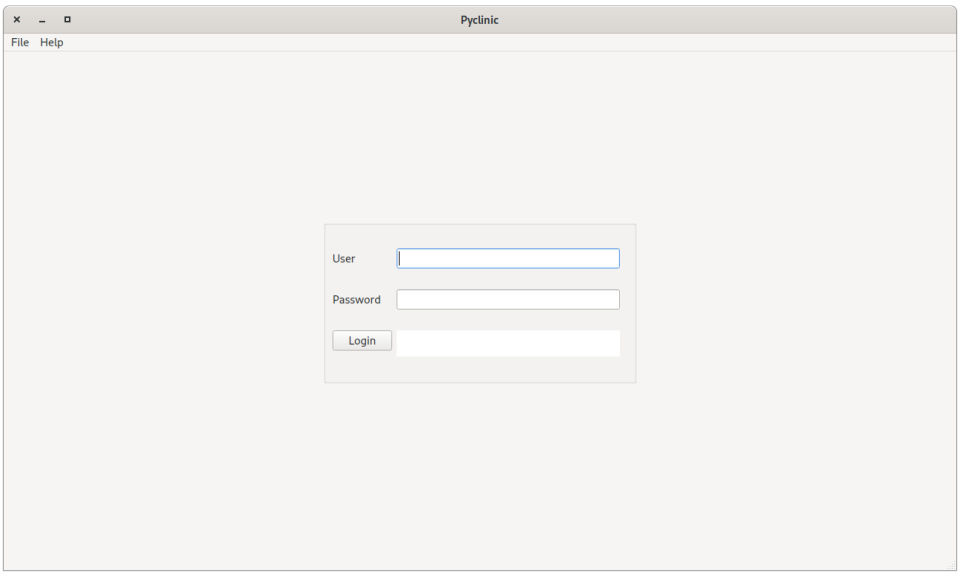

Figura 9.1: *Screenshot* da janela de *login*.

As ferramentas e/ou funcionalidades ficam separadas sob a forma de abas de modo a que o interface não fique carregado com demasiada informação. Por forma a assegurar que dados não são acedidos de forma incorreta, certas funcionalidades do interface ficam bloqueadas até que seja feita uma ação que as liberte como por exemplo escolher um utente antes de entrar no menu de exames.

Ações de uso repetido como aceder a funcionalidades comuns a interfaces ficam mapeadas com atalhos no teclado. Funcionalidades críticas como *logout* e sair do programa ficam escondidas numa barra de menu por forma a que o utilizador não as ative por um clique acidental. Estes exemplos de design podem ser observados na Figura 9.2.

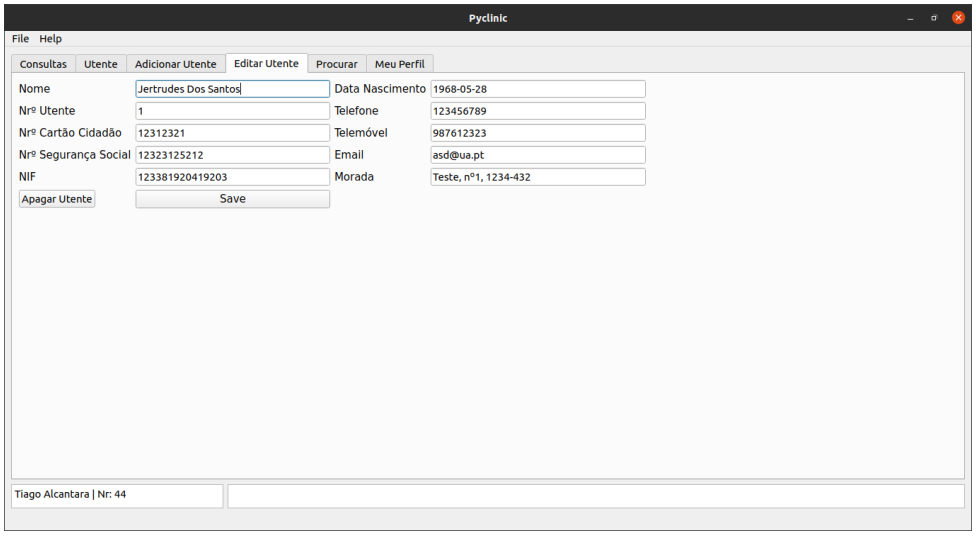

Figura 9.2: *Screenshot* da janela de *login*.

A navegação entre entradas nos formulários fica simplificada com o uso da tecla "TAB"e garantiu-se que a transição entre entradas é navegada de forma ordeira.

9.3 Testes de Robusteza

Nesta fase do projeto, os testes foram realizados ao longo do desenvolvimento da aplicação, nomeadamente na fase final, pois foi nesta fase que o grupo se dedicou mais aos testes para observar o que se podia melhorar e corrigir, para uma melhor apresentação da aplicação.

Os testes realizados consistiam em verificar se a aplicação continha o comportamento esperado, e também verificar a reação da mesma, caso a má utilização da aplicação ou o caso de utilizador insir dados incorretos.

Com o resultado dos testes, o grupo melhorava alguma funcionalidade, ou corrigia algum erro que aparece-se, para uma melhor interpretação por parte do utilizador.

Capítulo 10

Conclusões

Neste projeto foi proposto e implementado com sucesso um *software* de gestão para uma clínica dentária com capacidade para organizar dados e guardar-los numa base de dados passível de ser acedida numa rede interna ou até externa à clínica.

Foi desenvolvido software onde as funcionalidades foram conseguidas com recurso à linguagem de programação Python e o interface gráfico foi desenhado com o programa Qt Designer que usa a *framework* Qt como *backend* para desenhar as janelas. Para satisfazer a necessidade de guardar/consultar dados de uma forma segura e organizada, foi implementada uma base de dados em linguagem MySQL que fica alojada na rede. Esta base de dados facilita também a funcionalidade de *login* já que os dados de utilizador ficam também guardados na rede. Por forma a garantir rastreabilidade de todos os eventos realizados por profissionais de saúde ou de funcionários administrativos, foi implementada uma funcionalidade de *loger* que guarda todos os eventos realizados. Estima-se que quase todas as funcionalidades propostas foram implementadas com exceção da atualização em tempo real das consultas à base de dados que não foi implementada devido ao grau de complexidade acrescido que não se verificou como justificável durante as últimas etapas do projeto.

Com este projeto o Grupo 5 teve a oportunidade de desenvolver melhor aptidão à linguagem de programação Python, superando assim o seu desafio auto proposto, e também a oportunidade de implementar os conhecimentos adquiridos de bases de dados agora num caso prático virado para um uso real. Foi também possível ganhar conhecimentos com ferramentas alternativas às que tinham sido propostas para o desenvolvimento de interface, proporcionando-nos assim conhecimentos com o *software* estado de arte usado pelas principais empresas de desenvolvimento de aplicações gráficas.

Olhando para o futuro, conseguimos imaginar melhorias que podem ser implementadas, como por exemplo no desenvolvimento da interface usar botões ou descrições que usem ficheiros com tabelas de nome/descrição por forma a que seja possível distribuir o *software* com capacidade de trocar de língua, trocando apenas no código um ficheiro que contém todo o texto a ser desenhado para a língua em questão. De um ponto de vista funcional seria também interessante implementar funcionalidade para alertar um utente, por email ou telefone, de uma consulta quando esta se estiver a aproximar.

Bibliografia

[1] Electrotechnical and Computer Science Conference, *Algorithm design in Python for cybersecurity*, Sept. 2019.

[2] J. M. Redondo and F. Ortin, "A comprehensive evaluation of common python implementations," *IEEE Software*, vol. 32, pp. 76–84, jul 2015.

[3] A. A. Khwaja, M. Murtaza, and H. F. Ahmed, "A security feature framework for programming languages to minimize application layer vulnerabilities," *Security and Privacy*, vol. 3, no. 1, p. e95, 2020.

[4] M. C. Neto, S. S. Andrade, and R. L. Novais, "Cross-platform multimedia application development: For mobile, web, embedded and iot with qt / qml," in *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, WebMedia '17, (New York, NY, USA), p. 23–26, Association for Computing Machinery, 2017.

[5] D. Bauer and C. Cavonius, "Improving the legibility of visual display units through contrast reversal," *Ergonomic aspects of visual display terminals*, no. 1980, pp. 137–142, 1980.