Lea Venusa Maurício dos Anjos Marino Arone

Licenciada em Ciências da Engenharia Eletrotécnica e de Computadores

Desenvolvimento duma infraestrutura para gestão de medicamentos

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador: Professor Doutor João Almeida das Rosas,
Professor auxiliar, Universidade Nova de Lisboa

Júri:

Presidente: Doutor Rui Alexandre Nunes Neves da Silva - FCT/UNL

Vogais: Doutor João Almeida das Rosas - FCT/UNL (Orientador)
Doutor Filipe de Carvalho Moutinho - FCT/UNL (Arguente)

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Novembro, 2020

Desenvolvimento duma infraestrutura para a gestão de medicamentos

À minha mãe.

Agradecimentos

A realização desta dissertação não seria possível sem o apoio e incentivo de pessoas muitas importantes e as quais sou eternamente grata.

Em primeiro lugar quero agradecer ao orientador da presente dissertação, o Professor João Almeida das Rosas, por ter aceitado orientar uma das tarefas mais importantes do meu percurso académico. Agradeço imenso pela sua disponibilidade, ajuda, apoio e orientação.

Gostaria também de agradecer ao meu irmão, Filipe Isaías, pelo apoio e conselhos dados durante todo o meu percurso académico. Agradecer também pela sua ajuda ilimitada e por estar disponível para ajudar sempre que eu precisasse.

Por fim, gostaria de agradecer à minha mãe, que sempre contribuiu para o meu crescimento pessoal e que me transmitiu valores pelos quais me guio diariamente. Por todo carinho e apoio durante todos os momentos da minha vida.

Resumo

O medicamento é uma tecnologia atualmente usada na cura e prevenção de grande parte das patologias existentes. Porém, se usados de forma incorreta, os medicamentos tornam-se num risco à saúde e acabam por desempenhar o papel oposto no que toca à saúde e bem-estar de quem os necessita. Fazer a gestão dos mesmos pode tornar-se numa tarefa complicada. Quando se juntam fatores como o avanço da idade, grande número e variedade de medicamentos a administrar, essa tarefa torna-se ainda mais complexa.

De modo a combater este problema, a presente dissertação incide no desenvolvimento de um sistema capaz de fazer a gestão de medicamentos de um indivíduo. O sistema será composto por dois grandes componentes: uma aplicação android e um dispensador. Através dos dados inseridos pelo utente ou pelo seu cuidador na aplicação android, o sistema cria automaticamente uma agenda que permite fazer a gestão e visualização do estado das administrações dos medicamentos. O dispensador armazena os medicamentos e, na hora de cada medicação, o mesmo sistema emite um sinal sonoro para alertar o utilizador e dispensa a medicação adequada.

De um modo geral, este trabalho tem como objetivo o desenvolvimento de uma infraestrutura capaz de fazer a gestão de uma quantidade variada de medicação. Permitindo que os medicamentos sejam administrados sem falhas, aliviar a pressão e stress que existe por detrás da correta administração de fármacos e acima de tudo garantir o bem-estar e saúde de cada utilizador do sistema.

Palavras-chave: Medicamento, Gestão, Android, Aplicação, app.

Abstract

Medicines are a technology currently used in the healing and prevention of most existing pathologies. However, if used incorrectly, medicines become a health risk and end up playing the opposite role when it comes to the health and well-being of those in need. Managing medications can become a complicated task. When factors such as advancing age, large numbers and variety of medications are added to this task, it becomes even more complex.

To fight this problem, this dissertation focuses on the development of a system capable of managing drugs for an individual. The system will consist of two major components: an android application and a dispenser. Through the data entered by the user or his caregiver in the android application, the system automatically creates an agenda that allows the management and visualization of the state of the administrations of the medications. The dispenser stores the medications and, at the time of each medication, the same system beeps to alert the user and dispenses the appropriate medication.

In summary, this system aims to develop an infrastructure capable of managing any amount of medication. Allowing medications to be administered without fails, relieving the pressure and stress that exists behind the correct administration of drugs and above all ensuring the well-being and health of each user of the system.

Keywords: Medicine, Management, Android, Application, App.

ÍNDICE

LISTA DE FIGURAS ..... **XV**

LISTA DE TABELAS ..... **XVII**

1 INTRODUÇÃO **1**

1.1. Motivação e Enquadramento 1

1.2. Objetivos 2

1.3. Estrutura da Dissertação 3

2 REVISÃO DA LITERATURA **5**

2.1. Erro de medicação 5

2.2. Dispensadores de medicamentos 8

2.2.1. Tecnologias existentes 9

2.2.2. Vantagens e desvantagens dos dispensadores automáticos de medicamentos 12

2.3. Serviço de Web Hosting 13

2.4. Linguagem PHP e Aplicações Web 15

2.5. Base de Dados 19

2.5.1. phpMyAdmin 19

2.5.2. SQL e MYSQL 20

2.6. Android Studio 23

2.6.1. Estrutura do projeto Android 24

2.7. Kit de simulação Staudinger 25

3 DESENVOLVIMENTO DO SISTEMA **27**

3.1. Requisitos funcionais 27

3.2. Arquitetura do projeto 30

3.3. Diagrama de sequência UML do sistema 31
3.4. Desenvolvimento no Ambiente Android Studio 33
3.4.1. Administração de medicamento 43
3.5. Serviços de Web Hosting 46
3.5.1. Base de dados 46
3.5.2. Módulos PHP 47
3.6. Dispensador e Aplicação Java 51
3.6.1. Componentes necessários 52
3.7. Conexão entre a APP Android e a aplicação JAVA 59

4 TESTES DO SISTEMA 63

4.1. Verificação 63
4.2. Validação do dispensador 75

5 CONCLUSÕES 79

5.1 Síntese do trabalho desenvolvido 79
5.2 Resultados 80
5.3 Trabalhos futuros 81

Lista de Figuras

Figura 2.1 MedMinder [12]. 9
Figura 2.2: Dispensador automático de medicamentos 10
Figura 2.3 Dispensador automático de medicamento Hero [9]. 10
Figura 2.4: Serviço de web hosting 14
Figura 2.5: Estrutura linguagem PHP. Inspirado em [21]. 16
Figura 2.6: Comunicação PHP parte I. Inspirado em [21]. Símbolo Apache [22] 16
Figura 2.7: Comunicação PHP parte II. Inspirado em [23]. Símbolo Apache [22]. 17
Figura 2.8: Protocolo HTTP. 18
Figura 2.9: Interface web phpMyAdmin. 20
Figura 2.10: Criação de uma nova base de dados 21
Figura 2.11: Criação da tabela alunos. 21
Figura 2.12: Inserção dados na tabela alunos. 22
Figura 2.13: Comando SELECT para pesquisa de aluno 23
Figura 2.14: Resultado do comando SELECT para pesquisa de aluno. 23
Figura 2.15 Estrutura de um projeto android Studio 24
Figura 2.16: Kit de Simulação Staudinger. 25
Figura 3.1: Funcionamento do sistema. 28
Figura 3.2: Esquema representativo da metodologia de desenvolvimento do projeto. 30
Figura 3.3: Diagrama de sequência funcionamento geral da aplicação. Criado em [45] 32
Figura 3.4: Hierarquia View no ambiente Android Studio. 33
Figura 3.5: Layout início de sessão formato código. 34
Figura 3.6: Layout de início de sessão formato design. 34
Figura 3.7: Layout activity_main_menu.xml. 35
Figura 3.8: (1) Layout visualização dados medicamento, (2) Layout Lista de medicamentos 36
Figura 3.9: Inserção de novo medicamento na aplicação. 37
Figura 3.10: Deteção do código de barras através da câmara do dispositivo. 38
Figura 3.11: Caixa de diálogo com o resultado da leitura do código de barras 38
Figura 3.12: Inserção de medicamento com após leitura do código de barras. 39
Figura 3.13: Seleção da data de início de um medicamento. 39
Figura 3.14: Mensagem de erro caso tempo de administração superior a 30 dias. 40
Figura 3.15: Layout Agenda. 41
Figura 3.16: Visualização dos dados do utente na aplicação. 42

FIGURA 3.17: HISTÓRICO DA MEDICAÇÃO NA APLICAÇÃO. 43
FIGURA 3.18: FICHEIRO EXECUTABLESERVICE.JAVA 44
FIGURA 3.19: FICHEIRO CONFIRMACAO.JAVA 44
FIGURA 3.20: PROCESSO TOMA DE MEDICAMENTO NO AMBIENTE ANDROID STUDIO. 45
FIGURA 3.21: DIAGRAMA ENTIDADE RELACIONAMENTO 47
FIGURA 3.22: MÓDULO CONN.PHP 48
FIGURA 3.23: MÓDULO LOGIN.PHP 49
FIGURA 3.24: FLUXOGRAMA QUE APRESENTA O PROCESSO DOS MÓDULOS EM PHP. 50
FIGURA 3.25: ARQUITETURA JAVA VIRTUAL MACHINE. INSPIRADO EM [50]. 51
FIGURA 3.26: ESTRUTURA JNI. 52
FIGURA 3.27: FICHEIRO *HEADER* KIT.H. 54
FIGURA 3.28: FICHEIRO DISPENSADOR.JAVA 55
FIGURA 3.29: FICHEIRO SPLITTERLINE.H. 55
FIGURA 3.30: FICHEIRO KIT_JNI.C 56
FIGURA 3.31: FICHEIRO DISPENSADOR.JAVA COM CAMINHO DLL. 56
FIGURA 3.32: APLICAÇÃO JAVA. 57
FIGURA 3.33: DISPENSADOR LOGO DEPOIS DA CALIBRAÇÃO. 58
FIGURA 3.34: COMUNICAÇÃO TCP SOCKET 59
FIGURA 3.35: FICHEIRO SENDSOCKET.JAVA. 60
FIGURA 3.36: FICHEIRO MESSANGESENDER.JAVA 60
FIGURA 3.37: SOCKET AMBIENTE NETBEANS 61
FIGURA 4.1: AMBIENTE INICIAL DA APLICAÇÃO. 64
FIGURA 4.2: TENTATIVA DE INÍCIO DE SESSÃO COM DADOS ERRADOS. 65
FIGURA 4.3: INÍCIO DE SESSÃO EFETUADO COM SUCESSO E MENU PRINCIPAL. 66
FIGURA 4.4: INSERÇÃO DE UM NOVO MEDICAMENTO. 67
FIGURA 4.5: VERIFICAÇÃO DA INSERÇÃO DO NOVO MEDICAMENTO. 68
FIGURA 4.6: DISPENSADORCOM MEDICAMENTO NA GAIOLA 69
FIGURA 4.7: DISPENSADOR ANTES E DEPOIS DE INSERÇÃO DE MEDICAMENTO. 69
FIGURA 4.8: VERIFICAÇÃO DA LISTA DE MEDICAMENTOS DE UM UTENTE 70
FIGURA 4.9: NOTIFICAÇÃO NO DISPOSITIVO MÓVEL. 71
FIGURA 4.10: APLICAÇÃO JAVA COM MENSAGEM *GET* 72
FIGURA 4.11: PROCESSO DE REMOÇÃO DE MEDICAMENTO. 72
FIGURA 4.12: VERIFICAÇÃO DA AGENDA. 73
FIGURA 4.13: SELEÇÃO OUTRO DIA NA AGENDA 74
FIGURA 4.14: HISTÓRICO ANTES DA CONFIRMAÇÃO DO MEDICAMENTO. 74
FIGURA 4.15: HISTÓRICO APÓS ADMINISTRAÇÃO DO MEDICAMENTO. 75
FIGURA 4.16: VALIDAÇÃO DO DISPENSADOR. 76

Lista de Tabelas

TABELA 2.1 TABELA COMPARATIVA DAS DIFERENÇAS ENTRE OS DISPENSADORES AUTOMÁTICOS DE MEDICAMENTOS 12
TABELA 3.1: REQUISITOS FUNCIONAIS NECESSÁRIOS. 29
TABELA 3.2: FICHEIROS PHP MAIS IMPORTANTES E SUA DESCRIÇÃO. 48
TABELA 3.3: FICHEIROS *.C QUE PERMITEM A CONEXÃO COM O DISPENSADOR. 53
TABELA 3.4: FUNÇÕES ESSÊNCIAS DE BAIXO NÍVEL. 53

CAPÍTULO 1

INTRODUÇÃO

1.1. Motivação e Enquadramento

Os medicamentos desempenham um papel fundamental na cura e prevenção da maioria das doenças existentes. A pontualidade e a precisão da aquisição de medicamentos são fatores cruciais para o bom funcionamento da medicação.

Medicamentos são disponibilizados pelos serviços de saúde pelo mundo todo. A situação se agrava pela necessidade de prescrição para uma população que envelhece, e como consequência, as necessidades médicas tornam-se mais complexas e exigem a introdução de novos medicamentos. Todavia, com o aumento substancial de medicamentos vem um risco crescente de cuidados, levando a uma maior necessidade de gestão de medicamentos a cada dia que passa, tanto para adultos mais velhos quanto para médicos.

A administração errada de medicamentos pode levar a lesões temporárias ou permanentes, prejuízos e até a morte em casos mais extremos. Segundo a associação Portuguesa dos Farmacêuticos Hospitalares, os erros de medicação são responsáveis por sete mil mortes anualmente em Portugal [1]. Nos Estados Unidos da América, a administração incorreta de medicamentos afeta 1,3 milhões de pessoas por ano e o custo relacionado à hospitalização do paciente devido as consequências do mesmo chegam a atingir 76,6 bilhões de dólares [2].

Na maior parte dos casos, os erros de medicação apenas são descobertos quando as consequências são manifestadas pelo paciente, como a presença de sintomas ou reações adversas após algum tempo em que foi feita a medicação.

A categorização de severidade de erro médico foi realizada por diferentes autores, que avaliaram de acordo com a gravidade do erro, a necessidade de intervenções médicas e consequências [3].

O erro mais comum na administração de remédios é o uso dos mesmos em horários indevidos ou com intervalos irregulares. Cada fármaco tem um pico e queda de concentração no organismo de duração variada, por isso, cada medicamento deve ser introduzido no corpo com intervalos constantes. No caso da administração domiciliária, muitos pacientes tomam os medicamentos com horas de atraso ou confundem os horários.

Outro erro frequente é o erro na dosagem. Os médicos são conhecidos pela sua caligrafia muitas vezes ilegível, podendo revelar-se num problema na indicação da dosagem de cada fármaco. Para melhor funcionamento, os fármacos precisam atingir uma concentração especifica de acordo com as necessidades de cada indivíduo, não podendo esta dosagem ser inferior ou superior ao recomendado [4].

Os idosos estão mais sujeitos aos erros de administração de medicamentos, normalmente, a toma de medicamentos em idosos é feita em grande quantidade. Perante isso, são precisos cuidados redobrados no que toca a organização e administração, para garantir que a dosagem e horário dos fármacos não sejam afetados [4].

1.2. Objetivos

Esta dissertação tem como objetivo a criação de uma infraestrutura de gestão de medicamentos. O sistema será constituído por uma aplicação Android e por um dispensador. Este dispensador será comandando através de uma aplicação Java e o dispensador servirá para o armazenamento da medicação. Será desenvolvido um dispensador com as características fundamentais de um dispensador automático, como alarmes sonoros e a possibilidade de programação de acordo com as necessidades de cada paciente.

A aplicação Android será responsável pela introdução e visualização dos dados, permitindo uma identificação fácil dos medicamentos e dará aos pacientes ou cuidadores acesso a toda informação como, por exemplo, quando a última toma foi realizada. Este sistema pretende contribuir para uma redução significativa dos erros de medicação e ao mesmo tempo cobrir as necessidades dos pacientes.

1.3. Estrutura da Dissertação

A dissertação encontra-se descrita em cinco capítulos. O primeiro e presente capítulo, introdução, expõe brevemente a descrição do problema e um breve resumo dos objetivos a alcançar para a resolução do problema em causa.

No segundo capítulo, é realizada uma revisão da literatura, onde é feita uma descrição mais detalhada do problema previamente descrito e das suas consequências. Também são apresentadas algumas soluções já existentes e que são semelhantes ao sistema proposto. Posteriormente, é feita uma descrição das tecnologias que ajudarão na construção do sistema. Este capítulo é bastante significativo, pois contém a maior parte das informações necessárias para a construção do sistema.

No terceiro capítulo é apresentado o processo de desenvolvimento do sistema. Neste capítulo, é descrito o processo de criação de cada um dos componentes que farão parte do sistema e como a ligação é feita entre estes diferentes componentes.

No quarto capítulo, são apresentados testes efetuados ao sistema. Estes testes são compostos pelas interações do utilizador com o sistema de modo a averiguar o seu correto funcionamento. Neste capítulo também é apresentado um cenário de validação do sistema.

O quinto capítulo é dedicado às conclusões sobre o trabalho desenvolvido e são feitas algumas sugestões para trabalhos futuros.

Por fim, após as referências bibliográficas, encontram-se em anexo duas tabelas com informações que podem ser importantes para a compreensão do sistema desenvolvido.

CAPÍTULO 2

REVISÃO DA LITERATURA

Neste capítulo são apresentados os conceitos mais importantes para a perceção e enquadramento do problema descrito no capítulo anterior. O capítulo foi dividido em duas partes. Na primeira parte, é apresentado o conceito de erro de medicação, algumas das suas consequências e a análise e descrição de algumas ferramentas atualmente existentes para a gestão de medicamentos. A segunda parte foi reservada para o estudo das tecnologias que serviram de base para a criação do sistema proposto.

2.1. Erro de medicação

Um erro de medicação muitas vezes é apenas visto como a administração errada de um medicamento, mas os erros de medicação incluem ingestão da dose errada, perder ou saltar uma toma ou troca de medicamentos [5].

A definição de erro de medicação segundo a organização americana *National Coordinating Council for Medication Errors Reporting and Prevention* é a seguinte:

> "*Qualquer evento evitável no qual o uso inadequado da medicação pode prejudicar o doente enquanto a medicação está sob o controle dos profissionais de saúde, doentes ou consumidos. Tais eventos podem estar*

relacionados à prática profissional, produtos de saúde, procedimentos e sistemas de assistência médica incluindo a prescrição, comunicação de pedidos, rotulagem de produtos, embalagens, nomenclatura, composição, distribuição, administração, educação, monitoramento e uso" [6].

Os erros de medicação são muito comuns, principalmente em pacientes com uma prescrição com uma elevada quantidade de medicamentos. A maioria dos erros de medicação originam numa omissão, ao invés de acréscimos, ou atrasos no momento das doses.

Pacientes com doenças crónicas como hipertensão, em que não apresentam sintomas desagradáveis tendem a saltar tomas, apesar de nestes casos ser imperativo um regime estrito de medicação.

Algumas das consequências dos erros de medicação são desperdícios de medicamentos, progressão da doença, redução da qualidade de vida, maior uso de recursos médicos como visitas e internamentos hospitalares.

Estima-se que 50% dos cidadãos, em todo o mundo, cometem erros de medicação [7]. Só em Portugal, estima-se que 43 mil pessoas são admitidas em hospitais como resultado de problemas com medicamentos. Um estudo realizado nos Estados Unidos revelou que metade dos pacientes cometiam erros de medicação quando medicados em casa. O estudo contou com a participação de 851 pessoas hospitalizadas por doenças cardiovasculares em Hospitais Universitários. Metade dessa população recebeu as instruções comuns de alta hospitalar, a outra metade recebeu duas visitas de farmacêuticos, uns dias após a admissão hospitalar e outra após alta. Na segunda visita o farmacêutico elaborou um esquema detalhado onde explicava como administrar e a importância de cada medicação. Estes pacientes receberam telefonemas com intervalos que variavam entre um e quatro dias de modo a acompanhar a recuperação [8].

Apesar de todo o acompanhamento pós-alta hospitalar, metade dos pacientes teve um ou mais erros de medicação em casa durante o primeiro mês. Dos erros, 23% eram considerados graves e 2% eram fatais e muitos deles eram evitáveis. O número de erros foi praticamente igual entre os pacientes que tiveram acompanhamento após a alta e os que não tiveram acompanhamento.

O estudo também revelou que as pessoas não alfabetizadas ou que tinham problemas de leitura ou compreensão das instruções do médico, com a ajuda de um farmacêutico os erros de medição reduziram em 32% [8].

As consequências dos erros de medicação não são só na saúde. Anualmente, nos Estados Unidos, mais de 125 mil pessoas morrem devido à falta de capacidade de gestão dos seus medicamentos, somando aproximadamente, mais de 100 bilhões de dólares em custos hospitalares adicionais evitáveis.

O centro de controlo de intoxicações nos Estados Unidos recebeu entre os anos de 2000 e 2012, um total de 67 mil dados sobre erros graves de fármacos que ocorreram fora de instalações de saúde.

A frequência e as taxas dos erros de medicação aumentaram para todas as faixas etária, exceto para crianças com idade inferior a 6 anos. O erro mais comum foi dose incorreta (19.8%), seguido de tomar ou administrar o medicamento errado (18.4%) e tomar o medicamento duas vezes (15.7%). Os medicamentos mais frequentemente associados a estes erros pertenciam as categorias de medicação cardiovascular (20,6%), analgésicos(12%), hormonas (11%) e insulina e antidepressivos(8,6%) [9].

Para os idosos, as estatísticas são particularmente alarmantes. Estima-se qua cerca de 23% das admissões em casas de repouso ocorrem devido à incapacidade de uma pessoa com idade avançada autogerir a sua medicação prescrita em casa. Cerca de 58% de todos os idosos cometem algum tipo de erro de medicação, sendo que 26% cometem erros que podem levar a consequências graves. Estudos com pacientes idosos, em tratamentos a longo prazo para redução do colesterol revelaram que 57% param a medicação após 6 meses e 74% pararam ao fim de 5 anos [10].

2.2. Dispensadores de medicamentos

São variados os motivos que levam as pessoas a deixar de tomar a sua medicação de forma adequada, sendo o motivo mais comum o esquecimento. Um idoso, em média ingere diariamente cerca de 7 medicamentos diferentes, prescritos e não prescritos, tornando mais suscetível o esquecimento [10].

Ao longo dos anos, várias estratégias foram desenvolvidas de modo a não só ajudar os idosos assim como toda a população que necessita de controlar os seus medicamentos. Em resposta a estas necessidades, a indústria produziu uma ampla gama de diferentes sistemas de gestão de medicamentos.

Quando se trata de serviços de gestão de medicamentos as opções disponíveis são variadas. Um dispensador automático de medicamentos é a melhor escolha para pacientes que tomam vários medicamentos diariamente ou que possuam dificuldades de memória e gestão dos mesmos.

Os dispensadores de medicamentos são equipamentos que libertam medicamentos a uma hora especifica ou ajudam no armazenamento do medicamento de modo a facilitar na hora da toma. Para se usar um dispensador, é necessário encher o compartimento com a dose necessária e programar se este possuir esta capacidade [10].

Um dispensador automático de medicamentos não só ajuda que a toma da mediação seja feita na hora certa, como garante que a dosagem certa pré inserida é dispensada, reduzindo assim drasticamente os erros associados com a toma da medicação fora das horas indicadas e os erros de dosagem.

Quando chega a hora certa, o dispensador automaticamente liberta a dose necessária previamente definida. Grande parte dos dispensadores de medicamentos têm vários compartimentos para cada dose, e alguns chegam a ter quase 30 compartimentos.

Alguns dispensadores possuem alarmes sonoros de modo a chamar a atenção do utilizador para a toma dos medicamentos.

Seguidamente, serão apresentados alguns dos dispensadores de medicamentos existentes no mercado.

2.2.1. Tecnologias existentes

A. MedMinder

Os dispensadores de comprimidos MedMinder lembram o usuário de tomar os seus medicamentos com alertas visuais e/ou auditivos opcionais. Os lembretes ocorrem em intervalos de 30 minutos. MedMinder permite que se tenham quantos cuidadores forem necessários ligados ao sistema.

Primeiro o compartimento emite uma luz intermitente, o distribuidor de comprimidos emite um sinal sonoro e o usuário recebe o lembrete por telefone. No caso de o usuário não tiver tomado os medicamentos, todos os cuidadores serão notificados.

O dispensador possui um botão que pressionado permite comunicar com os profissionais de alerta médica no centro de monitoramento [11]. O dispensador encontra-se ilustrado na Figura 2.1.

Figura 2.1 MedMinder [12].

B. Philips Lifeline Medication Dispenser

O Philips Lifeline Medication Dispenser é uma boa opção para as pessoas que tomam medicamentos menos complexos. Os cuidadores também podem ser alertados sobre doses, erros, reabastecimentos e até se o sistema perder eletricidade.

Este sistema possui sessenta compartimentos e pode ser programado para até seis doses diárias. O dispositivo possui um gabinete bloqueado e fornece lembretes [8]. O dispensador encontra-se ilustrado na Figura 2.2.

Figura 2.2: Dispensador automático de medicamentos

Philips Lifeline Medication Dispenser [13].

C. Hero

Hero apenas liberta os medicamentos quando um botão é pressionado após um sinal sonoro. A gestão dos medicamentos é feita através de uma aplicação móvel. O dispensador automatizado de medicamentos Hero pode armazenar dez comprimidos diferentes para até noventa dias, independentemente do tamanho ou formato. Os cuidadores ou profissionais de saúde podem ser alertados sobre doses atrasadas ou não tomadas. A app também informa quando um medicamento está quase a acabar e precisa ser carregado [14]. O dispensador encontra-se ilustrado na Figura 2.3.

Figura 2.3 Dispensador automático de medicamento Hero [9].

D. EMMA by INRange

Os medicamentos são guardados em blisters, cada um contendo dez doses. Os medicamentos apenas estão acessíveis depois que a entrega de medicamentos seja aprovada. Para a entrega dos medicamentos, um utilizador autorizado deve estar conectado ao sistema. Isto é feito através da leitura do código QR presente no cartão que foi atribuído ao utilizador. Também é possível uma conexão ao sistema manualmente. O equipamento irá então dispensar os medicamentos num compartimento que abrir-se-á [15].

E. MEDCENTER Your Minder

Se o paciente estiver a tomar vários medicamentos ao mesmo tempo, poderá precisar de mais assistência técnica. Se apenas for necessário lembrar-se das horas que se deve ingerir o medicamento, um despertador pode ser o suficiente. O MEDCENTER oferece um despertador para medicamentos chamado Your Minder que possui uma tela digital com um altifalante. O sistema permite configurar até quatro alarmes diários. O sistema também inclui uma caixa de comprimidos e cada caixa contém quatro compartimentos de doses.

Um lembrete amigável e repetitivo notifica o horário, a data e a dose diária a ser tomada. Ao pressionar o botão de alarme reconhecido, o sistema informa quando a próxima toma deve ser feita [16].

A Tabela 2.1 resume as características dos dispensadores apresentados anteriormente. Através da tabela é possível verificar que não existem dois produtos com todas funcionalidades, obrigando assim ao utilizador escolher o produto que mais se adequa a si, abrindo mão de algumas características.

Tabela 2.1 Tabela Comparativa das diferenças entre os dispensadores automáticos de medicamentos

	Alarmes sonoros ou visuais	**Notificações por SMS, e-mail ou** aplicação	**Comunicação com os cuidadores ou** profissionais de saúde	Permite ser controlados e/ou programados a longa distância
MedMinder	✓	✓	✓	✗
Philips Lifeline Medication Dispenser	✓	✓	✓	✗
Hero	✓	✓	✓	✗
EMMA	✗	✓	✓	✓
MEDCENTER Low Profile Pill System	✓	✗	✗	✗

2.2.2. Vantagens e desvantagens dos dispensadores automáticos de medicamentos

Tomar os medicamentos pode ser uma tarefa facilmente esquecida, principalmente para aqueles com vidas mais ocupadas, horários variados ou idade avançada. Um dispensador de medicamentos irá lembrar-nos das horas, das doses e que medicação deve ser tomada, evitando assim que sejam esquecidos ou tomados incorretamente.

Algumas etiquetas presentes nas embalagens de medicamentos são impressas em letras pequenas e podem ser difíceis de ler. Medicamentos com nomes parecidos podem levar também a alguma confusão na hora da administração,

especialmente em pacientes com problemas de visão. Com um dispensador automático, os comprimidos podem ser carregados por um longo período, economizando tempo e diminuindo a frustração.

Os dispensadores reduzem a carga de trabalho dos profissionais de saúde e dos cuidadores. Também oferecem aos pacientes a liberdade de não terem de depender de outras pessoas para a toma dos seus medicamentos. Os pacientes e os cuidadores poderão preocupar-se com outros aspetos em suas vidas sem a necessidade de se preocuparem com medicamentos.

Deixar as embalagens de comprimidos à solta pode trazer perigos, pois crianças podem facilmente ingeri-los. Depois que os comprimidos são carregados no dispensador, o acesso a eles apenas é permitido nas horas pré-definidas, diminuindo significativamente o risco de crianças ou até mesmo outras pessoas consumirem os comprimidos acidentalmente.

Os dispensadores também possuem algumas desvantagens. Grande parte dos dispensadores de medicamentos existentes permitem que o usuário organize os medicamentos e defina os alarmes. Qualquer erro no processo de organização de medicamentos pode levar a erros de medicamentos, podendo causar graves consequências.

Com base nos produtos descritos acima, conclui-se que nem todos os dispensadores existentes são fáceis nem rápidos de configurar. Os processos até que o equipamento esteja em funcionamento devem ser repetidos sempre que um parâmetro, como data, hora ou tipo de medicamento seja alterado. Estes processos podem tornar-se difíceis para o utilizador, fazendo com que possa haver a necessidade da presença de um supervisor sempre que ocorram alterações de algum parâmetro.

2.3. Serviço de Web Hosting

Os serviços de web hosting são serviços que fornecem armazenamento para um site ou aplicação em um servidor na internet. Estes serviços tornam possível o acesso ao site ou aplicação por outros dispositivos ligados à internet. Os serviços de web hosting podem guardar páginas, ficheiros, informações, imagens, vídeos e outros diversos conteúdos [17].

De modo a ter acesso aos ficheiros na web estes necessitam estar guardados em um servidor. Um servidor é um computador físico que permite a conexão entre utilizadores da web e o site. O servidor funciona sem interrupções para que o site esteja sempre disponível. Uma breve ilustração deste processo encontra-se representado na Figura 2.4.

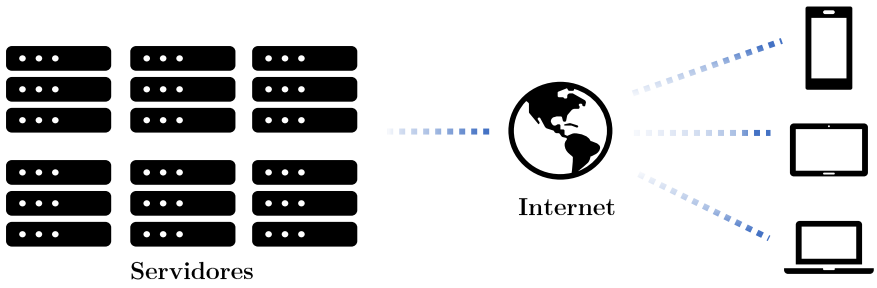

Figura 2.4: Serviço de web hosting.

Estas soluções de alojamento são categorizadas com base na quantidade de espaço necessário para o armazenamento dos ficheiros e da largura de banda. A largura de banda é a quantidade de dados ou recursos que serão transferidos através do site.

A maioria dos *web hosts* oferecem diferentes tipos de alojamento para atender às necessidades de diferentes utilizadores. Algumas dessas opções serão descritas a seguir.

Shared hosting (em português alojamento compartilhado) é a melhor opção para pequenos sites. Como o nome sugere, o utilizador compartilhará o mesmo servidor com outros sites, o que significa também uma compartilha de recursos.

O **VPS hosting** (Virtual Private Server, traduzido para o português Servidor Virtual Privado) permite dividir um servidor com outros utilizadores com a diferença que o *web host* dedica uma parte desse servidor para o nosso site.

Cloud hosting, ou alojamento em nuvem, permite armazenar dados através de um servidor em nuvem. Ao contrário do shared e VPS hosting, o cloud hosting não possui um servidor físico. Vários computadores são interconectados através da internet, formando uma rede virtual de servidores.

Dedicated hosting (em português Servidor dedicado) permite que o website tenha um servidor exclusivo. Com este tipo de de alojamento o site tem acesso a todos os recursos disponíveis.

O serviço de web host que foi usado no projeto é fornecido através do site 000webhost. O 000webhost possui um alojamento gratuito limitado, isto inclui o alojamento de um número limitado de sites, de espaço em disco e de largura de banda.

Este alojamento gratuito suporta a grande parte das linguagens de programação como o PHP, MySQL, HTML e muitas outras. Este alojamento grátis permite também coisas como [18]:

- Blogs em WordPress
- Sites de críticas e avaliações
- Sites pessoais
- Pequenos negócios
- Projetos escolares e universitários
- Pequenas lojas online

2.4. Linguagem PHP e Aplicações Web

PHP é um acrónimo para *Hypertext Preprocessor* cujo nome original era Personal Home Page [19]. PHP é normalmente usado para a comunicação no lado do servidor (server-side).

Como o nome indica server-side, em português "lado do servidor", significa que toda a ação acontece do lado do servidor [20]. Os scripts server-side são executados do lado do servidor em vez do cliente de modo a entregar respostas às ações do usuário.

Suponhamos que temos um computador que será o cliente, do outro lado temos um servidor e no meio encontra-se toda a infraestrutura da internet necessária para trazer todas as informações do servidor para o computador (cliente), como representado na Figura 2.5 [21].

Figura 2.5: Estrutura linguagem PHP. Inspirado em [21].

No servidor encontram-se os módulos PHP. O servidor precisa de ser especialmente preparado para executar estes módulos PHP e fornecer os resultados ao cliente. Para isso, ele usa uma ferramenta chamada apache para transformar uma máquina num servidor. O apache tem assim uma funcionalidade denominada por interpretador PHP. A partir do momento que interpretador começa a funcionar todo o código dentro de um ficheiro PHP começa a ser executado. Esse processamento vai gerar um ficheiro HTML. Nesse ficheiro apenas irão existir *tags* HTML ou qualquer outra tecnologia cliente-side [21]. O processo encontra-se ilustrado na Figura 2.6.

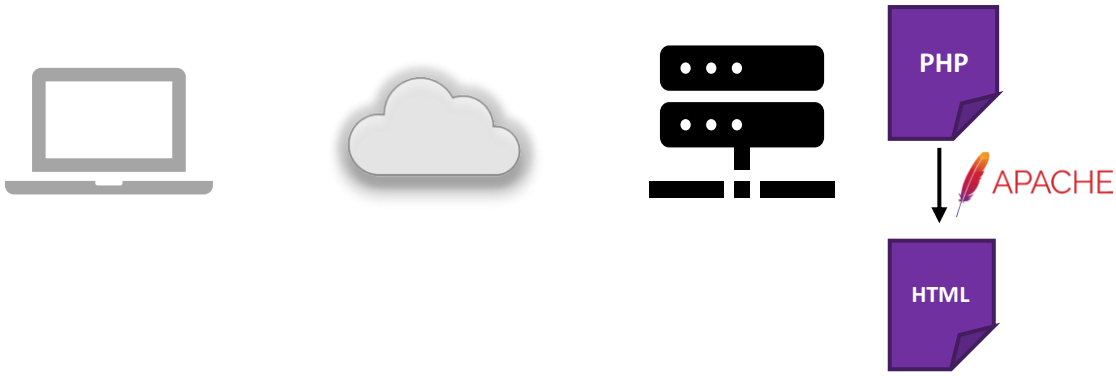

Figura 2.6: Comunicação PHP parte I. Inspirado em [21]. Símbolo Apache [22].

Esse ficheiro não vai ser guardado no servidor, ele vai ser diretamente enviado para o cliente que o solicitou, como representado na Figura 2.7. A partir desse momento o navegador do cliente passa a interpretar o HTML gerado no servidor [19].

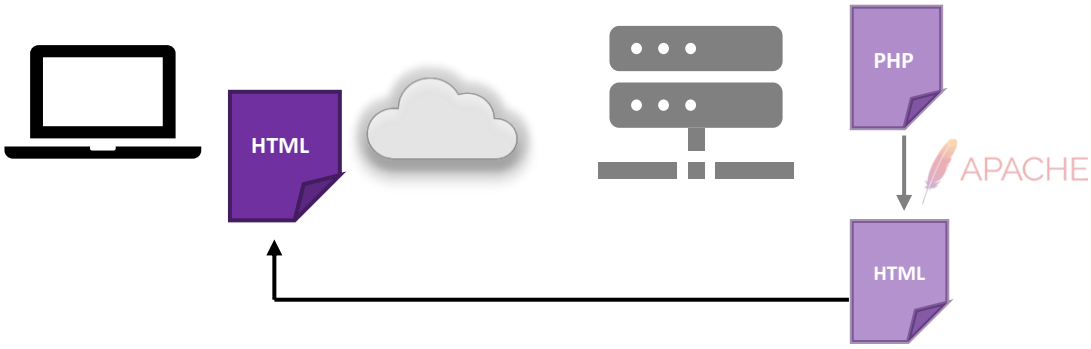

Figura 2.7: Comunicação PHP parte II. Inspirado em [23]. Símbolo Apache [22].

A comunicação client-side funciona de modo oposto, pois tudo acontece do lado do cliente. Quando o cliente solicita algo, o servidor responde enviando os códigos-fontes que são executáveis pelo navegador [23].

O servidor armazena ficheiros, por exemplo HTML. Se o cliente desejar visualizar um ficheiro HTML o servidor oferece uma cópia desse ficheiro. A partir do momento em que a cópia do ficheiro HTML se encontra do lado do cliente o navegador começa a processar esse ficheiro. Neste caso, o cliente tem mais trabalho, ele tem a função de fazer uma solicitação ao servidor, receber e processar o ficheiro. A única função do servidor é fornecer o ficheiro. Por isso esta tecnologia é conhecida como tecnologia cliente-side ou "lado do cliente" em português.

Uma aplicação web é composta por sistemas armazenados em um servidor remoto e utilizados através de um navegador ou através da internet [24]. A aplicação web necessita de um servidor web para gerir as solicitações do cliente, um servidor de aplicação para realizar as tarefas solicitadas e, ocasionalmente, de uma base de dados para armazenar qualquer informação necessária.

Uma aplicação web fornece ao utilizador a capacidade de interação com suas funcionalidades através de diferentes interfaces. As interfaces do usuário são as únicas partes visíveis para o utilizador e uma das partes mais importantes de qualquer aplicação web.

As aplicações web são compostas por uma ou mais páginas web que geralmente são criadas através de várias linguagens de programação e script, por exemplo o PHP no lado do servidor, e o JavaScript no lado do cliente. Estas páginas contêm uma combinação de conteúdos estáticos e/ou dinâmicos. Os usuários podem aceder a essas páginas usando os seus navegadores web. As aplicações web

podem ser executadas a partir de um servidor HTTP (web host) ou no dispositivo do utilizador [24].

HTTP, Hypertext Transfer Protocol, protocolo responsável por efetuar o tratamento de pedidos (ou *web requests).* Um web request é uma solicitação da web em que uma mensagem é transmitida entre o cliente ou navegadores da web para os seus servidores [25]. O servidor então recupera o conteúdo solicitado pelo cliente que irá resultar nas páginas web ou recursos interativos que serão exibidos. Uma representação básica do protocolo HTTP encontra-se ilustrada na Figura 2.8.

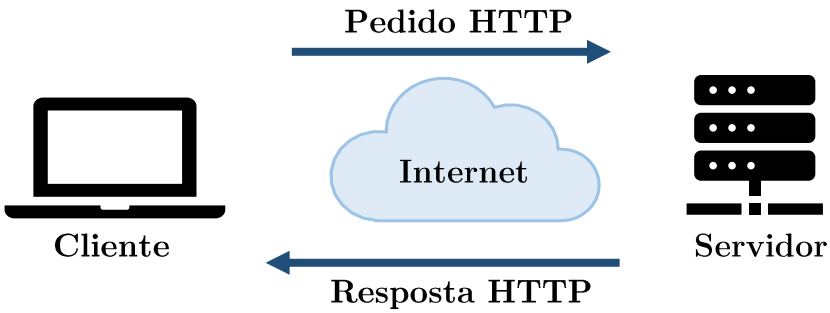

Figura 2.8: Protocolo HTTP.

O protocolo de comunicação HTTP suporta a comunicação interage com o cliente e o servidor, sendo a base para a comunicação de dados na World Wide Web [26]. O cliente irá enviar uma mensagem de requisição para o servidor e o servidor irá retornar uma mensagem de resposta para o cliente. O protocolo HTTP possui um conjunto de métodos de requisição. Estes métodos são responsáveis por indicar a ação que se pretende executar. Alguns destes métodos de requisição são [26], [25], [27]:

- **GET:** A requisição GET é efetuada quando o cliente pretende obter algum dado de algum recurso específico.
- **POST**: O POST tem como objetivo criar ou atualizar os recursos especificados no servidor.
- **PUT:** Este método permite que todas as representações do recurso destino sejam substituídas por um novo recurso no servidor.

- **PATCH:** Este método apenas modifica uma parte específica do recurso.
- **DELETE:** Remove um recurso específico.

Um serviço web consiste numa serie de componentes que permitem que diferentes máquinas interajam e colaborem entre si através de uma rede, de forma a alcançarem um objetivo comum [28]. Basicamente, um serviço web permite que os recursos de uma aplicação estejam disponíveis sobre a rede, para outras aplicações, de forma normalizada [29].

Os serviços web permitem a comunicação entre aplicações de uma maneira independente do sistema operacional e de linguagem de programação [30]. Adicionalmente, os serviços web não fornecem aos usuários uma interface interativa como nas aplicações Web, isto porque estes serviços permitem apenas a partilha de informação e foram destinados para serem usados apenas entre aplicações e distribuídas por distintas máquinas.

2.5. Base de Dados

2.5.1. phpMyAdmin

phpMyAdmin consiste numa aplicação do tipo Web que permite a administração das bases de dados MySQL. Este software permite a criação e remoção de bases de dados assim como a criação e manuseamento de tabelas, scripts e outros recursos existentes nas bases de dados. Algumas das funcionalidades e características do phpMyAdmin são as seguintes [31], [32]:

- Uma interface web, como ilustrado na Figura 2.9, para a interação com o utilizador.
- Suporta grande parte das funcionalidades MySQL e mysqli (uma extensão do MySQL)
- Permite administrar vários servidores
- Permite criar, editar, apagar e renomear bases de dados, tabelas, campos e informações
- Permite executar SQL-statements

- Importação de dados CSV ou SQL e exportação de dados no formato CSV, XML e Latex

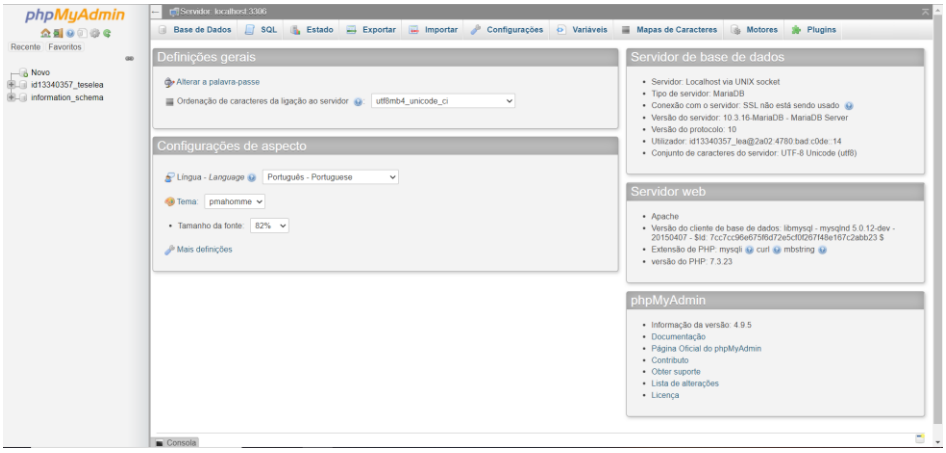

Figura 2.9: Interface web phpMyAdmin.

2.5.2. SQL e MYSQL

O SQL, do inglês Structured Query Language, consiste numa linguagem utilizada para criar e gerar bases de dados, incluindo a manipulação da informação existente, nomeadamente, criar, alterar, apagar e pesquisar informação [33].

A linguagem SQL é dividida em quatro subconjuntos de acordo com o tipo de operação a ser executada na base de dados. A descrição destes subconjuntos é realizada a seguir.

DDL -Data Definition Language

É o subconjunto que lida com as descrições do esquema da base de dados e é usado para criar e modificar a estrutura dos objetos da base de dados. Exemplos de comandos DDL [34]:

- **CREATE:** utilizado para criar um objeto na base de dados, por exemplo uma tabela. A Figura 2.10 ilustra a criação de uma nova base de dados com o nome escola, em que se utiliza o comando CREATE. A Figura 2.11 mostra a criação de uma tabela com o nome alunos e os respetivos campos a serem inseridos nessa tabela.

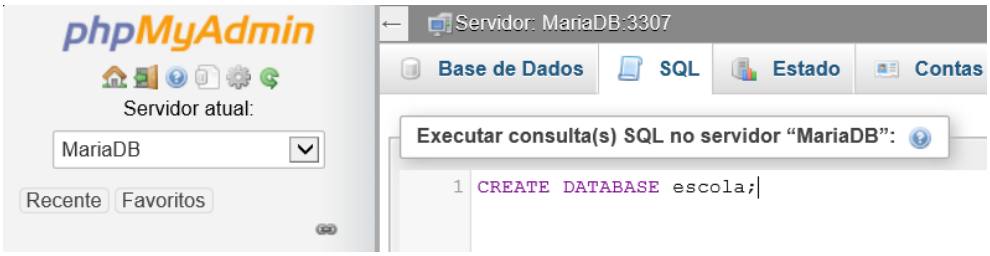

Figura 2.10: Criação de uma nova base de dados.

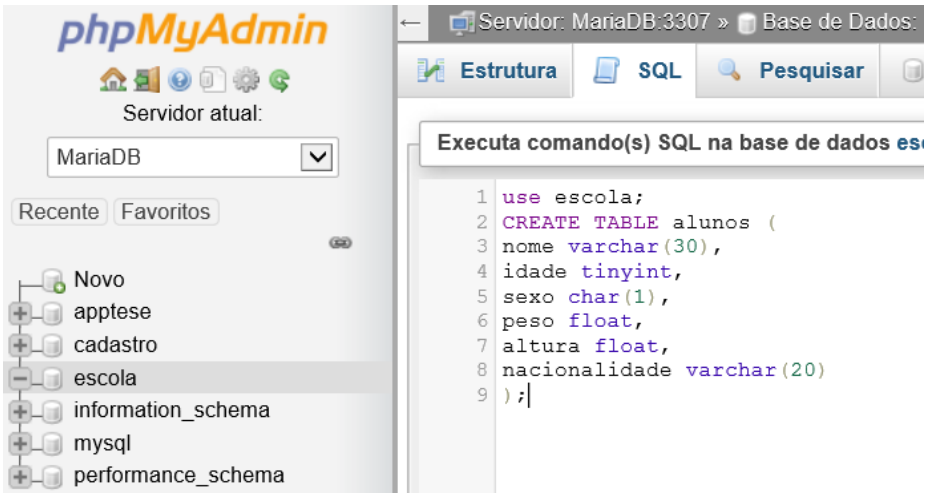

Figura 2.11: Criação da tabela alunos.

- **DROP:** utilizado para eliminar objetos da base de dados.
- **ALTER:** usada para alterar a estrutura da base de dados.

DML- Data Manipulation Language

Data Manipulation Language ou linguagem de manipulação de dados, é o subconjunto de linguagem SQL que permite operações sobre as bases de dado, tais como, instruções de inserção, modificação, remoção e consulta de informações [35]. Alguns comandos SQL deste subconjunto são:

- **INSERT**: utilizado para inserir um registo. A Figura 2.12 demonstra a inserção de dados na tabela alunos, criada anteriormente usando o comando INSERT.

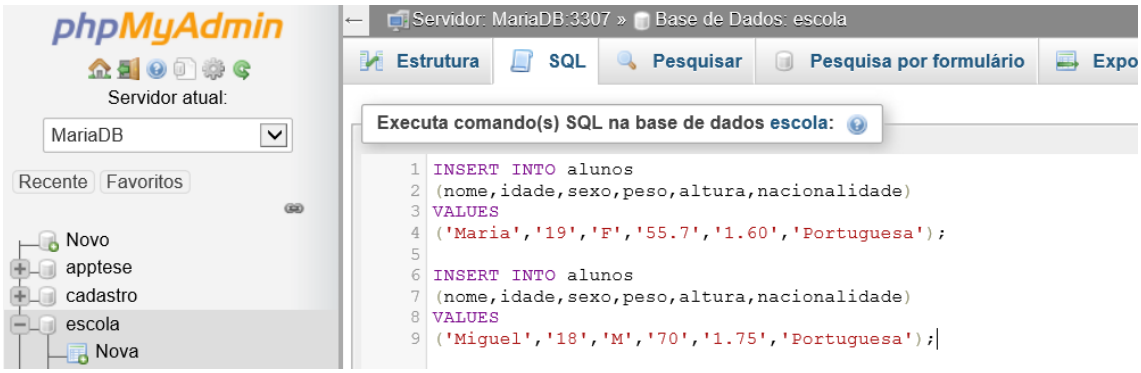

Figura 2.12: Inserção dados na tabela alunos.

- **UPDATE**: utilizado para mudar valores de dados em uma ou mais linhas numa tabela já existente.
- **DELETE**: utilizado para remover linhas existentes numa tabela.

DCL - Data Control Language

Data Control Language ou Linguagem de Controle de Dados, lida principalmente com os direitos, permissões e licenças para controlar quem tem acesso para ver e manipular os dados na base de dados. Dois comandos deste subconjunto são [34]:

- **GRANT:** fornece privilégios a um usuário para executar operações.
- **REVOKE:** remove a permissão de um utilizador de executar operações.

DQL - Data Query Language

Este subconjunto possui apenas um comando, mas é considerado como a parte do SQL mais usada. Este comando é o **SELECT** que permite a realização de consultas a dados pertencentes a uma tabela [34], [35]. A Figura 2.13 ilustra uma query que realiza uma pesquisa na base de dados escola. A query "SELECT * FROM alunos WHERE nome like 'Maria'" procura na tabela alunos, todos os registos em que o nome seja Maria. O resultado desta pesquisa encontra-se representada na Figura 2.14.

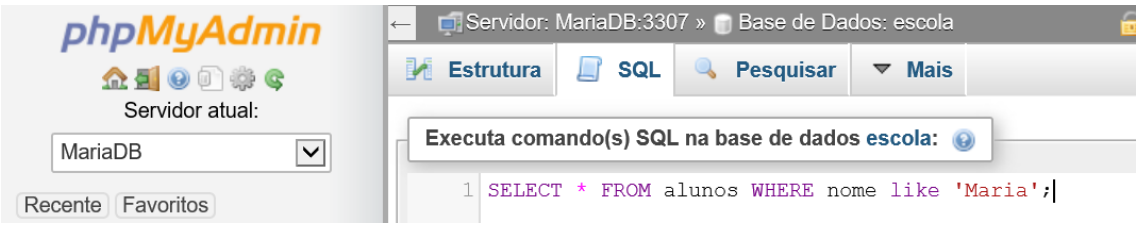

Figura 2.13: Comando SELECT para pesquisa de aluno.

Figura 2.14: Resultado do comando SELECT para pesquisa de aluno.

Um *Data Base Management System* (DBMS), em português sistema de gestão de banco de dados, são aplicações de software responsáveis pela gestão de dados. O MySQL é um DBMS que utiliza a linguagem SQL como interface. MySQL é também uma ferramenta gratuita.

2.6. Android Studio

O Android Studio consiste num ambiente de desenvolvimento para a criação de aplicações para Android. Este programa permite o funcionamento das aplicações em qualquer dispositivo Android. Algumas funcionalidade do Android Studio são [36]:

- Sistema de complicação Gradle. O Gradle é um sistema de compilação que junta recursos de outros sistemas de compilação e os junta em um só [37].
- Um emulador que simula dispositivos android no computador, de modo que seja possível realizar testes a aplicação em desenvolvimento. O emulador permite que não seja necessário um dispositivo físico [38] .
- Ferramentas que permitem testes e deteção de problemas como de desempenho ou incompatibilidade de versões [36].

2.6.1. Estrutura do projeto Android

Um projeto android possui um ou mais módulos com ficheiros código-fonte e de recursos. Os módulos podem ser do tipo: módulos de apps Android, módulos de biblioteca, módulos do Google App Engine.

Cada módulo da aplicação android contém obrigatoriamente as seguintes pastas:

- **Manifests**: contém o ficheiro AndroidManifest.xml. O ficheiro AndroidManifest.xml possui informações importantes sobre a app, para as ferramentas de compilação e para o sistema operacional android [39]. Neste ficheiro é essencial a declaração do nome do pacote da app, os componentes da app incluindo todos os serviços, *Broadcast receivers,* atividades e permissões que a app pode precisar para acessar partes protegidas do sistema [40].
- **Java:** Contém os ficheiros de código-fonte Java [36]**.**
- **Recursos:** Possui os recursos que não sejam código, como por exemplo os layouts XML.

A Figura 2.15 resume a estrutura de um projeto desenvolvido no ambiente Android Studio.

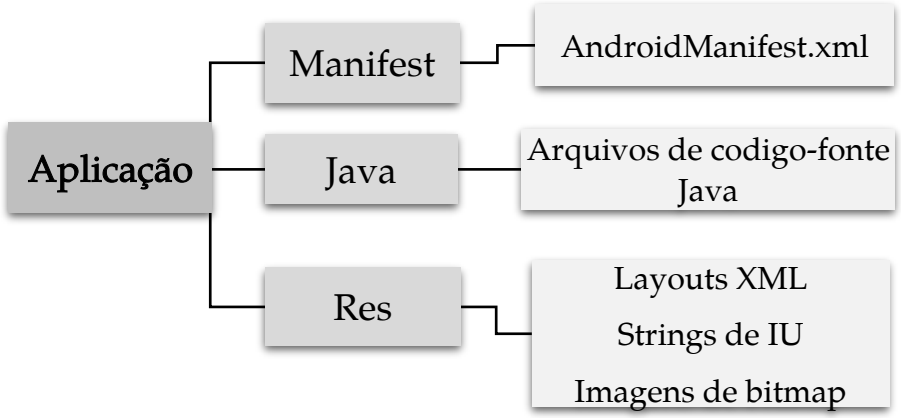

Figura 2.15 Estrutura de um projeto android Studio.

2.7. Kit de simulação Staudinger

O kit de armazenamento de alto nível simula um sistema automático de armazenamento de alto nível usado em muitos ambientes industriais. O kit está dividido em cinco prateleiras e cada prateleira esta subdividida em dez compartimentos, ou seja, o kit original é constituído por 50 posições livres. O modelo possui um dispositivo que permite operar no armazém, movível na direção do X e duas estações de carga e descarga. O dispositivo operacional de armazém possui uma gaiola movível no eixo do Z e um porta-paletes movível no eixo Y [41].

A Figura 2.16 representa uma imagem do kit Staudinger e a identificação de algumas zonas e elementos.

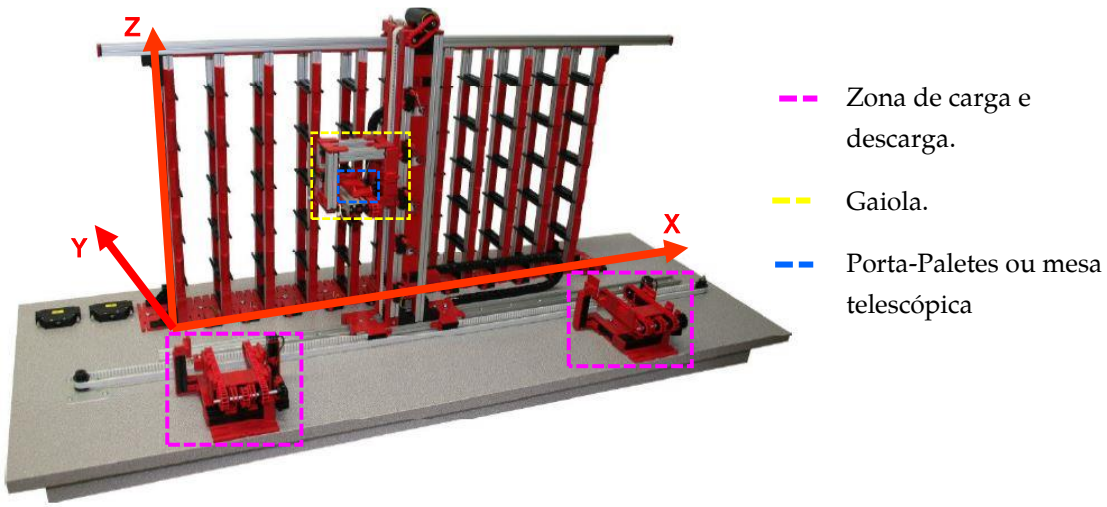

Figura 2.16: Kit de Simulação Staudinger.

O kit permite simular objetos a serem armazenados e retirados de cada prateleira. É composto por sensores que permitem o armazenamento e recuperação das paletes. Quando a zona de carga se encontra ocupada com uma palete, a mesa telescópica é movida para cima, a mesa pega na palete e é detetado através de um sensor. A palete é então levada para a prateleira designada através de uma rota otimizada por distancia, movendo-se simultaneamente nas direções X e Z.

O software de controlo necessita de registar a ocupação de um compartimento da prateleira para garantir um transporte rápido, mas acima de tudo para garantir um processo de armazenamento seguro. Para se realizar a retirada de um produto que se encontra num dos compartimentos das prateleiras, as etapas do processo são executadas de ordem inversa [42].

Abaixo encontra-se a descrição dos componentes que pertencem aos dados técnicos.

Componentes

Dados Técnicos:

Fonte de alimentação dos sensores e atuadores	:	24 V DC
Sensores:		
Interruptores de luz de reflexão	:	1
Barreira de luz unilateral	:	2
Chaves Mecânicas	:	23
Atuadores:		
Motores bidirecionais	:	5
Relay	:	3
Requisitos do sistema de controle :		
Inputs digitais	:	26
Outputs Digitais	:	11
Dimensões (largura × Comprimento ×Altura)	:	470×1260×600 mm

CAPÍTULO 3

DESENVOLVIMENTO DO SISTEMA

Na secção 3.1 são definidos os requisitos funcionais, para que o sistema cumpra com as exigências necessárias para uma correta gestão de medicamentos.

As restantes secções são reservadas para a descrição detalhada de todo o processo referente à construção do sistema. São também descritos o modo de funcionamento de todos os componentes constituintes do sistema.

3.1. Requisitos funcionais

Este sistema tem como propósito fornecer ao utilizador uma ferramenta de gestão de medicamentos que seja capaz de não só lembrar ao utente das horas da sua medicação, mas que também seja capaz de fornecer alguma informação ao cuidador.

Tendo em conta o público-alvo, é imperativo que a aplicação seja simples, com uma interface clara e de fácil manuseamento. O utilizador não deve perder muito tempo a tentar compreender o funcionamento do sistema e nem em que secções deve entrar para realizar uma ação.

A Figura 3.1 exemplifica um esquema básico do que se pretende que seja o funcionamento do sistema.

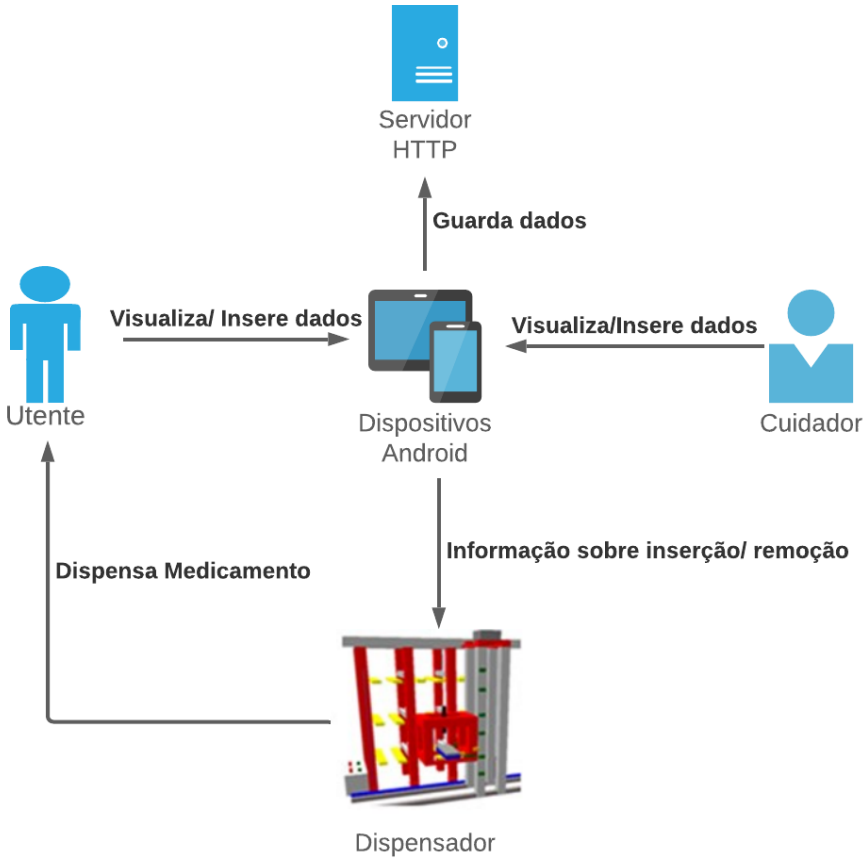

Figura 3.1: Funcionamento do sistema.

De acordo as especificações previamente citadas sobre dispensadores automáticos de medicamentos, e de modo que a solução proposta cumpra com estas exigências, foram definidos alguns requisitos que se encontram descritos na Tabela 3.1. O projeto a desenvolver deve ter como objetivo o cumprimento destes requisitos e os mesmos devem incluir tarefas básicas como inserção, remoção, lembretes de medicamentos e outras tarefas que complementam e auxiliam o bom funcionamento do sistema.

Tabela 3.1: Requisitos funcionais necessários.

Requisito	Descrição
R1	Registo do utilizador
R2	Início de sessão do utilizador
R3	Visualização da lista de medicamentos do utilizador
R4	Inserção e remoção de novos medicamentos na base de dados
R5	Inserção de novos medicamentos no dispensador
R6	Alarmes à hora da medicação
R7	Remoção da medicação da célula à hora pretendida
R8	Visualização da agenda/histórico da medicação

O primeiro e segundo requisitos, R1 e R2, têm como finalidade garantir que qualquer pessoa possa usufruir e gerir das funcionalidades do sistema de acordo com as suas próprias necessidades. Se o sistema não tivesse um método de identificação de utilizador, seria impossível este gerir e fornecer as informações necessárias e respetivas de cada utilizador.

O terceiro e quarto requisito, R3 e R4, permitem não só ao utilizador, mas também ao cuidador, verificar que medicamentos já foram inseridos no sistema. Apenas é possível responder ao problema de esquecimento e gestão incorreta de um medicamento se estes estiverem inseridos no sistema.

O requisito R5 responde ao problema de erro de dosagem e da toma do medicamento incorreto. Ao ser inserido a dosagem e medicamento correto para cada toma num compartimento do dispensador, não se corre o risco de ingestão de uma dose superior ou inferior da dose correta.

O sexto requisito visa lembrar o utilizador da toma de um medicamento na hora devida. Apesar do horário de cada medicamento se encontrar no sistema e ser visível ao utente, o utilizador pode não se lembrar de o ingerir na hora adequada, daí a necessidade dos alarmes.

O sétimo requisito complementa o quinto requisito em relação ao problema de erro da dosagem de medicamento. Dado que a disponibilização de um

medicamento é automática, o utente saberá que terá de ingerir aquela quantidade de medicação, evitando que tome uma dosagem acima ou abaixo da recomendada.

O oitavo e último requisito fornece ao utente e ao cuidador uma visão detalhada da agenda e do histórico da sua medicação. Com o histórico, o cuidador poderá verificar se algum medicamento não foi administrado e agir se necessário. A agenda permite que o utente tenha uma visão detalhada das suas futuras tomas.

3.2. Arquitetura do projeto

O projeto será composto por três grandes componentes: os serviços de web hosting, a aplicação android e o dispensador. Os serviços de web hosting serão responsáveis pelo armazenamento via web dos módulos PHP e da base de dados do projeto. A aplicação android será o componente que permitirá a comunicação do paciente com o dispensador e será realizado no ambiente de desenvolvimento Android Studio. O último componente é o dispensador que servirá para o armazenamento da medicação.

A Figura 3.2 ilustra um esquema representativo da metodologia de desenvolvimento para o dispensador automático de medicamentos proposto no projeto, assim como as interações entre os diferentes componentes do projeto.

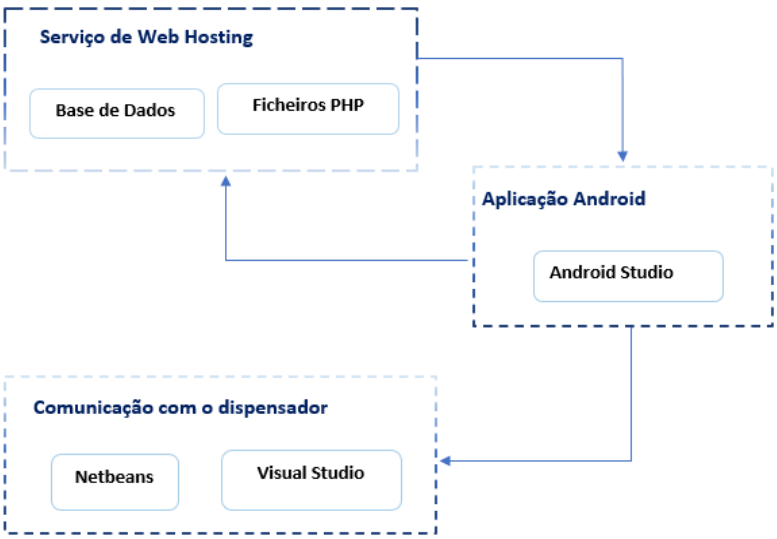

Figura 3.2: Esquema representativo da metodologia de desenvolvimento do projeto.

3.3. Diagrama de sequência UML do sistema

Os diagramas de sequência UML (*Unified Modelling Language*), são diagramas usados para a representação de uma sequência de processos e as mensagens trocadas entre os componentes de um sistema com a finalidade do desempenho com sucesso das suas funções [43][44].

Ocorrem inúmeros processos e comunicações quando o sistema desenvolvido se encontra em operação. Deste modo e de maneira a facilitar a compreensão geral destas comunicações, foi criado o diagrama de sequência UML representado na Figura 3.3.

O diagrama é constituído por um símbolo de ator que representará o utilizador do sistema e três símbolos de objetos, a aplicação que representara a aplicação android, a base de dados e o dispensador.

O diagrama inicia com uma tentativa de início de sessão do utilizador na aplicação android. A aplicação android envia a informação referente a tentativa de início de sessão para o web host e este verifica se os dados se encontram na base de dados. Caso os dados existam na base de dados, o início de sessão é realizado com sucesso e o menu principal é apresentado ao utilizador.

No diagrama estão representados três blocos opt. Os blocos opt representam opções que o utilizador pode realizar no sistema. O primeiro bloco opt descreve o processo caso o utilizador pretenda visualizar os seus medicamentos. O segundo bloco descreve processo de inserção de um novo medicamento no sistema. O terceiro e último bloco opt, é reservado para a descrição dos processos por detrás da visualização do histórico, agenda ou dos dados do utilizador.

Existe também um bloco alt, que representa as duas alternativas que o utilizador tem quando chega a hora da administração de um medicamento e a notificação é exibida.

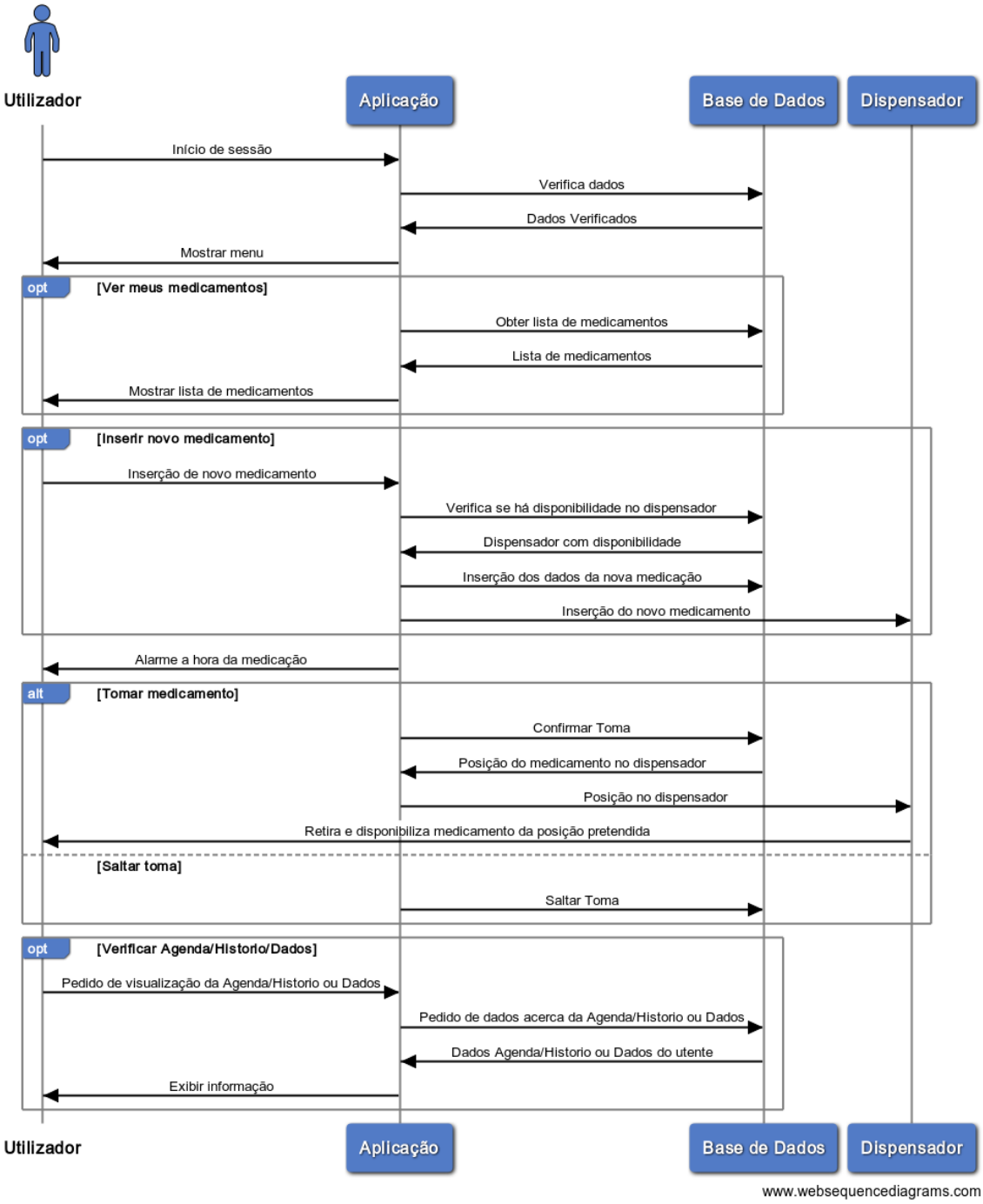

Figura 3.3: Diagrama de sequência funcionamento geral da aplicação. Criado em [45].

3.4. Desenvolvimento no Ambiente Android Studio

Como referido previamente, o ambiente utilizado para a criação da aplicação android e da interface de utilizador foi o Android Studio. Neste projeto, a aplicação android será o componente responsável pela interação entre o paciente e os restantes componentes. Na aplicação android, o utilizador terá a capacidade de inserir, retirar e fazer a gestão da sua medicação.

A aplicação android também permitirá que o cuidador possa iniciar sessão, com os dados do utente, para verificar o estado das administrações do utente.

Um layout no Android Studio é a estrutura de uma interface do utilizador. Esta interface é criada usando a hierarquia de *view* e *view groups,* como apresentado na Figura 3.4. Uma *view* mostra algo que torna possível a interação com o utilizador.

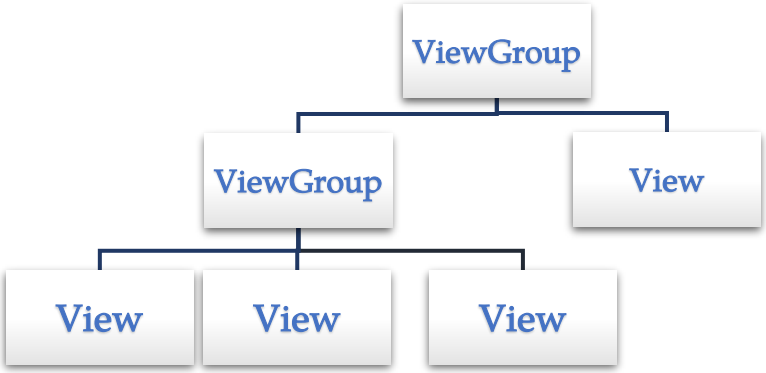

Figura 3.4: Hierarquia View no ambiente Android Studio.

As *views* são normalmente denominadas de *widgets* e podem conter uma serie de subclasses como *buttons*, *TextViews* ou *EditText*. As ViewGroup são frequentemente denominadas de "layouts" e podem possuir layouts do tipo LinearLayout ou ConstrainLayout. Na Figura 3.5 encontra-se representada uma ViewGroup em forma de código, este layout é o activity_main.xml e é primeiro que o utilizador tem acesso. Como se pode observar, neste ViewGroup existe um layout do tipo LinearLayout e o View possui algumas subclasses como ImageView e EditText.

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:orientation="vertical"
    tools:context=".MainActivity"
    android:background="#ffffff">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="257dp"
        android:layout_height="145dp"
        android:layout_gravity="center"
        android:foregroundGravity="center"
        android:paddingTop="50dp"
        app:srcCompat="@drawable/logo_tese" />
    <EditText
        android:id="@+id/name"
        android:layout_width="310dp"
        android:ems="10"
        android:layout_height="43dp"
        android:layout_marginTop="60dp"
        android:layout_marginLeft="40dp"
        android:hint="Nome de Utilizador"
        android:inputType="textPersonName"
       />
```

Figura 3.5: Layout início de sessão formato código.

A Figura 3.6 representa o mesmo layout, mas em forma de design. Este layout possui dois campos para serem preenchidos pelo utilizador, estes campos são: o nome de utilizador e a palavra-passe.

Figura 3.6: Layout de início de sessão formato design.

O layout possui também dois botões: um para o início de sessão e outro para a criação de uma conta nova. Esta view permite satisfazer o requisito R2 que tem como objetivo o início de sessão do utilizador.

Depois de o utilizador inserir os dados no layout e clicar no botão "Iniciar Sessão", estes dados são verificados na base de dados. Caso o início de sessão seja aprovado, o layout activity_main_menu.xml, representado na Figura 3.7, que possui o menu principal, é apresentado.

Figura 3.7: Layout activity_main_menu.xml.

O menu apresentado neste layout é composto por quatro opções que são:

- **Meus Medicamentos** – Esta opção satisfaz o requisito R3 que permite a visualização dos medicamentos no sistema. Nesta opção o utilizador tem acesso à lista dos seus medicamentos (layout menu_medicamentos.xml, Figura 3.8 (1)). Esta opção permite a inserção ou remoção

de medicação e dos dados que permitem a sua gestão. O layout desta ação encontra-se representado na Figura 3.9. Se o utilizador já tiver previamente inserido medicamentos no sistema, a opção "Meus Medicamentos" também permite a visualização dos dados desses medicamentos. Na Figura 3.8 imagem (2), encontra-se um exemplo de visualização de uma lista com dois medicamentos.

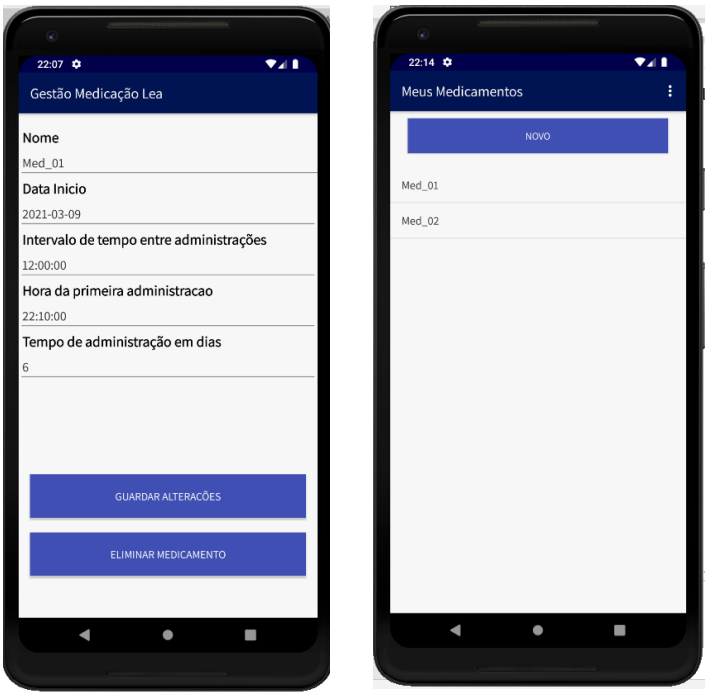

Figura 3.8: (1) Layout visualização dados medicamento, (2) Layout Lista de medicamentos

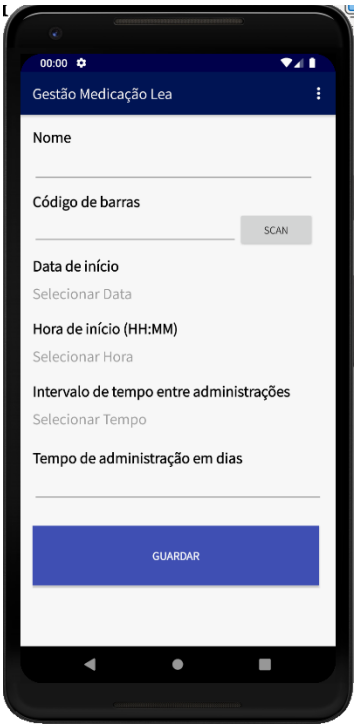

Figura 3.9: Inserção de novo medicamento na aplicação.

A Figura 3.9 apresenta o layout para a inserção de um novo medicamento. Este processo permite satisfazer o requisito R5 que tem como finalidade e inserção de um novo medicamento no sistema. Este layout apenas é apresentado ao utilizador se o dispensador possuir alguma posição livre, caso contrário, uma mensagem é apresentada ao utilizador informando que o dispensador se encontra cheio. Nesta secção existem os seguintes campos que devem ser preenchidos com especial atenção:

- **Nome:** Onde o utilizador deve inserir o nome da medicação.
- **Código de barras:** Caso o utilizador deseje inserir o código de barras do medicamento. Para tal, basta carregar no botão "scan" e irá aparecer a câmara do dispositivo conforme apresentado na Figura 3.10.

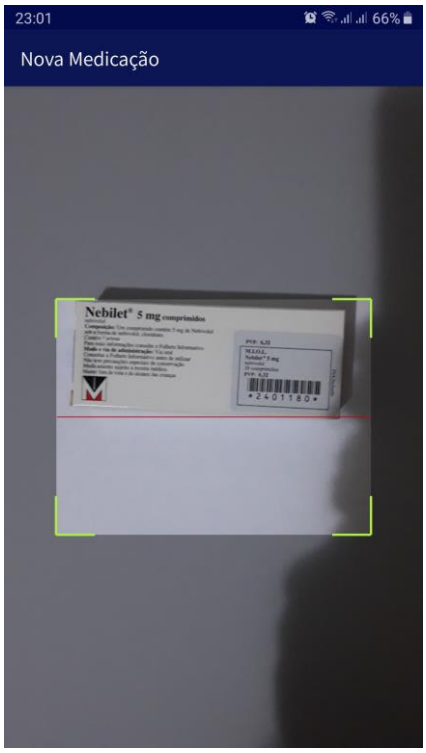

Figura 3.10: Deteção do código de barras através da câmara do dispositivo.

O utilizador só precisa aproximar a câmara do dispositivo ao código de barras do artigo. A leitura é feita de imediato e irá aparecer uma caixa de diálogo com o resultado lido, como representado na Figura 3.11.

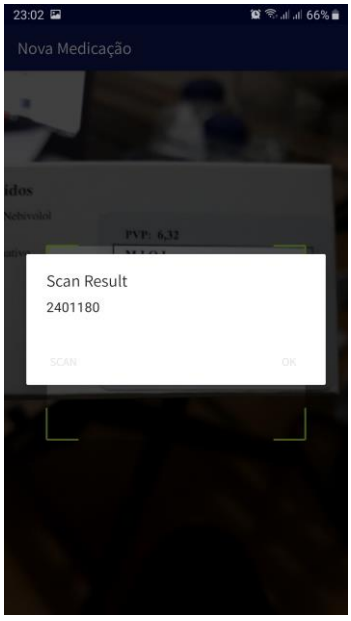

Figura 3.11: Caixa de diálogo com o resultado da leitura do código de barras.

A caixa de diálogo possui dois botões, “SCAN” e “OK”. O utilizador precisa de confirmar o número lido carregando no “OK” ou pode voltar a fazer scan se escolher a opção “SCAN”. Caso o utilizador confirme o número lido, o layout para a inserção dos medicamentos volta a aparecer, mas com campo de código de barras preenchido com o valor obtido, conforme representado na Figura 3.12.

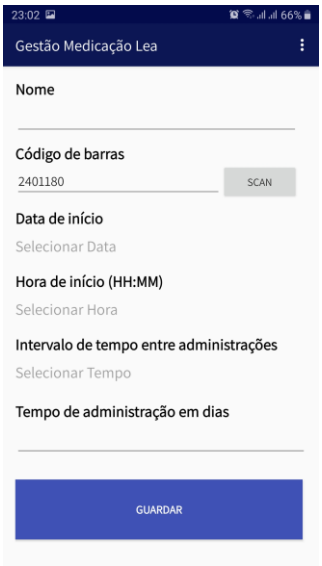

Figura 3.12: Inserção de medicamento com após leitura do código de barras.

· **Data início:** Aqui é introduzida a data da primeira toma da medicação no formato apresentado na Figura 3.13**.**

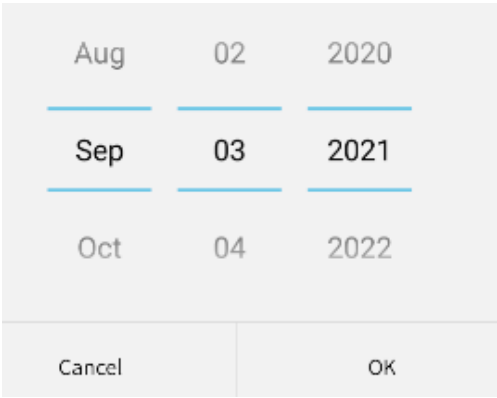

Figura 3.13: Seleção da data de início de um medicamento.

- **Hora de início:** Neste campo o utilizador introduz a hora de início do respetivo medicamento. A hora é inserida no formato hh:mm.
- **Intervalo de tempo entre administrações:** Como o nome indica, neste campo é inserido o intervalo de tempo em horas entre as administrações do medicamento.
- **Tempo de administração em dias:** Aqui pretende-se que seja colocado o tempo máximo de administração do medicamento em dias. Para não sobrecarregar demasiado a base de dados, o tempo máximo de administração do medicamento foi definido para 30 dias. Caso seja inserido um valor superior a 30, o sistema automaticamente reduz esse número para 30 e apresenta uma mensagem de erro, conforme a Figura 3.14. Se a medicação exigir um período superior a 30 dias, terá de ser feita uma nova inserção desse medicamento após o fim dos 30 dias iniciais.

Figura 3.14: Mensagem de erro caso tempo de administração superior a 30 dias.

Após a inserção correta dos dados do medicamento e do click no botão "GUARDAR", a aplicação android envia os dados para o web host. Estes dados são então calculados e inseridos na base de dados. A agenda também é criada com a data e hora de todas as tomas durante o número de dias inserido no campo "tempo administração em dias".

Sempre que se realiza um início de sessão com sucesso ou inserção de um novo medicamento, o sistema recebe a agenda da base de dados. Com os dados da agenda, o sistema verifica se existe a necessidade de criação de um novo alarme. Caso exista, o sistema cria os respetivos alarmes para lembrar os utentes da toma da medicação.

- **Agenda** – Esta opção disponibiliza ao utilizador a agenda detalhada da medicação do utente. É apresentado o nome de cada medicação para o dia selecionado assim como a hora que este deve ser administrado. As setas de seleção neste layout permitem avançar ou recuar um dia para a visualização do dia pretendido. Uma representação da agenda é representada na Figura 3.15.

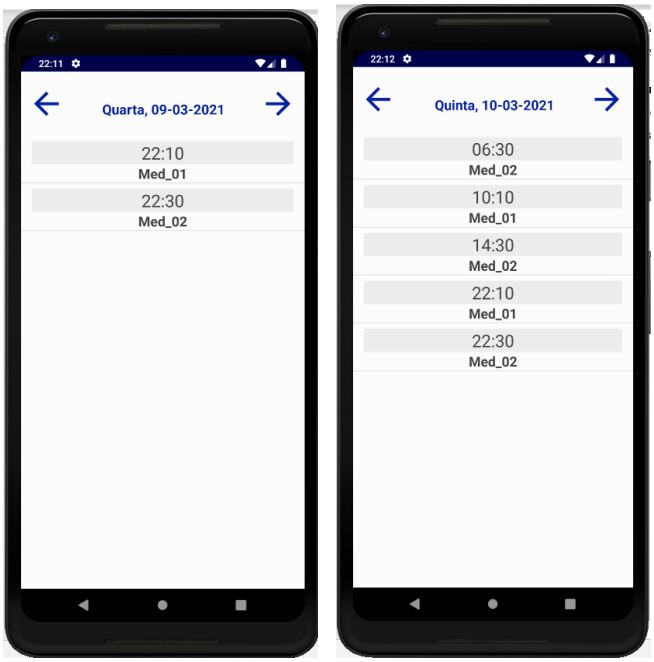

Figura 3.15: Layout Agenda.

- **Dados** – Permite ao utente visualizar os seus dados pessoais, como o exemplo da Figura 3.16.

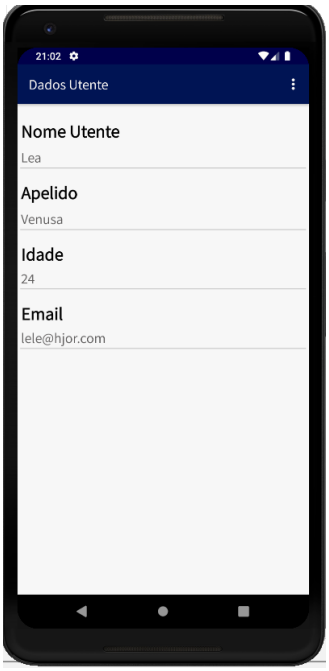

Figura 3.16: Visualização dos dados do utente na aplicação.

- **Histórico** – Apresenta o histórico da medicação do utente. A medicação pode ter dois estados: sem resposta em que a letra é apresentada a vermelho e administrada com a letra verde. Enquanto a opção "Tomar" não for selecionada, o estado do medicamento será sempre sem resposta. Caso o utente escolha tomar o medicamento, o estado já muda para "Administrada". Na Figura 3.17, encontra-se um exemplo em que o estado do medicamento "Med_01" se encontra a vermelho e com a mensagem "sem resposta", enquanto o medicamento "Med_02" possui o estado na cor verde e com a mensagem "Administrada".

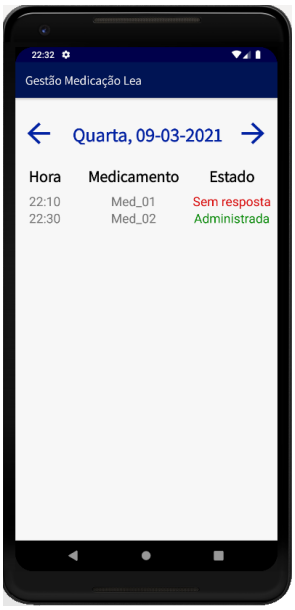

Figura 3.17: Histórico da medicação na aplicação.

3.4.1. Administração de medicamento

Como referido anteriormente, a agenda guarda todas as datas e horas de todas as tomas durante o número de dias inserido no campo "tempo administração em dias".

Quando chega a hora da administração de um medicamento, o dispositivo exibe uma notificação que aparece ao mesmo tempo em que é emitido um breve som. Esta notificação é constituída pelo nome do medicamento a ser administrado, um botão "Tomar" e um botão "Saltar". Se o utente pretender tomar o medicamento deve selecionar o botão "Tomar" para permitir a libertação do medicamento do dispensador. Caso não pretenda administrar, basta carregar no botão "Saltar". A logística por detrás da escolha de cada um destes botões é realizada no ficheiro ExecutableService.java, cujo um excerto é representado na Figura 3.18.

```
//Click confirm/Tomar button
Intent ConfIntent= new Intent(context,Confirmacao.class);
ConfIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

//Click Saltar toma
Intent SaltarIntent= new Intent(context,SaltarToma.class);
SaltarIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

String MedName=intent.getStringExtra( name: "Medication name");
String TimeMed = intent.getStringExtra( name: "TimeMed");
String NotificationID = intent.getStringExtra( name: "NotID");
Log.d( tag: "Lele execute time med",TimeMed);
```

Figura 3.18: Ficheiro ExecutableService.java.

Caso o utente escolha a opção "Tomar", os dados da ação são enviados para o ficheiro Confirmacao.java, Figura 3.19. Neste ficheiro os dados são guardados em *strings* e a notificação é cancelada. A informação sobre o medicamento é então enviada para o web host, que atualiza o estado da toma para "Administrada" na base de dados e verifica a posição do dispensador em que o medicamento se encontra.

```
String MedName = getIntent().getStringExtra( name: "MedName");
String TimeMed=getIntent().getStringExtra( name: "TimeMed");
String NotificationID = getIntent().getStringExtra( name: "NotID");

String Username = ((MyApplication) this.getApplication()).getUsername();
String pass  = ((MyApplication) this.getApplication()).getPass();
NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
mNotificationManager.cancel(Integer.parseInt(NotificationID));

BackgroundWorker backgroundWorker2=new BackgroundWorker( ctx: this);
backgroundWorker2.execute("ConfirMed",Username,pass,MedName,TimeMed);
```

Figura 3.19: Ficheiro Confirmacao.java.

A aplicação android recebe a informação do webhost referente à posição do medicamento. A aplicação android envia então a ordem de remoção com a posição do medicamento para a aplicação Java e esta trata da remoção e disponibilização ao utente. O resumo deste processo encontra-se simplificado no diagrama da Figura 3.20.

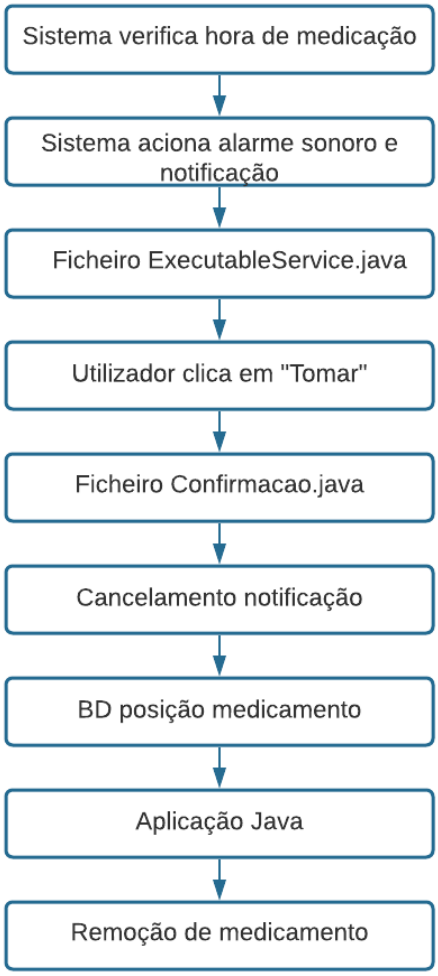

Figura 3.20: Processo toma de medicamento no ambiente Android Studio.

3.5. Serviços de Web Hosting

Os serviços de web hosting encontram-se divididos em duas categorias: os ficheiros PHP e a base de dados.

3.5.1. Base de dados

O sistema armazena os dados numa base de dados em MySQL. A Figura 3.21 ilustra um diagrama de entidade relacionamento, com as tabelas criadas na base de dados e respetivos relacionamentos. A base de dados é constituída pelas tabelas:

- **dados_registados** – É a tabela responsável por armazenar os dados do utilizador. Esta tabela é fundamental para o cumprimento do requisito R2, pois é aqui que são verificados os dados de início de sessão.
- **utente_medicacao** – Responsável pelo armazenamento dos medicamentos pertencentes a cada utente. Quando o sistema pretende obter a lista de medicamentos de cada utente, é esta tabela que é consultada. Esta tabela permite a satisfação do requisito R3. Sem uma correspondência entre o utente e a sua medicação, a tarefa de visualização da lista de medicamentos torna-se complexa.
- **medicacao** – Tabela onde são guardados os dados de cada fármaco inserido no sistema, de modo a satisfazer o requisito R4. Esta tabela não tem nenhuma informação acerca da medicação de cada utente. A tabela apenas guarda o nome do medicamento e o seu código de barras.
- **kit** – De acordo com cada utente, esta tabela guarda a informação sobre a disponibilidade do dispensador e a posição em que cada medicamento se encontra. A presente tabela corresponde aos requisitos R5 e R7. Sem uma informação correta sobre a disponibilidade e ocupação do dispensador, a inserção e remoção de medicamentos do mesmo torna-se impossível.
- **agenda** – Possui todas as datas e horas referentes a toma de cada medicação de cada utente. Satisfaz um dos principais requisitos do sistema, o requisito R7, cuja descrição é "alarme à hora da medicação". Esta funcionalidade também permite a satisfação do requisito R8, pois é nesta tabela, que os dados para a visualização do histórico ou da agenda são retirados.

- **time_medicacao** – Guarda a informação inserida pelo utente quando este faz a inserção de um novo fármaco. É nesta tabela que os dados como o tempo de administração e intervalo de tempo entre cada administração são guardados.

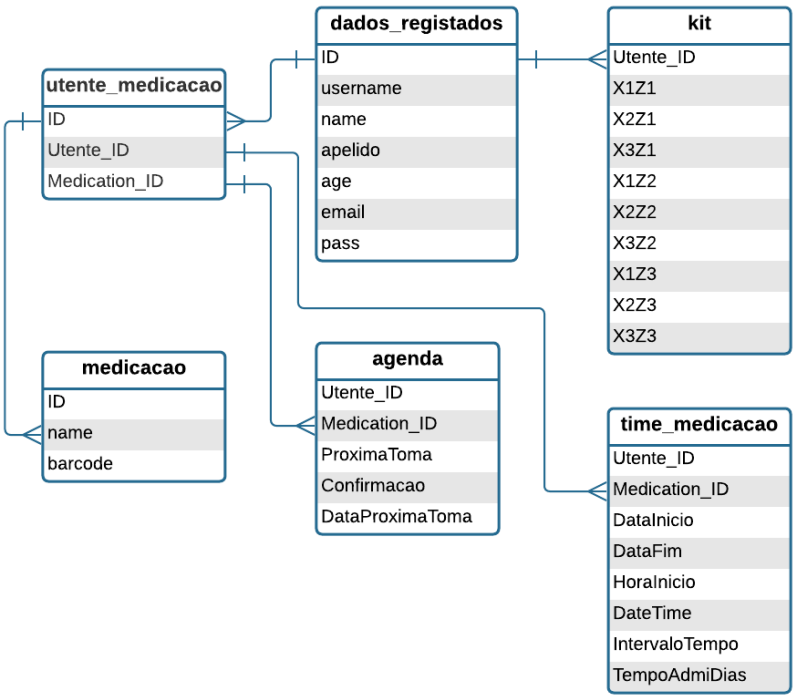

Figura 3.21: Diagrama entidade relacionamento.

3.5.2. Módulos PHP

PHP é uma das linguagens de programação usadas no desenvolvimento web. Como referido no capitulo dois, o PHP é geralmente usado para o desenvolvimento de aplicações que são capazes de criar conteúdos dinâmicos na World Wide Web [46].

A comunicação entre a aplicação android e a base de dados é quase sempre feita através de módulos PHP. Os módulos recebem os dados que foram enviados pela aplicação android, formatam estes dados e os enviam para a base de dados.

Na Tabela 3.2 estão apresentados alguns destes módulos que contribuíram para o bom funcionamento do sistema e a descrição das suas tarefas. Os restantes módulos encontram-se descritos em anexo.

Tabela 3.2: Ficheiros PHP mais importantes e sua descrição.

Módulo PHP	Descrição
conn	Realiza a conexão entre os módulos PHP e a base de dados.
login	Módulo responsável pela verificação dos dados de início de sessão.
register	Trata da receção dos dados para a criação de uma conta nova e da sua inserção na base de dados.
get_table_medications	Pesquisa na base de dados e retorna um vetor com a medicação do utente.
Insert_medication	Introduz na base de dados os dados da nova medicação.
AgendaShow	Pesquisa na base de dados a agenda do utente. Retorna um vetor com o nome da medicação e a hora que esta deve ser administrada.
Historico	Trata da verificação do estado que cada medicação após a hora atual da pesquisa. Retorna um vetor com a medicação, a hora que esta supostamente deveria ter sido administrada e o seu estado.

Em primeiro lugar, e em qualquer módulo PHP usado neste projeto, será necessário fazer a conexão com a base de dados, de acordo com os respetivos detalhes da mesma. Essa conexão é feita através do módulo conn.php. O módulo conn é por isso chamado no início de todos os módulos PHP criados.

/public_html/conn.php

```php
<?php
$db_name  ="id13340357_teselea";
$mysql_username ="id13340357_lea";
$mysql_password ="t4j6wqFo$6v3KCAY";
$server_name  ="localhost";
$conn = mysqli_connect($server_name,$mysql_username,$mysql_password,$db_name);

?>
```

Figura 3.22: Módulo conn.php.

Na Figura 3.22 encontra-se representado o módulo conn.php. Primeiro, são descritos os detalhes de acesso à base de dados usada no projeto e na linha 6 é feita a conexão com a mesma.

Se o utilizador não possuir uma conta na app este deve selecionar a opção de realizar um novo registo na aplicação android. O módulo responsável pelo tratamento dos dados inseridos para a criação de uma conta e de os enviar para a base de dados é o módulo register.php.

Caso o utilizador já possua uma conta, este insere os dados para a realização do início de sessão, e no módulo login.php é feita uma verificação dos dados na base de dados. Caso os dados do cliente estejam corretos e este se encontre na base de dados, o módulo login.php retorna uma mensagem informando que o login foi efetuado com sucesso. A Figura 3.23 ilustra o módulo login.php. Na linha 9 é feita a verificação com a base de dados e caso seja positiva a mensagem de sucesso é enviada para aplicação android.

```
1  <?php
2  require "conn.php";
3  $user_name =$_POST["user_name"];
4  $user_password =$_POST["password"];
5  $mysql_qry="select * from dados_registados where username like '$user_name'
       and pass like '$user_password';";
6
7  $result = mysqli_query($conn , $mysql_qry);
8
9  if(mysqli_num_rows($result) > 0){
10     $row=mysqli_fetch_assoc($result);
11     echo "Login success";
12
13 }else{
14     echo "Login not sucess";
15 }
```

Figura 3.23: Módulo login.php.

O módulo em PHP que segue o módulo login.php varia de acordo com as intenções do utilizador. Se este pretender aceder ao seu histórico, o módulo em PHP invocado será o histórico.php, caso pretenda aceder a agenda, o módulo em PHP chamado pela aplicação será o agenda.php.

O fluxograma representado na Figura 3.24 apresenta o processo completo do percurso dos módulos em PHP conforme as decisões do utente.

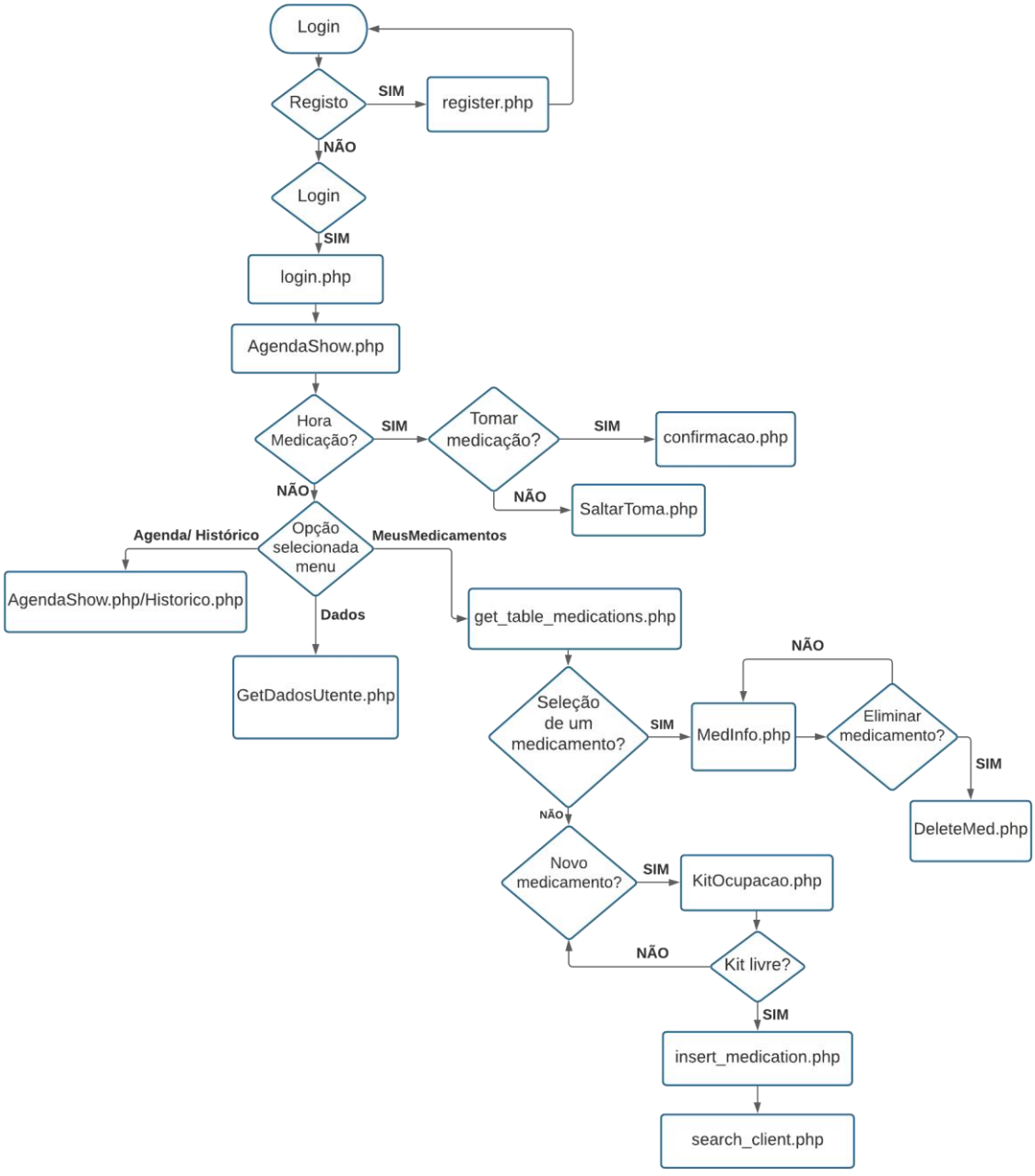

Figura 3.24: Fluxograma que apresenta o processo dos módulos em PHP.

3.6. Dispensador e Aplicação Java

Devido à atual situação de pandemia de COVID-19, não foi possível o acesso ao dispensador físico localizado nas instalações da faculdade. Em alternativa, recorreu-se uma versão simulada do dispensador.

Um dos grandes fatores da linguagem Java é a sua portabilidade, ou seja, podemos esperar que um código Java funcione perfeitamente em qualquer máquina capaz de correr uma máquina virtual java.

Uma máquina virtual Java ou como é conhecida JVM , do inglês Java Virtual Machine, é uma maquina virtual Java que permite a execução de programas desenvolvidos em ambiente Java [47], [48]. JVM permite a execução dos programas Java em qualquer dispositivo ou sistema operativo [49].

A Figura 3.25 ilustra a arquitetura de uma Java Virtual Machine. No topo desta arquitetura encontram-se as aplicações Java. As aplicações java funcionam no topo desta arquitetura e o dispensador está ligado ao nível do hardware, sendo então necessário a passagem pelas barreiras existentes entre estes dois elementos. Por norma uma aplicação em Java não consegue passar da JVM, por isso, é que uma aplicação java pode funcionar na maior parte dos computadores independentemente do sistema operativo, pois JVM garante a compatibilidade visto que da JVM para cima é tudo uniforme.

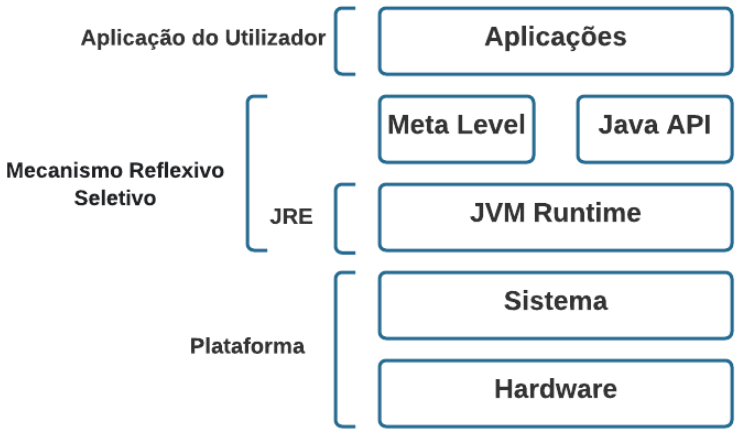

Figura 3.25: Arquitetura Java Virtual Machine. Inspirado em [50].

Contudo, pode ser necessário usar código compilado nativamente para uma arquitetura específica. Alguns dos motivos podem incluir a necessidade de lidar com algum tipo de hardware, melhorar o funcionamento de processos complexos ou a reutilização de alguma biblioteca pré-existente.

De modo a alcançar estes objetivos, e a tornar possível a comunicação com o dispensador, foi criado o Java Native Interface (JNI). Java Nativa interface (JNI) é uma interface que permite a interação entre códigos escritos em linguagens diferentes [51]. JNI funciona como uma ponte entre a nossa máquina virtual java e um código nativo, normalmente C ou C++ [52]. A Figura 3.26 representa a estrutura JNI.

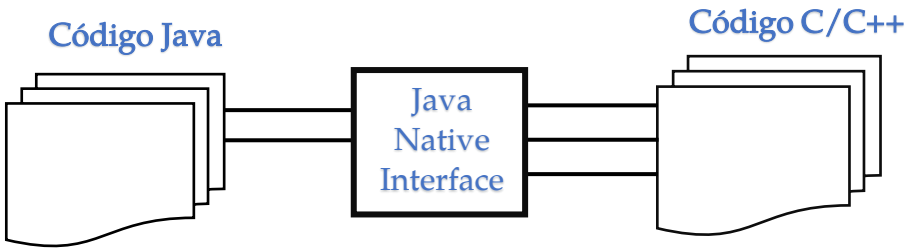

Figura 3.26: Estrutura JNI.

As bibliotecas compartilhadas serão transformadas em ficheiros DLL no caso do projeto em desenvolvimento. Estes ficheiros são ligados dinamicamente, apenas incluindo o endereço da biblioteca. Esta ligação permite que as funções criadas estejam num espaço de memória e sempre que necessário um programa possa acedê-las, sem a necessidade de existirem varias cópias das funções [53].

3.6.1. Componentes necessários

Serão necessários quatro componentes para o desenvolvimento da aplicação Java, estes elementos são:

- **Compilador C/C++**

  É necessário um compilador com a capacidade de gerar uma biblioteca compartilhada para a plataforma desejada. Neste projeto, o compilador C/C++ utilizado foi o do Visual Studio.

- **Código nativo**

Como referido no ponto anterior, neste projeto o código nativo será escrito usando a ferramenta Visual Studio. Começou-se pela inserção dos ficheiros *.c, fornecidos pelos docentes da unidade curricular Sistemas de Tempo Real do curso Engenharia Eletrotécnica e de Computador na Universidade Nova de Lisboa. Estes ficheiros são os apresentados na Tabela 3.3. Eles irão permitir a conexão com a componente de hardware do dispensador.

Tabela 3.3: Ficheiros *.c que permitem a conexão com o dispensador.

sim.server.c and mongoose.c	Gere a interação com o dispensador
NIDAQmx.lib	Permite a conexão com o DAQ board
Interface.c	Permite a mudança entre dispensador e simulador

O ficheiro lowlevel.c foi desenvolvido para a gestão das funcionalidades de baixo nível. Foram criadas as funções, apresentadas na Tabela 3.4, com base nos atuadores e sensores presentes no dispensador, de modo a garantir o funcionamento desejado do dispensador. Umas descrições mais detalhadas de grande parte destas funções criadas encontram-se em anexo.

Tabela 3.4: Funções essências de baixo nível.

Função	Descrição
moveXLeft/moveXRight	Move a gaiola do dispensador (elemento que transporta o medicamento) para a esquerda/direita.
moveZDown/moveZUp	Move a gaiola num movimento ascendente/ descendente no eixo Z.
has_piece	Verifica se a gaiola possui alguma peça.
is_at_z_down(), is_at_z_up()	Verifica se a gaiola se encontra na posição superior ou inferior no eixo Z.
gotoX/ gotoY/ gotoZ	Desloca a gaiola para a posição X, Y, Z desejada.

callibrate	Calibração do dispensador para que a gaiola se encontre numa posição reconhecível.
putPiece_task/ getPiece_task	Coloca/retira uma peça de uma determinada posição.

- **Ficheiro Header JNI**

Num ficheiro denominado kit.h foram especificadas todas as funções desenvolvidas no ficheiro lowlevel.c. Estas funções permitirão a utilização do programa nativo. Encontra-se ilustrado na Figura 3.27, um excerto do ficheiro kit.h, com algumas das funções especificadas no ficheiro.

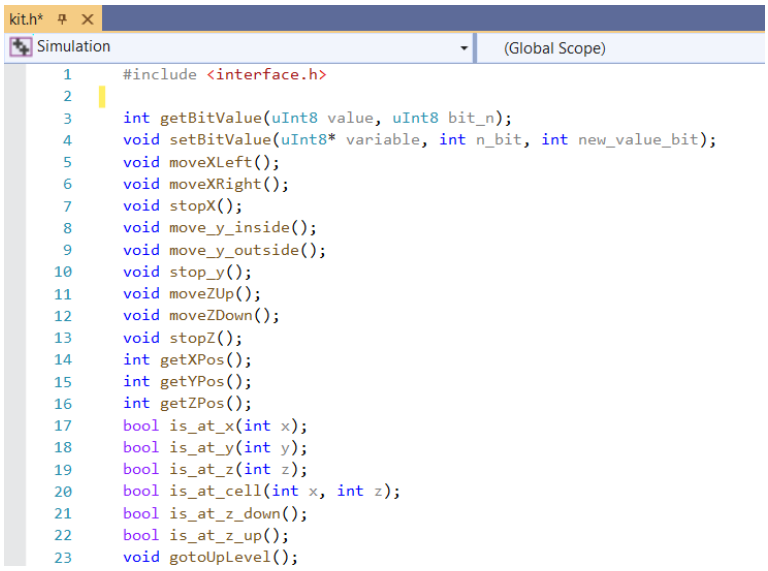

Figura 3.27: Ficheiro *header* kit.h.

- **Código Java**

O ambiente usado para o desenvolvimento da aplicação Java foi o NetBeans. Começou-se pela criação da classe Dispensador.java. Neste ficheiro, foi realizada uma cópia das assinaturas criadas no ficheiro kit.h. Estas assinaturas são feitas de acordo com as convenções de nomenclatura Java. A Figura 3.28 ilustra o ficheiro Dispensador.java, onde é possível visualizar todas as cópias de assinaturas feitas e as repetitivas nomenclaturas.

```
*/
public class Dispensador {

static{
    System.load("C:\\str\\Simulation\\x64\\Debug\\Simulation.dll");
}

static synchronized native public void moveXLeft();
static synchronized native public void moveXRight();
static synchronized native public void moveYDown();
static synchronized native public void moveYUp();
static synchronized native public void moveZUp();
static synchronized native public void moveZDown();
static synchronized native public void stopX();
static synchronized native public void gotoX (int x_dest);
static synchronized native public void gotoZ(int z);
static synchronized native public void goto_xz(int x, int z);
static synchronized native public void callibrate();
static synchronized native public void putPiece_task();
static synchronized native public void getPiece_task();
static synchronized native public boolean has_piece();

//static synchronized native public void callibration();
static native public int getXPos();
static native public int getYPos();
static native public int getZPos();
static native public void createDigitalInput(int port);
```

Figura 3.28: Ficheiro Dispensador.java.

A partir da classe Dispensador.java foi gerado um outro ficheiro header. Denominou-se este ficheiro Dispensador.h. Um excerto do ficheiro encontra-se representado na Figura 3.29.

SplitterLine.h

Source History

```
/*
 * Class:     SplitterLine
 * Method:    moveXLeft
 * Signature: ()V
 */
JNIEXPORT void JNICALL Java_SplitterLine_moveXLeft
  (JNIEnv *, jclass);

/*
 * Class:     SplitterLine
 * Method:    moveXRight
 * Signature: ()V
 */
JNIEXPORT void JNICALL Java_SplitterLine_moveXRight
  (JNIEnv *, jclass);

/*
 * Class:     SplitterLine
 * Method:    moveYDown
 * Signature: ()V
 */
JNIEXPORT void JNICALL Java_SplitterLine_moveYDown
  (JNIEnv *, jclass);
```

Figura 3.29: Ficheiro SplitterLine.h.

Depois de criado, os dados dentro do ficheiro Dispensador.h foram transferido para o Projeto Visual Studio e completados de acordo com a Figura 3.30.

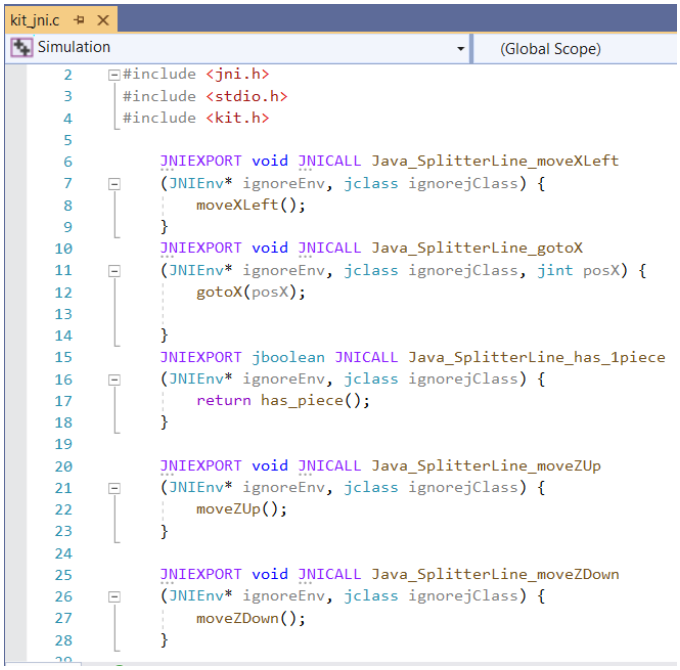

Figura 3.30: Ficheiro kit_jni.c.

A próxima etapa consiste em transformar o projeto Visual Studio numa biblioteca dinâmica DLL. O projeto Visual Studio em desenvolvimento, já possui as especificações necessárias para que seja transformado numa biblioteca dinâmica DLL, sendo apenas necessário efetuar o *build* do mesmo. Logo após o *build*, o caminho DLL gerado, Figura 3.31, foi copiado para o ficheiro java Dispensador.Java no ambiente Netbeans [54].

```
public class Dispensador {

static{
    System.load("C:\\Dissertação\\Dispensador\\Simulation\\x64\\Debug\\Simulation.dll");
}
```

Figura 3.31: Ficheiro Dispensador.Java com caminho DLL.

Como referido anteriormente, devido a situação pandémica vivida no presente ano, não foi possível o acesso ao dispensador físico localizado nas instalações na faculdade. Como consequência foi usado um simulador do mesmo. Além do simulador possuir características praticamente idênticas ao dispensador físico, este permite o desenvolvimento do programa, bem como testes e correção de erros.

Num contexto real, existiria um sistema físico equipado com um pequeno teclado no próprio dispensador, de modo a permitir que o utilizador tivesse um acesso manual ao dispensador. Como o projeto foi desenvolvido num sistema virtualizado, o sistema físico que permitiria a interação do dispensador com o utilizador foi substituído por uma aplicação em Java com o layout igual ao ilustrado na Figura 3.32. O simulador do dispensador possui apenas 9 posições livre, que se verificou suficiente para o desenvolvimento do sistema.

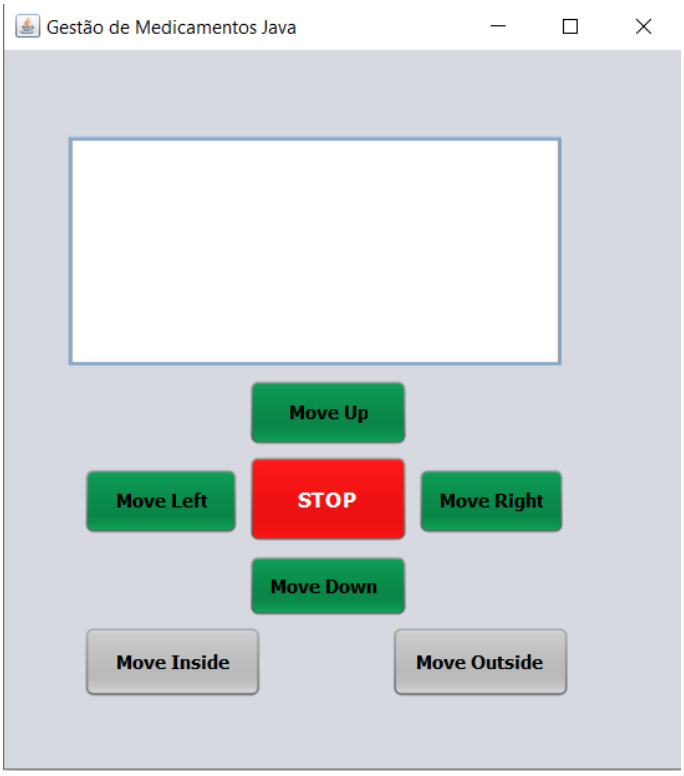

Figura 3.32: Aplicação Java.

A aplicação Java é composta por um painel de exibição de mensagem e de sete botões que são:

- **Move Up** – Move o kit num sentido ascendente no eixo Z.
- **Move Down** – Move o kit num sentido descendente no eixo Z.
- **Move Left** – Move o kit para a esquerda no eixo X.
- **Move Right** – Move o kit para a direita no eixo X
- **Move Inside** - Move a gaiola para dentro no eixo Y.
- **Move Outside** – Move a gaiola para fora no eixo Y.
- **Stop** – Interrompe o movimento em todos os eixos.

Depois destes passos, já será possível fazer a conexão ao dispensador. Para aceder ao dispensador é necessário correr o projeto criado no ambiente NetBeans e colocar o seguinte link na barra de pesquisa do nosso navegador: *http://localhost:8081/dispensador.html*. O dispensador e a interface de utilizador java ficarão visíveis e será feita a calibração para a posição X=2 e Y=2. A Figura 3.33 mostra o dispensador após o processo de calibração.

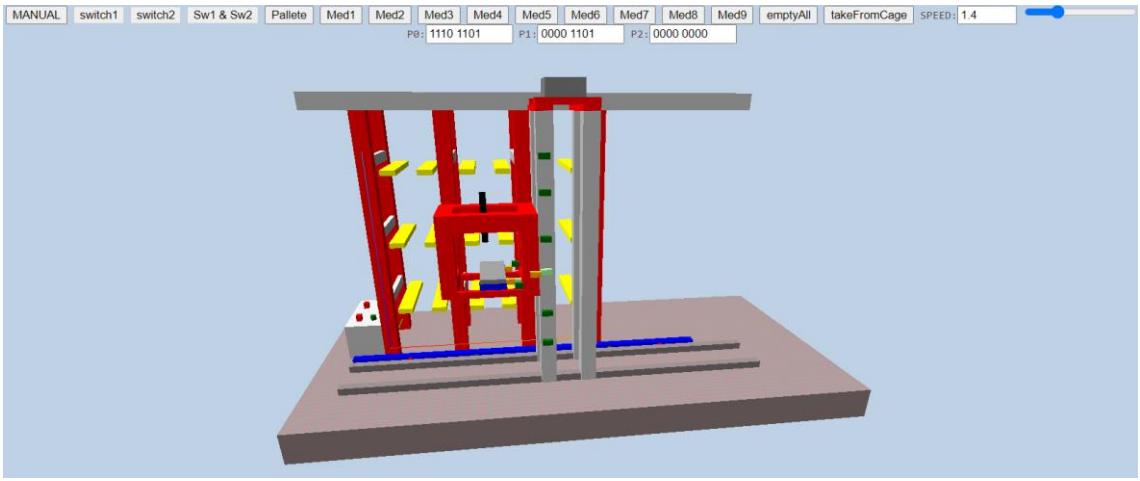

Figura 3.33: Dispensador logo depois da calibração.

3.7. Conexão entre a APP Android e a aplicação JAVA

Será necessário criar uma ligação entre a App Android e a aplicação JAVA, de modo que a aplicação android consiga comunicar com a aplicação Java e consequentemente com dispensador. Este envio de informação é feito através de sockets TCP.

Imaginemos dois dispositivos como representados na Figura 3.34, ambos os dispositivos estarão conectados a uma rede sem fios e terão um endereço IP e uma porta. É imperativo que ambos os dispositivos estejam conectados a mesma rede WIFI, para que a comunicação seja feita com sucesso.

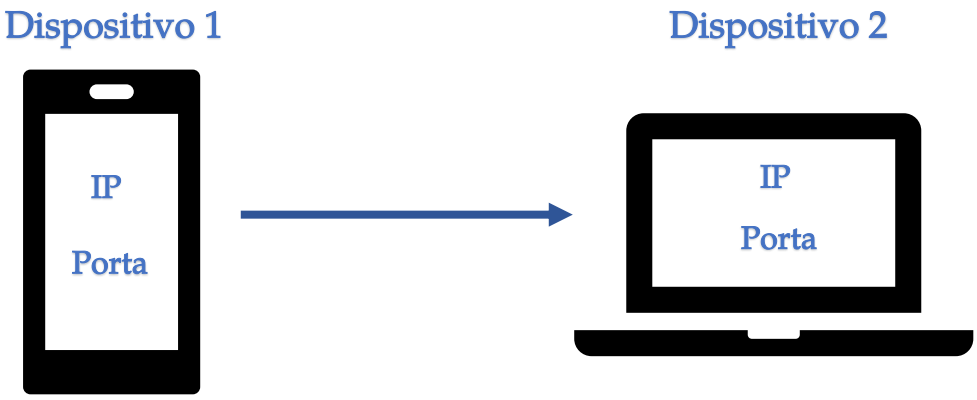

Figura 3.34: Comunicação TCP Socket.

Foi desenvolvido no ambiente Android Studio uma classe chamada Socket. A classe Socket servirá para a realização da comunicação via TCP Socket entre a aplicação android e a aplicação java.

Como o Android Studio facilita imenso este processo, apenas sendo necessário a declaração de alguns elementos. Encontra-se representado na Figura 3.35 o ficheiro SendSocket.java responsável pela comunicação via Socket da aplicação. Este ficheiro é chamado quando a base de dados retorna a posição do medicamento a ser dispensado do dispensador.

```
public class SendSocket extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        MessangeSender messangeSender = new MessangeSender();
        String pos= getIntent().getStringExtra( name: "Position");
        //Log.d("Lele send socket",pos);
        messangeSender.execute(pos);
        Intent intent = new Intent( packageContext: this,MainMenuAfterLogin.class);
        this.startActivity(intent);
    }
}
```

Figura 3.35: Ficheiro SendSocket.java.

Na aplicação android, o socket é criado num outro ficheiro denominado MessangeSender.java, apresentado na Figura 3.36. Neste ficheiro também é definido o endereço IP do dispositivo e a porta que será usada, que no caso é a porta 6000.

```
    protected Void doInBackground(String... voids) {

        String message = voids[0];

        try {
            s = new Socket( host: "192.168.1.5", port: 6000);
            pw= new PrintWriter(s.getOutputStream());
            pw.write(message);
            pw.flush();
            pw.close();
            s.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

Figura 3.36: Ficheiro MessangeSender.java

O ambiente NetBeans também possui uma função para a comunicação Socket. Para isso, basta criar uma variável do tipo ServerSocket, como ilustrado na Figura 3.37, com o número da porta igual a definida no ambiente android Studio.

```
ss = new ServerSocket(6000);
```

Figura 3.37: Socket ambiente NetBeans

As mensagens partilhadas através de sockets são apenas para ordenar a inserção ou remoção de um medicamento de uma posição específica do dispensador. Um exemplo de mensagem partilhada é "GET X1Z1", que ordena a remoção do medicamento atualmente presente na posição x=1 e z=1 do dispensador.

CAPÍTULO 4

TESTES DO SISTEMA

De modo a demonstrar e comprovar o funcionamento do sistema dedicou-se este capítulo para a apresentação de alguns testes realizados ao sistema. Estes testes são totalmente baseados nas interações do utilizador/cliente com a aplicação.

Os testes irão incluir várias etapas, desde o processo de início de sessão, inserção e remoção de medicação, visualização da agenda/histórico e de outros processos do sistema.

Será feito uma verificação e validação do funcionamento do sistema de modo a averiguar se os requisitos funcionais descritos no capítulo anterior são cumpridos com sucesso.

4.1. Verificação

O processo de verificação consiste na avaliação de um sistema de modo a garantir que os requisitos funcionais previamente definidos sejam alcançados [55]. Numa fase pós desenvolvimento, a verificação envolve a realização de testes com várias etapas, com o objetivo de garantir que o produto ou serviço continua a satisfazer os requisitos e especificações definidas numa fase inicial do projeto [56].

No capítulo anterior foram definidos oito requisitos funcionais. Neste capítulo são realizados testes ao sistema de modo a avaliar se o sistema desenvolvido satisfaz todos esses requisitos com sucesso.

Requisito 1 - Registo de um novo utilizador

O primeiro requisito funcional definido foi a introdução de um novo utilizador no sistema, caso este ainda não se encontre inserido. O ambiente inicial, que é primeiramente apresentado ao utilizador, encontra-se representado na Figura 4.1.

Figura 4.1: Ambiente inicial da aplicação.

Para a realização do início de sessão com sucesso o utilizador necessita colocar o nome de utilizador e a palavra-passe correspondente.

De modo a verificar o sucesso do primeiro requisito fez-se uma tentativa de início de sessão com dados que não se encontravam inseridos no sistema. Como resultado dessa tentativa falhada, o sistema mostra uma caixa diálogo com a mensagem "Dados Errados" (ilustrado na Figura 4.2). Esta mensagem de erro é apresentada quando a palavra-passe ou nome de utilizador ou ambos são inseridos incorretamente ou não se encontram na base de dados.

Figura 4.2: Tentativa de início de sessão com dados errados.

Requisito 2 – Início de sessão do utilizador

Após o início de sessão ser realizado com sucesso uma caixa de diálogo com a mensagem "Login efetuado com sucesso" é apresentado por breves segundos junto com o menu geral, como apresentado na Figura 4.3. É no menu geral que será possível a realização das principais interações entre o utilizador e o sistema. Na parte superior do menu geral é exibido o nome do utilizador do utente que iniciou a sessão. De seguida, encontram-se os botões com as opções para outras áreas da aplicação. Cada botão é composto pelo nome da área que o utilizador pretende entrar e de uma breve descrição do que é possível realizar dentro dessa secção da aplicação. No canto superior direito da aplicação, zona representada por uma circunferência vermelha na Figura 4.3, encontra-se um submenu onde é possível o término da sessão. Para tal basta que o utilizar clique no símbolo ⋮ e clicar no botão "Terminar Sessão".

Figura 4.3: Início de sessão efetuado com sucesso e menu principal.

Requisito 3 – Inserção de um novo medicamento no sistema

Pode-se considerar esta etapa como uma das mais importante do sistema. Para a inserção de um novo medicamento no sistema o utilizador precisa entrar na área "Meus Medicamentos". Após o click, é apresentado um layout com um botão na parte superior com a palavra "Novo". É então necessário um novo click nesse botão para a aparição do layout representado na Figura 4.4. Este layout permitirá a inserção dos dados do novo medicamento.

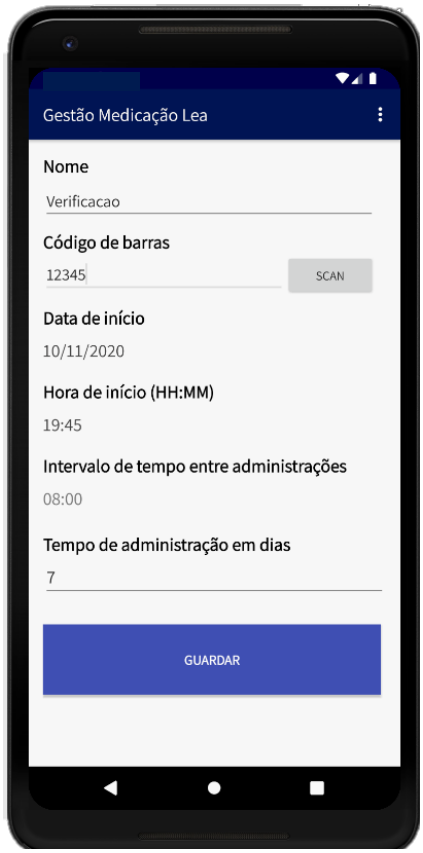

Figura 4.4: Inserção de um novo medicamento.

Foi realizada a inserção do medicamento com o nome "Verificacao" e com o código de barras 12345. Este medicamento tem como data e hora da primeira toma o dia 10/11/2010 às 19h45. O intervalo de tempo entre administrações é de 8 horas e o tempo de administração é de 7 dias.

Após o click no botão "GUARDAR", o menu principal volta a ser apresentado. De modo a verificar se o medicamento foi realmente inserido é necessário visualizar a lista de medicamentos do utente, e, para tal, o utente deve selecionar a opção "Meus Medicamentos". O resultado desta ação encontra-se ilustrado na Figura 4.5, onde, é possível observar, que o medicamento com o nome "Verificacao" encontra-se na lista de medicamentos do utente.

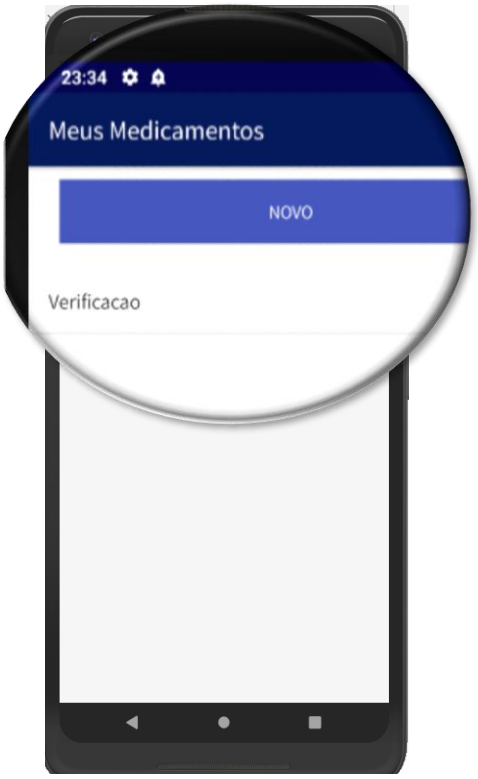

Figura 4.5: Verificação da inserção do novo medicamento.

Requisito 4 – Inserção de um novo medicamento no dispensador

No sistema desenvolvido o dispensador serviu para o armazenamento da medicação. Após a inserção do medicamento na aplicação, o dispensador deve, automaticamente e logo de seguida, colocar o medicamento numa posição não ocupada. Primeiro a aplicação Android precisa enviar uma mensagem para a aplicação Java com informações à cerca do procedimento. A mensagem "Put X1Z1" é recebida e ilustrada no painel da aplicação Java, Figura 4.6. A mensagem indica que o medicamento, atualmente presente na gaiola, deve ser colocado na posição do dispensador X=1 e Z=1.

A Figura 4.7 ilustra a fase inicial em que o dispensador possui um medicamento na sua gaiola enquanto espera por alguma ordem e a segunda fase onde o mesmo medicamento foi inserido na posição previamente designada.

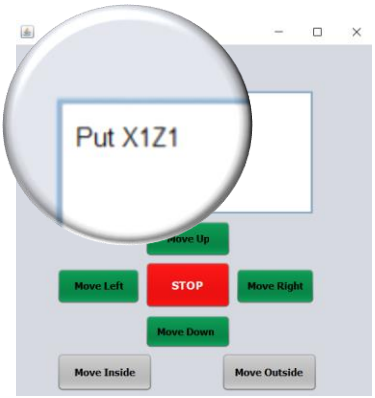

Figura 4.6: Dispensadorcom medicamento na gaiola

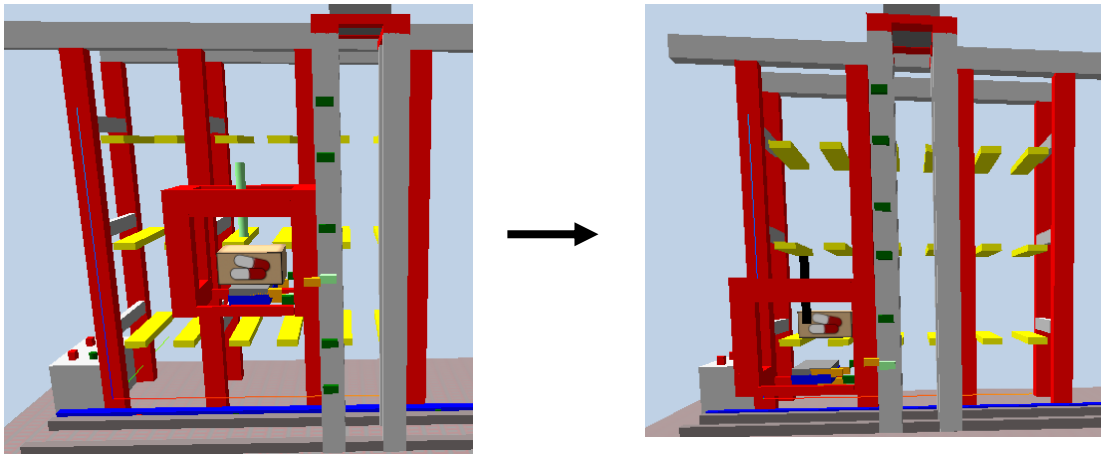

Figura 4.7: Dispensador antes e depois de inserção de medicamento.

Requisito 5 -Visualização da lista de medicamentos na aplicação

Este requisito foi verificado quando se realizou a verificaçao do terceiro requisito, que consite na colocação com sucesso de um novo medicamento no sistema. Na Figura 4.5 é ilustrada a lista de medicamentos atualmente inseridas pelo utente, que até ao momento, apenas é o medicamento com o nome "Verificacao".

Foi inserido um novo medicamento com o nome "Verificacao 2" para averiguar a capacidade do sistema de expor todos os medicamentos em forma de lista. Como verificado na Figura 4.8 o sistema cumpre com sucesso este requisito.

Figura 4.8: Verificação da lista de medicamentos de um utente.

Requisito 6 – Lembrete na hora da medicação

Um dos objetivos do sistema desenvolvido é ajudar o utente na administração da sua medicação e uma parte importante desta ajuda é ajudar que uma toma não seja esquecida ou atrasada. No momento da inserção de cada medicação, o utente introduziu dados acerca da administração do medicamento, como a data e a hora da primeira administração, o intervalo de tempo entre administrações e durante quantos dias este medicamento deve ser administrado.

Após a inserção destes dados, o sistema automaticamente cria uma agenda para manusear a toma de cada medicamento. O sistema foi desenvolvido de modo que o dispositivo móvel emita um pequeno som na hora pretendida para a administração do medicamento.

Relembremos que o medicamento previamente inserido no sistema com o nome "Verificação" possui como data da primeira administração 10/11/2020 às 19h45.

Na Figura 4.9 é possível confirmar que às 19h45 min, uma notificação com o nome do medicamento "Verificacao" foi exposta ao utilizador. Como é obvio através da imagem não é possível ouvir o som que foi emitido simultaneamente com a notificação.

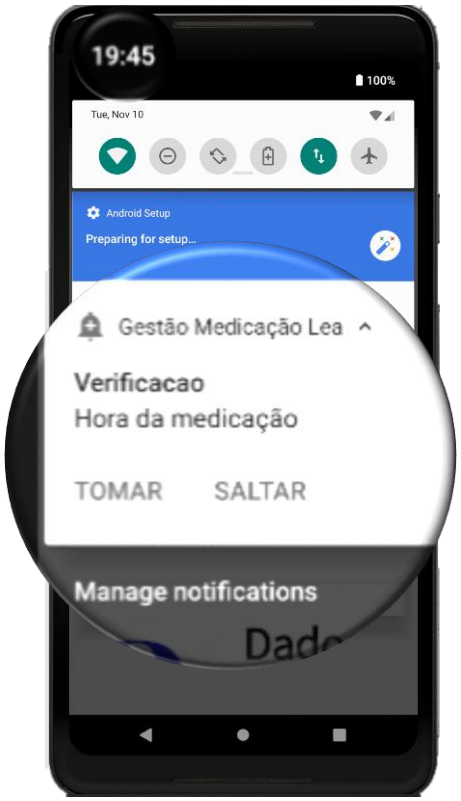

Figura 4.9: Notificação no dispositivo móvel.

Requisito 7 – Remoção de medicamento do dispensador após confirmação da toma

Após o cliente clicar no botão tomar na notificação, a aplicação android envia uma mensagem para a aplicação java com a execução pretendida e a posição do medicamento que deve ser ingerido. Neste caso a mensagem enviada é "*Get X1Z1*", como ilustrado na Figura 4.10.

Cabe à aplicação Java tratar do processo da remoção do medicamento da posição pretendida do dispensador. O processo, depois deste click do utilizador, encontra-se ilustrado na Figura 4.11.

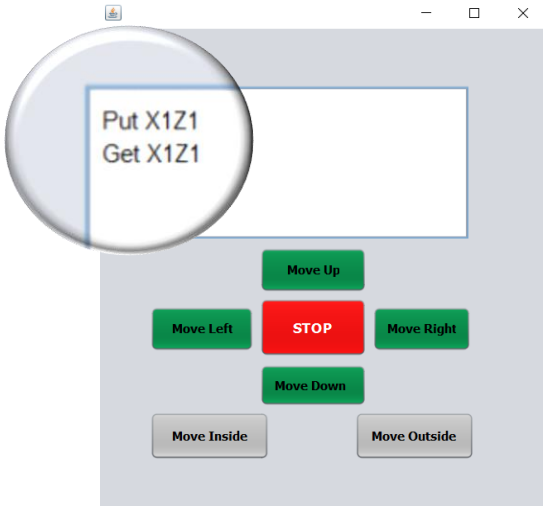

Figura 4.10: Aplicação Java com mensagem *Get*.

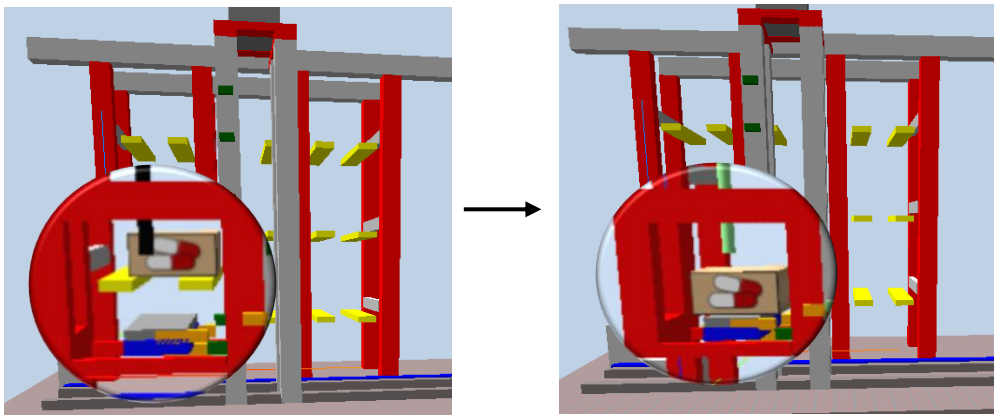

Figura 4.11: Processo de remoção de medicamento.

Requisito 8 – Visualização da agenda e histórico

A agenda possui o registo das futuras tomas do utente. De modo a ter acesso à agenda o utente deve selecionar a opção "Agenda" do menu principal. Recordemos que o medicamento inserido com o nome "Validacao" possui a sua primeira toma no dia 10/11/2020 às 19:45. Após o click na opção "Agenda", a imagem representada na Figura 4.12 foi disponibilizada ao cliente.

Na parte central superior aparece o dia e duas setas para a seleção do dia que se pretende a visualização da agenda. Na parte inferior encontra-se a lista da medicação que deve ser ingerida no presente dia e a respetiva hora.

Figura 4.12: Verificação da agenda.

A primeira toma da medicação foi realizada às 19h45 do dia 10-11-2020. Ao adicionarmos 8 horas, que é o intervalo de tempo entre administrações, verificamos que a segunda toma deve ser feita no dia 11-11-2020 às 3h45 e a terceira toma às 11h45 e assim sucessivamente. Se o utente clicar na seta do lado direito, avança um dia na agenda, e obtém um layout com o aspeto igual ao da Figura 4.13. Fica assim verificado que a agenda exibe corretamente o horário das medicações.

Figura 4.13: Seleção outro dia na agenda.

O histórico do sistema apenas apresenta as medicações cuja hora de administração se encontra no passado. Através da Figura 4.13, podemos observar que no dia 11-11-2020 o medicamento "Verificacao" devia de ter sido administrado três vezes: a primeira às 3h45, a segunda às 11h45 e a última às 19h45.

O utente ignorou todas as outras notificações que ocorreram antes até às 19h45 e no histórico todas as medicações para o dia 11-11-2020 possuem o estado "Sem resposta", como ilustrado na Figura 4.14.

Figura 4.14: Histórico antes da confirmação do medicamento.

Às 20h33 foi selecionada a opção "Tomar" da última notificação do dia (19h45 min). Após esta seleção e do click na seção histórico, este foi atualizado e a última toma do dia passou do estado "Sem resposta" para o estado "Administrado", Figura 4.15, informando assim que o medicamento "Verificacao" foi entregue ao utilizador com sucesso.

Figura 4.15: Histórico após administração do medicamento.

4.2. Validação do dispensador

A validação é um processo que tem como objetivo avaliar o sistema para determinar se este satisfaz os requisitos especificados [57]. Este processo deve fornecer provas de que o sistema resolve os problemas definidos e satisfaz as necessidades do utilizador. Por norma, a validação de um sistema é realizada pelo cliente ou utilizador [58].

A implementação de um cenário de validação seria altamente dispendiosa em termos financeiros e de recursos humanos. Seria necessário facultar algumas dezenas ou centenas de unidades do dispensador para serem distribuídas pelas pessoas que fariam parte da validação. No caso destas pessoas não possuírem um dispositivo movel com a capacidade de instalação da aplicação android, um teria

de ser fornecido. A disponibilização destes recursos encontra-se fora dos limites da presente dissertação.

Como agravante, a situação atual que estamos a viver devido à pandemia COVID-19 e das medidas de contenção e de prevenção, tornou impossível qualquer situação que exija um contacto próximo com pessoas, inviabilizando um processo de validação adequado. Em alternativa, será descrito um cenário de validação do dispensador e dos passos envolvidos na sua validação.

Começar-se-ia por organizar um grupo de indivíduos, preferencialmente que ingerissem uma quantidade substancial de medicamentos diariamente e com diferentes problemas de gestão de medicação.

O grupo seria dividido em dois, com o mesmo número de pacientes em cada um dos grupos. Ambos os grupos receberiam as informações sobre a correta toma dos medicamentos. Para uma metade do grupo os seus medicamentos seriam inseridos no dispensador desenvolvido, como exemplificado na Figura 4.16. Ambos grupos seriam atentamente monitorados por um médico ou outra pessoa igualmente qualificada.

Seria pedido ao grupo sem acesso ao dispensador que realizasse um registo da hora em que administrava cada um dos fármacos. Diariamente ou semanalmente, seria realizada uma contagem aos medicamentos destes grupos de modo a verificar se algum medicamento foi esquecido e não foi ingerido.

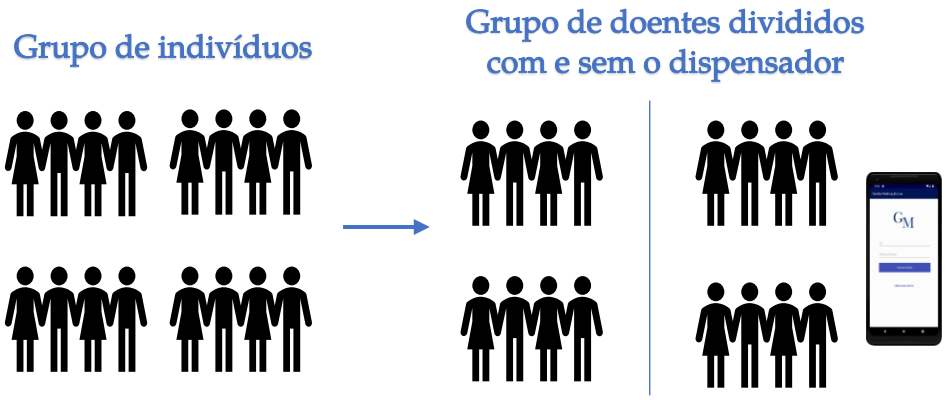

Figura 4.16: Validação do dispensador.

O segundo grupo não teria a necessidade de realizar o registo das horas de cada administração isto porque, a aplicação Android envia essa informação para a base de dados quando o paciente clica no botão "Tomar" da notificação de cada medicação.

Passado o tempo necessário para a obtenção de uma quantidade substancial de dados, estes seriam recolhidos e analisados. Ao grupo sem o dispensador, seria feita uma contagem dos medicamentos que ainda restavam a cada utente, e uma análise das horas que cada individuo registou a toma de cada medicamento. Com os dados de todos os utentes, seria realizado um cálculo de modo a obter-se uma percentagem dos medicamentos que não foram ingeridos ou que foram ingeridos incorretamente.

Os dados referentes a toma de cada medicamento da outra metade do grupo, seriam recolhidos diretamente da base de dados. Far-se-ia o mesmo procedimento realizado a outra metade do grupo, uma análise e cálculo das tomas com erros.

Com esta análise de ambos os grupos, esperar-se-ia certificar que o grupo com o dispensador registaria uma percentagem muito inferior de falhas, comparada com a percentagem do grupo sem o dispensador. Como consequência da menor percentagem de falhas no grupo com dispensador, a quantidade de erros de medicação seria também inferior. Esperar-se-ia então inferir que a utilização do dispensador levaria a uma diminuição dos atrasos na ingestão dos medicamentos, a redução de falhas, de acidentes como a toma duplicada de um medicamento, reduzindo assim os erros de medicação e como consequência uma gestão melhorada dos medicamentos e do bem-estar dos utentes.

CAPÍTULO 5

CONCLUSÕES

5.1 Síntese do trabalho desenvolvido

Esta dissertação teve como objetivo a criação de uma infraestrutura de gestão de medicamentos. O sistema é constituído por uma aplicação Android, um dispensador e um servidor para o alojamento de informações e dados. O dispensador é usado para armazenar os medicamentos e é controlado através de uma aplicação Java que possui uma pequena interface.

A aplicação Android é responsável pela comunicação do utilizador com os restantes componentes do sistema. Esta parte do sistema é responsável pela inserção, remoção e atualização de todos os dados necessários para a realização da correta gestão dos medicamentos. A aplicação Android permite que o cuidador possa iniciar sessão e ter acesso ao histórico do utente, de modo a verificar se ocorreu algum erro ou problema com a administração de algum medicamento.

O sistema foi desenvolvido de modo que após a inserção de um medicamento no sistema, uma agenda seja criada e guardada numa base de dados. Quando chega a hora da ingestão de um medicamento, a aplicação Android emite um breve som de alarme de modo a alertar o utente. Cabe ao utente, decidir se deseja administrar o medicamento ou ignorar a notificação. Caso o utente decida ingerir, o medicamento será retirado da posição em que se encontra no dispensador e disponibilizado ao utente.

O sistema desenvolvido é então uma ferramenta que funciona como um dispensador de medicamentos, com a capacidade de possuir uma agenda programável em que o utilizador ou cuidador configura os horários corretos para a administração dos fármacos, de acordo com a prescrição médica ou com as necessidades de cada utilizador.

Este sistema é importante pois de acordo com o levantamento realizado na revisão da literatura, concluiu-se que ocorrem muitos acidentes relacionados com a toma da medicação e, que só cerca de 50% dos cidadãos em todo o mundo tomam corretamente os medicamentos.

5.2 Resultados

No capítulo quatro, foi apresentada uma verificação ao funcionamento do sistema. Esta verificação possibilitou concluir que o sistema cumpre todos os requisitos previamente definidos no terceiro capítulo e que permite aliviar a preocupação e trabalho que exige a correta gestão de medicamentos.

Na verificação foi possível averiguar que o sistema é capaz de reconhecer cada utilizador com base nos dados de início de sessão, e nos casos em que os dados sejam incorretos ou não pertencessem aos dados já existentes na base de dados, o sistema não permite o início de sessão.

O sistema desenvolvido também é capaz de comunicar ao utilizador a hora da toma de cada medicação através de um sinal sonoro que ocorre simultaneamente quando aparece uma notificação no dispositivo móvel. O sistema permite também realizar a inserção e remoção dos medicamentos no dispensador usado no projeto.

Esta verificação permite concluir que se cumpriram todos os objetivos iniciais estabelecidos, criando-se uma ferramenta de auxílio na gestão de fármacos, que pode ser usada para melhorar o bem-estar e saúde dos utentes.

5.3 Trabalhos futuros

Com trabalhos futuros, podem ser considerados funcionalidades adicionais ao sistema desenvolvido. Pode ser acrescentado a capacidade de comando do dispensador a longa distância, ou seja, permitir que a comunicação possa ser feita entre o dispensador e a aplicação android sem a necessidade de estes estarem conectados a mesma rede WIFI. Outra característica podia ser a repetição do alarme que é emitido quando aparece a notificação da toma da medicação, ou seja, se o utente não carregar no botão que permite a toma do medicamento um sinal sonoro seria emitido por exemplo a cada 5 minutos.

Bibliografia

[1] "Erros de medicação causam 7 mil mortos - Sociedade - Correio da Manhã." [Online]. Available: https://www.cmjornal.pt/sociedade/detalhe/erros-de-medicacao-causam-7-mil-mortos. [Accessed: 28-Jan-2020].

[2] C. M. Allen EL, Barker KN, "Draft guidelines on preventable medication errors," 1992, 49.3: 640-8.

[3] "NCC MERP Index for Categorizing Medication Errors." [Online]. Available: https://www.nccmerp.org/sites/default/files/indexColor2001-06-12.pdf. [Accessed: 29-Jan-2020].

[4] "7 erros cometidos com os medicamentos e que podem ser fatais | Cuidamos." [Online]. Available: https://cuidamos.com/artigos/7-erros-cometidos-com-medicamentos-que-podem-ser-fatais. [Accessed: 29-Jan-2020].

[5] D. R. Mager, "Medicatio[€n Errors and the Home Care Patient," 2007 . Available: https://nursing.ceconnection.com/ovidfiles/00004045-200703000-00003.pdf. [Acessed: 20 - Jan - 2020].

[6] "About Medication Errors | NCC MERP." [Online]. Available: https://www.nccmerp.org/about-medication-errors. [Accessed: 28-Jan-2020].

[7] "Vital Health - A importância do uso correto e racional do medicamento." [Online]. Available: https://www.vitalhealth.pt/opiniao/3655-a-importância-do-uso-correto-e-racional-do-medicamento.html. [Accessed: 16-Nov-2020].

[8] "Medication errors a big problem after hospital discharge - Harvard Health Blog - Harvard Health Publishing." [Online]. Available: https://www.health.harvard.edu/blog/medication-errors-a-big-problem-after-hospital-discharge-201207095012. [Accessed: 21-Oct-2020].

[9] N. L. Hodges, H. A. Spiller, M. J. Casavant, T. Chounthirath, and G. A. Smith, "Non-health care facility medication errors resulting in serious medical outcomes," *Clin. Toxicol.*, vol. 56, no. 1, pp. 43–50, Jan. 2018, doi: 10.1080/15563650.2017.1337908.

[10] "Importance of Taking Medications Correctly – A Place for Mom." [Online]. Available: https://www.aplaceformom.com/planning-and-advice/articles/importance-of-taking-medications. [Accessed: 21-Oct-2020].

[11] "Automated Pill Dispensers for Independent Senior Life." [Online]. Available: https://www.medminder.com/. [Accessed: 30-Jan-2020].

[12] "medminder - Pesquisa Google." [Online]. Available: https://www.google.com/search?q=medminder&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjRnpHygaznAhXxTxUIHbuHBRMQ_AUoAXoECA0QAw&biw=767&bih=708&dpr=1.25#imgrc=1Zshu-uxb1ZxxM: [Accessed: 30-Jan-2020].

[13] "philips Medication Dispensing System – Pesquisa Google." [Online]. Available: https://www.google.com/search?q=philips+Medication+Dispensing+System&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjM_JDT4rPnAhWC2eAKHcBkBv4Q_AUoAXoECAsQAw&biw=1536&bih=722&dpr=1.25#imgrc=7tS7pc0uWA_BUM. [Accessed: 02-Feb-2020].

[14] "Hero Medication Dispenser, Manager, & Pill Organizer | Hero." [Online]. Available: https://herohealth.com/. [Accessed: 02-Feb-2020].

[15] "INRange Systems | Makers of RMMS & E-Kit." [Online]. Available: http://www.inrangesystems.com/. [Accessed: 02-Feb-2020].

[16] "31 Day Low Profile Monthly Organizer and Talking Pill Reminder." [Online]. Available: https://www.medcentersystems.com/Low-Profile-Monthly-Organizer-and-Reminder-p/7072-8combo(w70942).htm. [Accessed: 30-Jan-2020].

[17] "Hospedagem de sítios web – Wikipédia, a enciclopédia livre." [Online]. Available: https://en.wikipedia.org/wiki/Web_hosting_service. [Accessed: 04-Oct-2020].

[18] "Alojamento Web grátis com PHP, MySQL e cPanel, Sem Anúncios." [Online]. Available: https://pt.000webhost.com/. [Accessed: 10-Oct-2020].

[19] "O que é PHP? Guia Básico de Programação PHP." [Online]. Available: https://www.hostinger.pt/tutoriais/o-que-e-php-guia-basico/. [Accessed: 26-Oct-2020].

[20] "Server-side – Wikipédia, a enciclopédia livre." [Online]. Available: https://pt.wikipedia.org/wiki/Server-side. [Accessed: 01-Nov-2020].

[21] "Como funciona o PHP - Curso PHP Iniciante #02 - Gustavo Guanabara - YouTube." [Online]. Available: https://www.youtube.com/watch?v=Eup6utTPe2U&list=PLHz_AreHm4dm4beCCCmW4xwpmLf6EHY9k&index=2. [Accessed: 16-Nov-2020].

[22] "apache png – Pesquisa Google." [Online]. Available: https://www.google.com/search?q=apache+png&tbm=isch&hl=pt-PT&sa=X&ved=2ahUKEwjH-biUvIftAhWm2OAKHeOrDTgQBXoECAEQKQ&biw=1519&bih=722#imgrc=xmtsYEizfN2fZM. [Accessed: 16-Nov-2020].

[23] "What Do Client-Side and Server-Side Mean? | Client Side vs. Server Side | Cloudflare." [Online]. Available: https://www.cloudflare.com/learning/serverless/glossary/client-side-vs-server-side/. [Accessed: 01-Nov-2020].

[24] "Aplicação web – Wikipédia, a enciclopédia livre." [Online]. Available: https://en.wikipedia.org/wiki/Web_application. [Accessed: 26-Nov-2020].

[25] "Web Request - Source Defence." [Online]. Available: https://sourcedefense.com/glossary/web-request/. [Accessed: 26-Nov-2020].

[26] "Hypertext Transfer Protocol – Wikipédia, a enciclopédia livre." [Online]. Available: https://pt.wikipedia.org/wiki/Hypertext_Transfer_Protocol. [Accessed: 26-Nov-2020].

[27] "Métodos de requisição HTTP - HTTP | MDN." [Online]. Available: https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods. [Accessed: 26-Nov-2020].

[28] "Web service – Wikipédia, a enciclopédia livre." [Online]. Available: https://pt.wikipedia.org/wiki/Web_service. [Accessed: 29-Nov-2020].

[29] "What are web services? Definition from WhatIs.com." [Online]. Available: https://searchapparchitecture.techtarget.com/definition/Web-

services?_ga=2.49022566.1147559175.1606412877-2064580325.1606412877. [Accessed: 29-Nov-2020].

[30] "Web Services (O que é, motivos para uso, como funciona, protocolos SOAP/REST) - YouTube." [Online]. Available: https://www.youtube.com/watch?v=RlB5wcuFvcc. [Accessed: 26-Nov-2020].

[31] "phpMyAdmin 3.3.5 - O MySQL no browser - Pplware." [Online]. Available: https://pplware.sapo.pt/software/phpmyadmin-3-3-5-o-mysql-no-browser/. [Accessed: 24-Nov-2020].

[32] "phpMyAdmin – Wikipédia, a enciclopédia livre." [Online]. Available: https://en.wikipedia.org/wiki/PhpMyAdmin. [Accessed: 24-Nov-2020].

[33] "SQL – Wikipédia, a enciclopédia livre." [Online]. Available: https://en.wikipedia.org/wiki/SQL. [Accessed: 24-Nov-2020].

[34] "Você precisa saber o que é SQL! - { Dicas de Programação }." [Online]. Available: https://dicasdeprogramacao.com.br/o-que-e-sql/. [Accessed: 24-Nov-2020].

[35] "SQL | DDL, DQL, DML, DCL and TCL Commands - GeeksforGeeks." [Online]. Available: https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/. [Accessed: 24-Nov-2020].

[36] "Conheça o Android Studio | Desenvolvedores Android." [Online]. Available: https://developer.android.com/studio/intro. [Accessed: 01-Nov-2020].

[37] "What is Gradle in Android Studio? - Stack Overflow." [Online]. Available: https://stackoverflow.com/questions/16754643/what-is-gradle-in-android-studio. [Accessed: 01-Nov-2020].

[38] "Run apps on the Android Emulator | Android Developers." [Online]. Available: https://developer.android.com/studio/run/emulator. [Accessed: 01-Nov-2020].

[39] "Visão geral do manifesto do aplicativo | Desenvolvedores Android." [Online]. Available: https://developer.android.com/guide/topics/manifest/manifest-intro. [Accessed: 01-Nov-2020].

[40] "App Manifest Overview | Android Developers." [Online]. Available: https://developer.android.com/guide/topics/manifest/manifest-intro.

[Accessed: 01-Nov-2020].

[41] "High Level Storage Warehouse."

[42] "wiki:study:evps:hrs [wiki.erazor-zone.de]." [Online]. Available: http://wiki.erazor-zone.de/wiki:study:evps:hrs. [Accessed: 28-Jan-2020].

[43] "O que é um diagrama de sequência UML? | Lucidchart." [Online]. Available: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-sequencia-uml. [Accessed: 15-Nov-2020].

[44] "Diagrama de sequência – Wikipédia, a enciclopédia livre." [Online]. Available: https://pt.wikipedia.org/wiki/Diagrama_de_sequência. [Accessed: 15-Nov-2020].

[45] "Untitled - WebSequenceDiagrams." [Online]. Available: https://www.websequencediagrams.com/. [Accessed: 28-Nov-2020].

[46] "PHP – Wikipédia, a enciclopédia livre." [Online]. Available: https://en.wikipedia.org/wiki/PHP. [Accessed: 25-Oct-2020].

[47] "Java virtual machine - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Java_virtual_machine. [Accessed: 15-Nov-2020].

[48] "What is the JVM? Introducing the Java Virtual Machine | InfoWorld." [Online]. Available: https://www.infoworld.com/article/3272244/what-is-the-jvm-introducing-the-java-virtual-machine.html. [Accessed: 15-Nov-2020].

[49] "(50) NFC Explained: What is NFC? How NFC Works? Applications of NFC - YouTube." [Online]. Available: https://www.youtube.com/watch?v=eWPtt2hLnJk&t=379s. [Accessed: 29-Jan-2020].

[50] "JVM architecture with the selective reflective behaviour | Download Scientific Diagram." [Online]. Available: https://www.researchgate.net/figure/JVM-architecture-with-the-selective-reflective-behaviour_fig2_220830752. [Accessed: 15-Nov-2020].

[51] "Java Native Interface: JNI Example." [Online]. Available: https://www.protechtraining.com/blog/post/java-native-interface-jni-example-65. [Accessed: 31-Oct-2020].

[52] "Guide to JNI (Java Native Interface) | Baeldung." [Online]. Available: https://www.baeldung.com/jni. [Accessed: 31-Oct-2020].

[53] “Difference between Static and Shared libraries - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/difference-between-static-and-shared-libraries/. [Accessed: 01-Nov-2020].

[54] “Str_lab_7 (10).” .

[55] I. C. Society, “IEEE Standard for Software Verification and Validation,” *IEEE(Institute Electr. Electron. Eng.*, vol. 2004, no. June, 2005.

[56] “Verification and validation - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Verification_and_validation. [Accessed: 10-Nov-2020].

[57] “A Importância da validação e da verificação.” [Online]. Available: https://www.devmedia.com.br/a-importancia-da-validacao-e-da-verificacao/24559. [Accessed: 11-Nov-2020].

[58] “Verification and validation - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Verification_and_validation. [Accessed: 11-Nov-2020].

Anexos

Tabelas existentes na Base de dados e descrição das suas tarefas

Tabela	Coluna	Descrição
agenda	ProximaToma	Data da próxima toma da medicação
	Confirmação	0 se a medicação foi administrada 1 caso contrário.
Dados registados	ID	Número de identificação do utente
	Username	Nome de utilizador do utente
	name	Nome do utente
	apelido	Apelido do utente
	age	Idade do utente
	Email	Email do utente
	pass	Palavra-passe do utente
kit	XAZB	0 se a célula do kit estiver vazia ou o número de identificação do medicamento que se encontra na posição X=A e Z=B
medicação	ID	Número de identificação do medicamento
	name	Nome do medicamento
	barcode	Código de barras do medicamento caso exista, caso contrário é atribuído o valor 0.
time_medicacao	DataInicio	Data de início da medicação no formato aaaa-mm-dd.
	DataFim	Data da última administração da medicação no formato aaaa-mm-dd.
	HoraInicio	Hora da primeira administração da medicação no formato aaaa-mm-dd.
	DateTime	Data da primeira administração no formato aaaa-mm—dd hh:mm:ss.
	IntervaloTempo	Intervalo de tempo entre administrações da medicação.
	TempoAdmiDias	Tempo de administração em dias da medicação.
utente_medicacao	Utente_ID	Número de identificação do utente
	Medication_ID	ID da medicação atribuída ao utente com o respetivo número de identificação.

Funções implementadas para o comando do kit

Função	Descrição
getBitValue	Dado um valor byte a função retorna o valor do seu bit n
setBitValue	É atribuído ao byte n o valor dado em byte
moveXLeft/moveXRight	Move a gaiola para a esquerda/direita
stopX/stopY/stopZ	Para todos os movimentos da gaiola no eixo X/Y/Z.
getXPos/getZPos/getYPos	Retorna o valor da posição atual da gaiola no eixo X/Z ou Y.
moveZDown/moveZUp	Move a gaiola num movimento ascendente/ descendente no eixo Z.
move_y_inside/move_y_outside	Move a gaiola para dentro/fora no eixo Y.
has_piece	Verifica se a gaiola possui alguma peça.
is_at_x, is_at_y, is_at_z	Verifica se a gaiola se encontra em alguma das posições X,Y ou Z.
is_at_z_down(), is_at_z_up()	Verifica se a gaiola se encontra na posição superior ou inferior no eixo Z.
gotoUpLevel/ gotoLowLevel	Desloca a gaiola para a posição superior/inferior no eixo Z.
gotoX/ gotoY/ gotoZ	Desloca a gaiola para a posição X,Y,Z desejada.
callibrate	Calibração do kit para que a gaiola se encontre numa posição reconhecível.

putPiece_task/ getPiece_task	Coloca/retira uma peça de uma determinada posição.