

Universidade da Beira Interior

Departamento de Informática



Departamento de
Informática

Nº 21 - 2019: *Construction of a Home Intelligent Assistant Service Oriented Architecture for Smart Home Environments*

Elaborado por:

Miguel Bruno Mendonça Gregório

Orientador:

Professor Doutor Bruno M. C. Silva

22 de Junho de 2019

Agradecimentos

Agradeço aos meus pais por me terem dado a possibilidade de frequentar o curso de Engenharia Informática, e por me apoiarem em todas as situações, sempre com os melhores conselhos que só os pais conseguem dar.

Agradeço a todos os amigos que levo desta licenciatura, mas agradeço principalmente aos meus colegas de laboratório Tomás Jerónimo, João Amaral, e Mauro Julião pelo fantástico ambiente de trabalho e entreaajuda que desenvolvemos ao longo destes meses.

Gostaria de agradecer também à minha namorada, por todo o apoio que me deu ao longo destes meses, mas principalmente por me fazer acreditar mais em mim e nas minhas capacidades.

Last but not least, agradeço ao Professor Doutor Bruno Silva por me orientar e apoiar durante todo o desenvolvimento deste projeto.

Sozinhos vamos rápido, juntos vamos longe.

Obrigado !

Conteúdo

Conteúdo	iii
Lista de Figuras	v
1 Introdução	1
1.1 Enquadramento	1
1.2 Motivação	1
1.3 Objetivos	2
1.4 Organização do Documento	3
2 Estado da Arte	5
2.1 Interoperabilidade	5
2.1.1 Interoperabilidade no Contexto da Saúde	6
2.1.2 Vantagens da Interoperabilidade no Contexto na Saúde	6
2.1.3 Obstáculos da Interoperabilidade no Contexto da Saúde	8
2.2 Assistentes Pessoais Virtuais	8
2.3 Conclusões	10
3 Tecnologias e Ferramentas Utilizadas	11
3.1 Introdução	11
3.2 Microsoft Visual Studio	11
3.3 Microsoft Azure	11
3.4 SQL Server Management Studio	12
3.5 Sublime Text	12
3.6 Android Studio	12
3.7 Conclusões	12
4 Engenharia de Software	13
4.1 Introdução	13
4.2 Engenharia de Software - Arquitetura Baseada em Serviços	13
4.2.1 Requisitos Funcionais	13

4.2.2	Requisitos Não-Funcionais	14
4.2.3	Casos de Uso	15
4.2.4	Diagramas de Atividade	17
4.2.5	Base de Dados	23
4.3	Engenharia de Software - Aplicação Móvel de Consulta	25
4.3.1	Requisitos Funcionais	25
4.3.2	Requisitos Não Funcionais	25
4.3.3	Casos de Uso	25
4.3.4	Diagrama de Atividade	26
4.4	Conclusões	27
5	Implementação	29
5.1	Introdução	29
5.2	Arquitetura do Sistema	29
5.3	Estrutura da Arquitetura Baseada em Serviços	31
5.3.1	Modelos de Dados (<i>Models</i>)	31
5.3.2	Serviços (<i>Controllers</i>)	32
5.3.3	Base de Dados	35
5.4	<i>Layer</i> de Registo	36
5.5	Aplicação Móvel	37
5.6	Conclusões	38
6	Demonstração e Validação da Arquitetura Baseada em Serviços	39
6.1	Introdução	39
6.2	<i>Personas</i>	39
6.2.1	<i>Persona 1</i> - Rui Marques	39
6.2.2	<i>Persona 2</i> - Ana Ferreira	41
6.2.3	<i>Persona 3</i> - Jorge Almeida	42
6.2.4	Outras <i>personas</i>	43
6.3	Demonstração e Validação	43
6.4	conclusões	46
7	Conclusões e Trabalho Futuro	47
7.1	Conclusões Principais	47
7.2	Trabalho Futuro	47
	Bibliografia	49

Lista de Figuras

2.1	Conceito de Interoperabilidade.	5
2.2	Interface da aplicação SmartTings Samsung [6].	9
4.1	Diagrama de casos de uso da arquitetura baseada em serviços. . .	16
4.2	Diagrama de atividade da interface da <i>layer</i> de registo e respetivo serviço.	17
4.3	Diagrama de atividade do serviço de <i>login</i>	18
4.4	Diagrama de atividade do serviço de inserção e atualização de dados. .	19
4.5	Diagrama de atividade do serviço de inserção de dados captados por bandas corporais.	20
4.6	Diagrama de atividade do serviço de seleção de dados captados por bandas corporais.	20
4.7	Diagrama de atividade do serviço de inserção de dados relativos a um evento de emergência.	21
4.8	Diagramas de atividade dos serviços de inserção de dados de tempo real.	22
4.9	Diagrama de atividade do serviço de seleção de utilizadores e da- dos associados a eles.	22
4.10	Esquema relacional da base de dados.	24
4.11	Diagrama de casos de uso da aplicação móvel.	26
4.12	Diagrama de atividade da aplicação móvel.	27
5.1	Esquema da arquitetura do sistema.	30
5.2	Interface de registo <i>Home Intelligent Assistant</i> (HIA).	36
5.3	Interface da aplicação.	38
6.1	<i>Persona</i> Rui Marques.	39
6.2	Registo da <i>persona</i> Rui na Base de Dados (BD).	40
6.3	Registo e informações <i>persona</i> Rui.	40
6.4	<i>Persona</i> Ana Ferreira.	41
6.5	Registo do evento de emergência da <i>persona</i> Ana na BD.	41

6.6	Registo dos dados de saúde da <i>persona</i> Ana na BD.	41
6.7	<i>Persona</i> Jorge Almeida.	42
6.8	Registo das leituras de bandas corporais da <i>persona</i> Jorge na BD. .	42
6.9	Registo de eventos de emergência da <i>persona</i> Jorge na BD.	42
6.10	Outras <i>persnonas</i> presentes neste sistema.	43
6.11	Associação dos serviços Web da arquitetura do sistema às dife- rentes <i>personas</i>	45

Lista de Excertos de Código

5.1	String de conexão à base de dados.	29
5.2	Declaração do modelo de dados "SOSEvent".	32
5.3	Serviço que adiciona um utilizador à BD.	33
5.4	Serviço que devolve informação acerca das leituras efetuadas por bandas corporais.	34
5.5	Criação da tabela tb_User na BD.	35
5.6	Chama do serviço que adiciona um utilizador na BD na <i>layer</i> de registo.	37
5.7	<i>Intent</i> de transferência de dados.	38

Acrónimos

AI	<i>Artificial Intelligence</i>
BD	Base de Dados
BLOB	<i>Basic Large Object</i>
BPM	Batimento (Cardíaco) por Minuto
CSS	<i>Cascading Style Sheets</i>
ECG	Eletrocardiograma
HIA	<i>Home Intelligent Assistant</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IDE	<i>Integrated Development Environment</i>
IoT	<i>Internet of Things</i>
JSON	<i>JavaScript Object Notation</i>
NAHIT	<i>National Alliance for Health Information Technology</i>
SHA-256	<i>Secure Hash Algorithm-256</i>
SQL	<i>Structured Query Language</i>

Capítulo 1

Introdução

1.1 Enquadramento

Nos dias de hoje, a grande maioria das pessoas têm na sua posse um *smartphone* com acesso à *Internet*, seja por *Wi-Fi*, dados móveis, ou pelos diversos pontos *Wi-Fi* disponíveis nas cidades. Em Portugal, por exemplo, verificou-se que mais de dois terços da população tem um *smartphone* [1].

O crescimento e a popularização dos dispositivos móveis, potenciou a utilização de Assistentes Pessoais Virtuais. Um estudo realizado pela Gartner, uma das mais importantes empresas de consultadoria na área da tecnologia, concluiu que os assistentes pessoais virtuais serão a grande tendência do mercado nos próximos anos [3].

Assistentes Pessoais Virtuais também se tornaram populares na área de *Smart Homes*. Estes sistemas são denominados de Assistentes de Casa Inteligentes (*Home Intelligent Assistant* (HIA))[2] e tipicamente integram uma arquitetura baseada em serviços. Estas arquiteturas incluem vários tipos de sistemas e aplicações, porém a grande maioria dos HIAs são resistentes ou incapazes de integrar novas soluções que não sejam originais dos seus fabricantes.

Este projeto vem responder diretamente à necessidade de uniformizar um leque de arquiteturas existentes, integrando-as num ecossistema único que englobe áreas como *smart health*, *smart home* e que seja capaz de integrar diferentes soluções.

1.2 Motivação

Existe no mercado, a necessidade de soluções integradas na área das tecnologias e sistemas da informação. Soluções interoperáveis que acompanhem o dia-a-dia do utilizador, e que sejam o máximo independentes possível. É também

necessário que essas soluções façam uma gestão de dados eficaz, sem perdas nem redundâncias. Penso que foram esses mesmos motivos que me levaram a escolher este projeto: simplificar a inserção e a consulta de dados, criando uma arquitetura baseada em serviços o mais unificada possível. O processo de *sign up* e *login* será também simplificado, uma vez que cada utilizador terá apenas uma conta com a qual poderá entrar nas várias aplicações.

Todas estas vantagens farão do HIA, um sistema mais unificado, sem dados redundantes, otimizando os recursos.

1.3 Objetivos

O principal objetivo deste projeto é a criação de uma arquitetura baseada em serviços para ecossistemas de casas inteligentes (*Smart Homes*), em particular para um HIA, que é um sistema que integra vários e diferentes serviços e/ou aplicações. É apresentada de seguida, uma lista mais detalhada dos objetivos deste projeto:

- Fazer um breve estudo do estado da arte em relação às soluções e arquiteturas já existentes, e aos conceitos base do projeto.
- Fazer um estudo sobre a arquitetura do sistema: analisar os requisitos necessários e das linguagens a utilizar.
- Construção e integração de serviços responsáveis por receber dados provenientes de várias aplicações e sensores que integram o HIA, guardando-os em Base de Dados (BD);
- Construção e integração de serviços responsáveis por responder aos pedidos das várias aplicações que integram o HIA devolvendo dados alojados na BD.
- Construção um serviço de *sign up* e *login* único, para as várias aplicações integradas no HIA, e respetiva interface gráfica.
- Construção uma aplicação móvel *Android*, capaz de consultar dados de cada utilizador.
- Elaborar e demonstrar variados testes que validão a arquitetura baseada em serviços para HIAs.

1.4 Organização do Documento

De modo a refletir o trabalho que foi realizado, este documento encontra-se estruturado da seguinte forma:

1. O primeiro capítulo – **Introdução** – apresenta o projeto, a motivação para a sua escolha, o enquadramento para o mesmo, os seus objetivos e a respetiva organização do documento.
2. O segundo capítulo – **Estado da Arte** – descreve os conceitos mais importantes no âmbito deste projeto como *Smart Health* e Interoperabilidade, Apresenta também vários Assistentes Pessoais Virtuais.
3. O terceiro capítulo – **Tecnologias e Ferramentas Utilizadas** – Apresenta uma breve descrição de cada *software* utilizado no desenvolvimento deste projeto, bem como em que área foi usado cada um.
4. O quarto capítulo – **Engenharia de Software** – descreve os requisitos funcionais e não funcionais, os casos de uso, diagramas de atividade e diagramas de classe da arquitetura baseada em sistemas, e da aplicação móvel. É também descrita a BD que integra o projeto.
5. O quinto capítulo – **Implementação** – contém toda a informação sobre a construção de todo o projeto. Aqui são também apresentados os excertos de código mais relevantes do projeto.
6. O sexto capítulo – **Demonstração e Validação da Arquitetura Baseada em Serviços** – apresenta cenários de teste e validação do ecossistema HIA. Este capítulo demonstra a validação de toda a arquitetura de sistema desenvolvida.
7. O sétimo capítulo – **Conclusões e Trabalho Futuro** – detalha as conclusões retiradas durante o desenvolvimento do projeto, e sugere melhorias a implementar no futuro.

Capítulo 2

Estado da Arte

2.1 Interoperabilidade

Interoperabilidade é a capacidade de diferentes tecnologias de informação comunicarem entre si, trocarem dados de forma consistente e eficaz, e usarem esses dados em benefício do seu próprio sistema.

Na figura abaixo 2.1 está representado o ponto chave: assume-se que as aplicações à esquerda têm informações necessárias à aplicação à direita, e que neste sistema a informação é acessível entre as várias aplicações. A interoperabilidade será alcançada se a aplicação recetora conseguir entender corretamente o significado da informação que recebe e se conseguir usá-la [14].

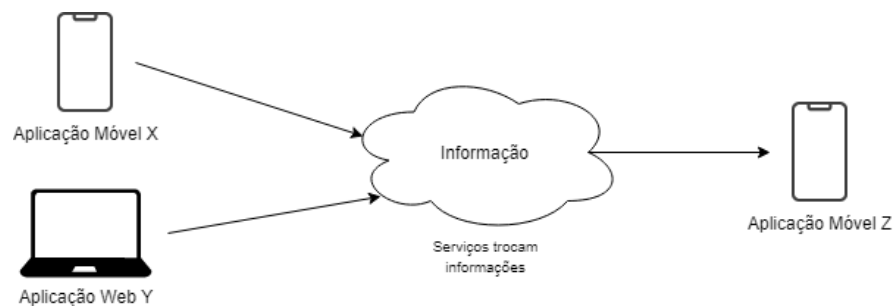


Figura 2.1: Conceito de Interoperabilidade.

2.1.1 Interoperabilidade no Contexto da Saúde

No contexto da saúde, não existe nenhuma definição padrão de interoperabilidade. No entanto, a *National Alliance for Health Information Technology* (NAHIT) define interoperabilidade como a capacidade de diferentes sistemas de tecnologia da informação e aplicações comunicarem entre si, trocando dados com máxima precisão e eficácia. É também necessário que o sistema recetor consiga interpretar e usar corretamente os dados que recebe [16].

Uma *Smart Health Technologie* - uma tecnologia inteligente na área da saúde - terá de ser capaz de guardar informações de saúde a partir de sensores cada vez mais avançados, armazenando-as e compreendendo-as. Deverá também de ser capaz de fornecer conselhos personalizados ou ações a realizar a partir dos dados recolhidos. A este conceito tecnológico que faz uso da interoperabilidade entre várias tecnologias dá-se o nome de *Smart Health* [17].

Neste contexto da saúde, a interoperabilidade pode ser dividida em várias categorias, tais como a interoperabilidade de informações trocadas entre aplicações de saúde, a interoperabilidade de registos eletrónicos de saúde (sensores) e interoperabilidade entre clínicas, hospitais e centros de saúde, onde são partilhados dados, resultados de exames, diagnósticos, entre outras.

2.1.2 Vantagens da Interoperabilidade no Contexto na Saúde

O domínio da saúde está atualmente a passar por uma mudança fundamental na sua abordagem à prestação de cuidados, e os sistemas de tecnologias da informação estão a tornar-se elementos indispensáveis da saúde. No entanto, continuam a existir ineficiências incessantes e falhas de qualidade vividas por profissionais de saúde e pacientes. Existe então a necessidade de entender o papel que a interoperabilidade desempenha na partilha de dados e o uso que várias aplicações darão ao mesmo dado.

É apresentada, de seguida, uma lista de benefícios da interoperabilidade na saúde[18]:

- **Fácil acesso aos registos dos pacientes:** geralmente, os pacientes recebem cuidados de uma ampla gama de prestadores de saúde (hospitais, laboratórios, farmácias, clínicas, centros de saúde, médicos individuais, entre outros). Isto leva a uma fragmentação da informação dos pacientes nos vários prestadores de cuidados de saúde. No entanto, num sistema interoperável, os médicos podem ter acesso aos registos dos pacientes, estando estes armazenados em sistemas heterogéneos. Isto irá melhorar os processos de saúde, dando ao prestador de cuidados de saúde as informações específicas do paciente que precisa de consultar de uma forma eficaz. Além disso, com a interoperabilidade completa nesta área, os pacientes podem ter também

acesso total aos seus registos médicos, o que permite uma melhor gestão da sua saúde.

- **Fácil compreensão de termos médicos:** os prestadores de cuidados de saúde conseguirão entender melhor os termos e conceitos que são transmitidos de um sistema para o outro. Neste caso, a interoperabilidade garante que serão usadas as mesmas terminologias médicas em todos os sistemas de comunicação. Por isso, os prestadores de serviços de saúde podem facilmente analisar os dados de todos os sistemas que colaboram para o diagnóstico do paciente.
- **Redução de erros médicos:** As receitas passadas por médicos deveriam ter em conta acontecimentos passados, receitas anteriores, alergias, entre outros. Regularmente este processo é feito com erros e desrespeitando algumas restrições dos pacientes. Por exemplo, um estudo realizado em ambiente hospitalar estima que 18% dos erros médicos que incluem administração de medicamentos devem-se à inadequada disponibilidade de informações do paciente. Assim, os erros médicos são uma grande preocupação para a área da saúde, uma vez que é a sexta maior causa de morte em hospitais [12][15]. Uma das maneiras de evitar erros médicos é assegurar a interoperabilidade completa na área da saúde, garantindo que os dados relacionados com a saúde de cada paciente são formatados de forma a permitir que os diferentes sistemas de informação entendam tanto a estrutura como o conteúdo das informações partilhadas.
- **Redução dos custos relacionados com a saúde:** Um dos principais desafios que o setor de saúde enfrenta é o aumento dos custos associados a cada utente [12]. Por exemplo, se os vários sistemas de saúde dos Estados Unidos da América operassem em interoperabilidade, seria entregue ao estado uma poupança de 77,8 mil milhões de dólares por ano [12].
- **Integração de diferentes registos de saúde:** São produzidas enormes quantidades de dados nos diversos subsistemas de saúde tais como laboratórios, hospitais, clínicas, aplicações móveis, *softwares* relacionados com a saúde, entre outros. No entanto, integrar na arquitetura informações de aplicações autonomamente desenvolvidas é uma tarefa difícil, uma vez que o objetivo inicial de quem as desenvolve não é a cooperação e a partilha de dados. Atualmente, uma arquitetura que englobe vários sistemas de informação requer portabilidade, colaboração, e integridade de dados. A interoperabilidade garante que os vários sistemas dentro da mesma arquitetura se integrem perfeitamente uns aos outros, construindo uma arquitetura ideal.

- **Melhorar o apoio ao tratamento de doenças crónicas:** Na sua maioria das vezes, o tratamento de doenças crónicas envolve vários médicos e profissionais de saúde. Um sistema interoperável tornará mais fácil para os prestadores de saúde detetar problemas graves, consultado o histórico de outros pacientes com os mesmos sintomas ou as mesmas doenças.

2.1.3 Obstáculos da Interoperabilidade no Contexto da Saúde

Não há dúvida de que a interoperabilidade tem um grande impacto positivo na saúde. No entanto, a falta de interoperabilidade dos sistemas e serviços de saúde tem sido identificada como uma das principais falhas na área da saúde. Por exemplo, um médico com consultório particular pode ter dificuldade em obter informações completas sobre um paciente seu que está hospitalizado; também esse mesmo médico pode repetir exames, consultas ou outros procedimentos que tenham sido efetuados num hospital público, uma vez que ele não tem informação prévia sobre o paciente.

Consequentemente, esta secção avalia as barreiras que impedem a interoperabilidade nos cuidados de saúde[18]:

- **Complexidade no domínio da saúde:** O domínio da saúde é muito complexo uma vez que envolve muitos atores, como médicos, enfermeiros, farmacêuticos, técnicos de laboratório que colaborativamente participam no tratamento dos pacientes. Cada um desses atores gera informações que são necessárias para outros atores. No entanto, alguns prestadores de cuidados de saúde podem não estar dispostos a partilhar informações relacionadas à saúde de algum paciente.
- **Diferentes terminologias:** Um problema com o qual nos podemos deparar é o facto de serem usados diferentes conceitos, nomenclaturas ou ontologias, para situações iguais ou semelhantes. O crescimento das terminologias e ontologias no domínio da saúde é exponencial. Um sistema interoperável dificilmente conseguirá compreender diferentes nomenclaturas e chegar à conclusão de que têm o mesmo significado.
- **Resistência à mudança:** existe uma falta de uniformidade dos sistemas de informação dos diferentes prestadores de saúde. Existe também uma falta de confiança relativa à privacidade e segurança dos dados.

2.2 Assistentes Pessoais Virtuais

Um assistente pessoal virtual é uma ajuda virtual capaz de monitorizar e aconselhar o utilizador. Atualmente, os assistentes pessoais virtuais são capazes de

executar tarefas com mais agilidade, precisão, e até afeto, do que um ser humano comum, garantindo ao consumidor um atendimento personalizado e uma experiência única [7]. De seguida, são apresentados alguns dos mais recentes e mais usados assistentes pessoais virtuais:

- **Google Assistant:** é possível usar este assistente através do dispositivo *Google Home* ou através do *smartphone* pessoal. Com ele é possível criar notas e lembretes, fazer traduções, obter direções através do mapa, controlar aparelhos *smart* domésticos, efectuar chamadas, responder a qualquer pergunta, entre outras funcionalidades [11].
- **SmartThings Samsung:** é um assistente integrado numa aplicação móvel. Com ele é possível controlar todos os dispositivos que normalmente estão presentes numa casa: televisão, ar condicionado, máquina de lavar e secar roupa, *robot* aspirador, frigorífico, luzes, sensores, alarme, entre outros [6]. Na seguinte figuras (2.2) é possível visualizar a interface da aplicação.

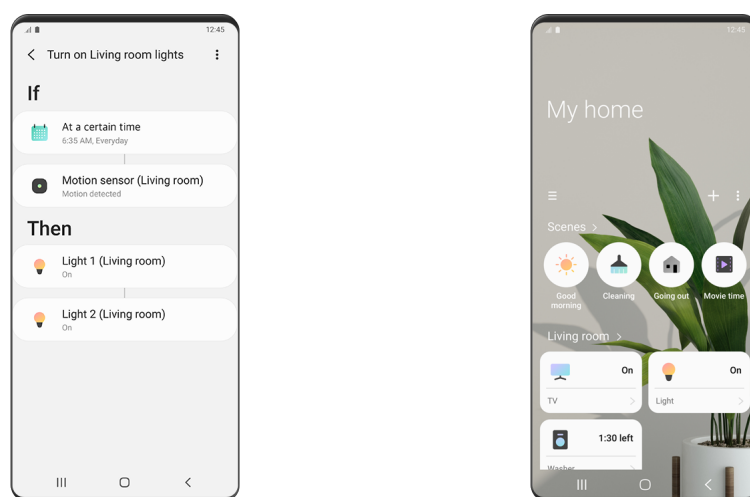


Figura 2.2: Interface da aplicação SmartThings Samsung [6].

- **CareAngel:** Mais direccionado para a área da saúde, este assistente virtual é capaz de manter uma conversa com o utilizador, fazendo-lhe perguntas chave que irão detetar o estado de saúde do utente. Todas essas informações são guardadas em tempo real, e, através da aplicação ou da plataforma *online* é possível alterar alguns dados, consultar diagnósticos clínicos, consultar listas de medicamentos a tomar, entre outras opções [4].

2.3 Conclusões

A usabilidade eficaz de tecnologias da informação nesta área aliada à interoperabilidade entre todas as tecnologias, depende do quão bem as tecnologias forem concebidas a nível da interação humano-máquina e máquina-máquina. Será também necessário que organizações como hospitais e centros de saúde estejam disponíveis para uma mudança que substitui os registos manuscritos e as pilhas de papel em informação eletrónica.

O estudo nesta secção foi alinhado com o objetivo principal deste projeto, construir uma arquitetura baseada em serviços para HIAs na área de *Smart Health* capaz de incluir todas as vantagens mencionadas em cima (2.1.2), e tentando também contornar os obstáculos apresentados anteriormente (2.1.3).

Capítulo 3

Tecnologias e Ferramentas Utilizadas

3.1 Introdução

Este capítulo descreve as ferramentas e tecnologias utilizadas no desenvolvimento deste projeto, com o objetivo de melhorar a percepção de como o mesmo foi desenvolvido.

3.2 Microsoft Visual Studio

O *Microsoft Visual Studio* [9] é um ambiente de desenvolvimento integrado (*Integrated Development Environment* (IDE)) da Microsoft, para desenvolvimento de software incluído no *.NET Framework*, e dedicado a linguagens de programação tais como C, C++, C#, entre outras. É também uma ferramenta de desenvolvimento na área *web*, nomeadamente na criação de aplicações e serviços *web*, aplicações móveis, entre outras.

Para este projeto, a partir do *Microsoft Visual Studio*, foi criada uma arquitetura baseada em serviços, mais concretamente uma *ASP.NET Core Web Application* em C#, capaz de realizar a comunicação entre o cliente e o servidor.

3.3 Microsoft Azure

O *Microsoft Azure* [10] é uma plataforma integrada de serviços na *cloud*, tais como computação, Bases de Dados, *Internet of Things* (IoT), *Artificial Intelligence* (AI) + *Machine Learning*, redes, entre outros.

Neste projeto, o *Microsoft Azure* foi utilizado para criar um servidor, integrar uma BD, e guardá-lo na *cloud*. Foi também usado para guardar a *web application* desenvolvida.

3.4 SQL Server Management Studio

O *SQL Server Management Studio* [8] é um software usado para configurar, gerir e administrar todos os componentes do Microsoft SQL Server. Nele é possível atuar sobre qualquer objeto presente no servidor, nomeadamente nas Bases de Dados.

Neste projeto, foi usado, maioritariamente para criar tabelas *Structured Query Language* (SQL) e introduzir e editar dados nelas. Foi utilizado para realizar testes a toda a arquitetura deste projeto.

3.5 Sublime Text

O *Sublime Text* é um editor de texto, maioritariamente usado por desenvolvedores para criar e editar código.

Foi usado neste projeto para desenvolver a interface de registo do utilizador (*HyperText Markup Language* (HTML)), e respetivos ficheiros *Cascading Style Sheets* (CSS) e *JavaScript*.

3.6 Android Studio

O *Android Studio* é um IDE da *Google* destinado ao desenvolvimento de aplicações para o sistema operativo *Android* [5]. Possui também um *emulador Android* para que seja possível testar as aplicações desenvolvidas, bem como um editor visual de *layout*.

Neste projeto, foi usado para desenvolver a aplicação de consulta de dados, e o *layout* da mesma. O *emulador* acabou por não ser usado, uma vez que se torna mais rápido e eficaz testar num *smartphone* real - foi usado um *Xiaomi Mi8*.

3.7 Conclusões

Neste capítulo foram descritos todos os *softwares* utilizados no decorrer do desenvolvimento deste projeto, bem como as linguagens que estão associadas a eles.

Capítulo 4

Engenharia de Software

4.1 Introdução

Neste capítulo serão descritas as vertentes de engenharia de software do projeto, como os requisitos funcionais e não funcionais, a base de dados, e os casos de uso. Este capítulo será dividido entre a engenharia de software da arquitetura baseada em serviços, e a engenharia de software relacionada com a aplicação de consulta de dados.

A função principal da arquitetura desenvolvida é a consulta e inserção de dados efetuada através de várias aplicações. Para respeitar essa heterogeneidade, será necessária uma arquitetura capaz de lidar com diferentes Sistemas Operativos, com aplicações desenvolvidas em diferentes linguagens, bem como vários *requests* e *responses* a ocorrer simultaneamente [19].

4.2 Engenharia de Software - Arquitetura Baseada em Serviços

4.2.1 Requisitos Funcionais

Os requisitos funcionais representam todas as funcionalidades da arquitetura: definem o que a arquitetura é capaz de executar em termos de serviços. No caso deste projeto, os requisitos funcionais são os seguintes:

- Possuir um serviço de registo do utilizador na BD, através de email e *password*. Informações básicas do utilizador como morada, primeiro e último nome, número de telemóvel, género e data de nascimento também deverão ser guardadas com este serviço.

- Possuir um serviço que irá recolher à BD as informações básicas relativas a cada utilizador.
- Ter um serviço que confirme o *login*. Receberá um email e uma password, e deverá procurar na base de dados, se o email existe, e se a sua password corresponde à recebida. Deverá depois devolver o resultado.
- Possuir um serviço onde será possível introduzir dados na BD relativos à informação básica de saúde de cada utilizador, tais como: peso, altura, tipo sanguíneo, alergias, diabetes, epilepsia, asma, e outras informações que o utilizador considere necessárias.
- Ter um serviço que irá à BD recolher as informações básicas de saúde relativas a cada utilizador.
- Ter um serviço onde será possível guardar na BD informações recolhidas por uma aplicação móvel que recolhe dados através de leituras de bandas corporais. Na BD será guardada uma matriz por cada leitura efetuada, num determinado intervalo de tempo.
- Possuir um serviço capaz de recolher da BD as informações de cada utilizador, referentes aos dados recolhidos pela aplicação mencionada acima.
- Possuir um serviço que guarde na BD todas as informações relativas a situações de emergência de cada utilizador. Estas informações serão recolhidas através da aplicação própria para o efeito.
- Ter um serviço capaz de fazer uma consulta à BD acerca das situações de emergências passadas de cada utilizador.
- Possuir um serviço que guardará na BD, em tempo real, dados de cada utilizador, tais como temperatura corporal, Batimento (Cardíaco) por Minuto (BPM), coordenadas de localização, entre outros.
- Ter um serviço onde será possível aceder aos dados de tempo real de cada utilizador.

4.2.2 Requisitos Não-Funcionais

Os requisitos não-funcionais deste arquitetura dividem-se em requisitos de disponibilidade e de eficiência.

Requisitos de Disponibilidade

- Os vários serviços da arquitetura deverão funcionar sempre que alguma aplicação os chamar - para isso terão de estar permanentemente alojados numa plataforma de serviços *cloud*.
- A presença de múltiplas aplicações neste sistema, implica que este terá de ser capaz de fornecer vários serviços simultaneamente - concorrência de recursos/serviços [19].
- A arquitetura deverá estar preparada para enviar e consultar dados da BD em qualquer altura, com a exceção de falha na cobertura de rede.

Requisitos de Eficiência

Todos os serviços terão de responder aos pedidos efetuados pelas aplicações o mais rapidamente possível, não podendo exceder os 5 segundos em condições de rede consideradas normais.

4.2.3 Casos de Uso

- Registar utilizador.
- Confirmar dados de *login* do utilizador.
- Guardar dados básicos do utilizador.
- Consultar dados básicos do utilizador.
- Guardar dados de saúde do utilizador.
- Consultar dados de saúde do utilizador.
- Guardar dados da leitura de bandas corporais do utilizador.
- Consultar dados da leitura de bandas corporais do utilizador.
- Guardar dados relativos a eventos de emergência do utilizador.
- Consultar dados relativos a eventos de emergência do utilizador.
- Guardar dados de tempo real relativos a cada utilizador.
- Consultar dados de tempo real relativos a cada utilizador.

Diagrama de Casos de Uso

De seguida (4.1), é apresentado o diagrama de casos de uso da arquitetura baseada em serviços:

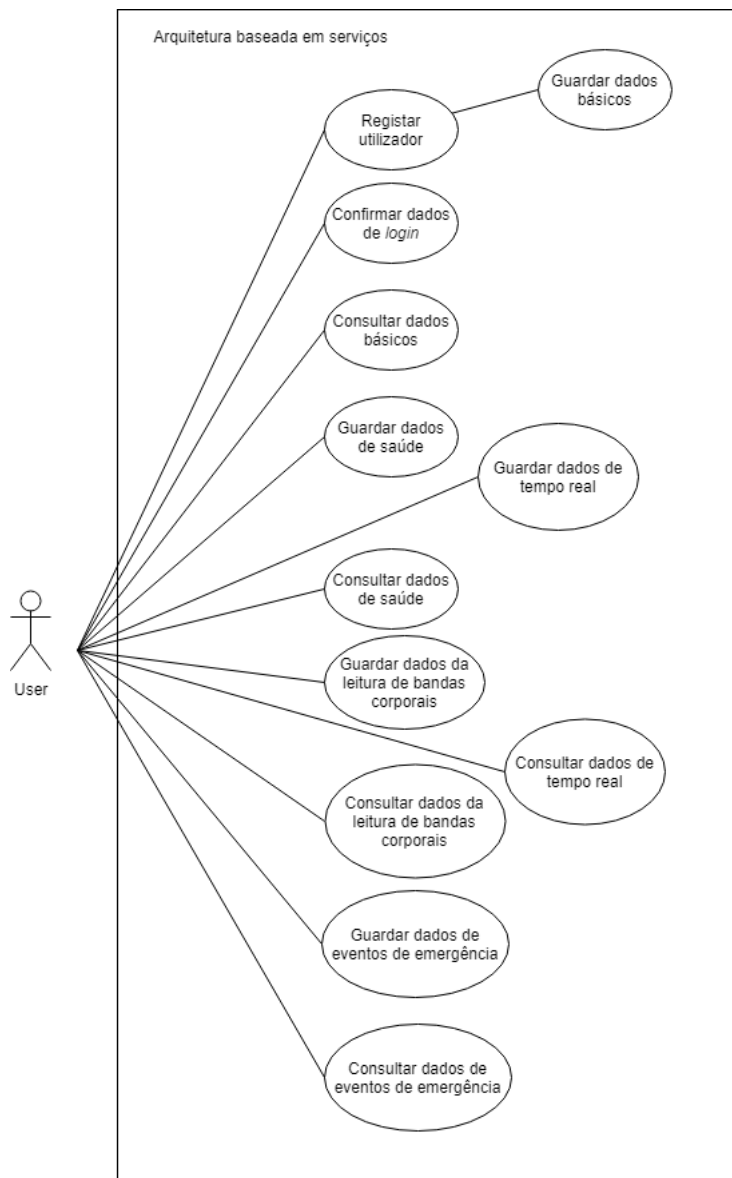


Figura 4.1: Diagrama de casos de uso da arquitetura baseada em serviços.

4.2.4 Diagramas de Atividade

Todas as aplicações que fazem parte do HIA terão um registo único. Os utilizadores apenas terão de se registar uma vez, e poderão depois usufruir de todas as outras aplicações, usando o mesmo *login*. No diagrama seguinte (4.2), é apresentado o diagrama de atividade de registo no grupo HIA. Inicialmente o utilizador preenche os dados de registo pedidos na interface de registo. Ao carregar em "registar", caso os dados introduzidos sejam válidos, são enviados para o serviço de registo. Aí, é executado um comando SQL que verifica se o *email* já consta na BD. Se não constar, os dados serão introduzidos na BD.

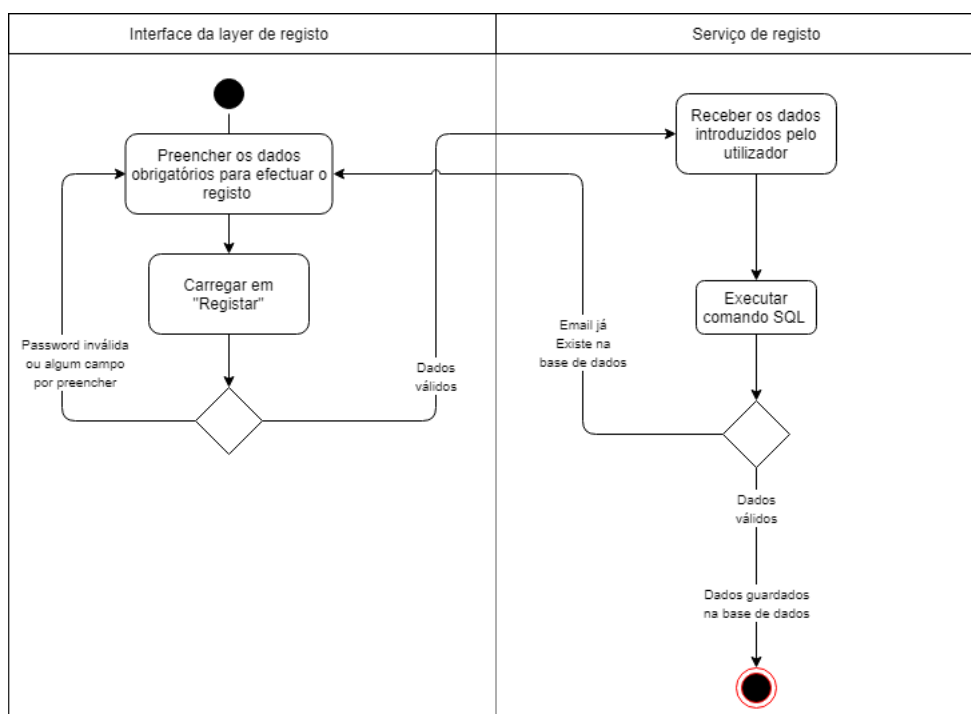


Figura 4.2: Diagrama de atividade da interface da *layer* de registo e respetivo serviço.

O próximo diagrama (4.3) apresenta todo o processo necessário para uma aplicação efetuar o *login* do utilizador. Este serviço recebe um *email* e uma *password*. De seguida irá executar um comando SQL que procura na BD se o *email* existe, e se a *password* que lhe está associada é a mesma que foi fornecida. Em caso afirmativo, o serviço devolverá informação básica e informação de saúde relativa ao utilizador que possui aquele *email*, em formato *JavaScript Object Notation* (JSON). Caso contrario, o serviço apenas termina, não devolvendo qualquer informação.

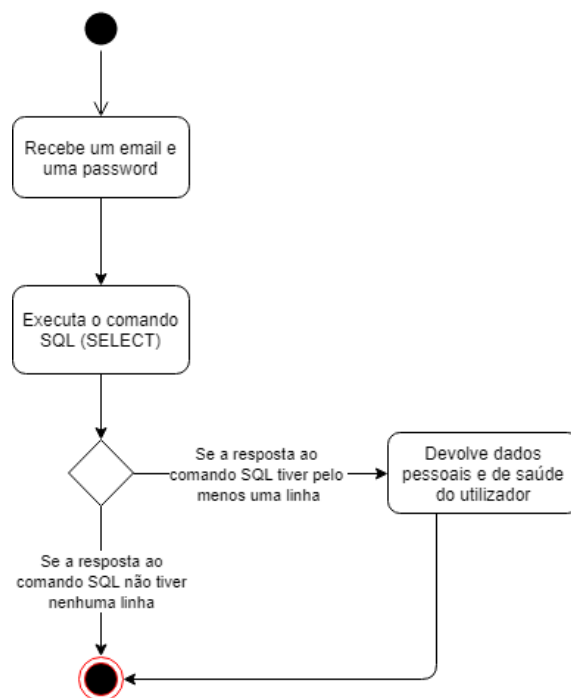


Figura 4.3: Diagrama de atividade do serviço de *login*.

Neste próximo diagrama (4.4), é apresentado o processo necessário para introduzir ou atualizar alguns dados pessoais e de saúde, tais como o número de telemóvel, morada, peso, altura, tipo sanguíneo, alergias, entre outros. Inicialmente, o serviço recebe da aplicação, todos os dados que pretende inserir ou atualizar na base de dados, bem como o *email* do utilizador que possui esses mesmos dados. De seguida, o serviço executa um comando SQL que é responsável por adicionar ou alterar os dados do respetivo utilizador. Caso o comando seja executado com sucesso, o serviço devolverá uma *flag* com o valor "1". Caso contrário, a *flag* será devolvida com o valor "0".

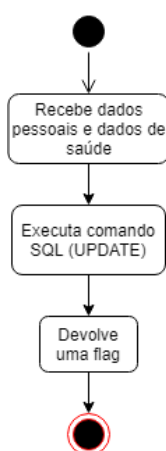


Figura 4.4: Diagrama de atividade do serviço de inserção e atualização de dados.

O próximo diagrama (4.5) descreve o serviço que é responsável por adicionar à BD os dados relativos às leituras efetuadas por bandas corporais. O serviço começa por receber os dados referentes a uma leitura (uma matriz em formato *Basic Large Object* (BLOB)) , bem como o *email* do utilizador ao qual os dados pertencem, e a data em que foram captados. De seguida é executado um comando de inserção SQL, com o objetivo de adicionar à BD os dados recebidos, bem como o *email* e a data. Caso o comando seja executado com sucesso, o serviço devolverá uma *flag* com o valor "1". Caso contrário, a *flag* será devolvida com o valor "0".

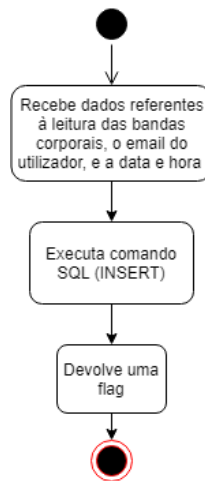


Figura 4.5: Diagrama de atividade do serviço de inserção de dados captados por bandas corporais.

O diagrama que se segue (4.6) descreve o serviço responsável por devolver dados relativos às leituras efetuadas por bandas corporais. O serviço recebe o *email* do utilizador e a data pretendida para consultar uma leitura. De seguida é executado um comando de seleção SQL que irá à BD procurar por ocorrências que contenham o *email* e a data fornecidos ao serviço. Por fim, essas ocorrências são guardadas no serviço e enviadas para o cliente em formato JSON. Caso não sejam encontradas ocorrências, será apenas devolvido um JSON vazio.

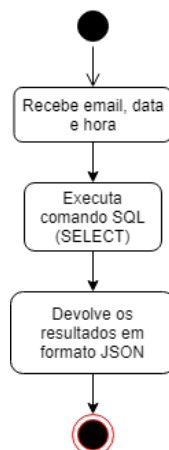


Figura 4.6: Diagrama de atividade do serviço de seleção de dados captados por bandas corporais.

O diagrama seguinte (4.7) apresenta o serviço que é responsável por adicionar à BD os dados relativos a um evento de emergência: o *email* do utilizador, a data do evento, o nome do operador responsável pelo evento, e o relatório. Estes dados serão enviados para o serviço. De seguida, é executado um comando SQL com o objetivo de os adicionar os dados a uma tabela da BD que guarda todas as situações de emergência já ocorridas. Caso a inserção seja efetuada com sucesso, o serviço devolverá uma *flag* com o valor "1". Caso contrario, a *flag* devolverá o valor "0".

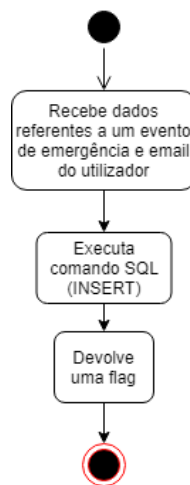


Figura 4.7: Diagrama de atividade do serviço de inserção de dados relativos a um evento de emergência.

Os diagramas que se seguem (4.8) descrevem os serviços responsáveis por adicionar à BD valores que são lidos em tempo real, tais como: latitude, longitude, BPM e temperatura. Inicialmente, cada serviço recebe o seu respetivo valor, seguido do *email* do utilizador. De seguida é executado um comando SQL com o objetivo de atualizar o correspondente dado na BD. Por fim, é devolvida uma *flag* que contém o valor "1" caso o comando seja executado com sucesso, e "0" caso falhe.

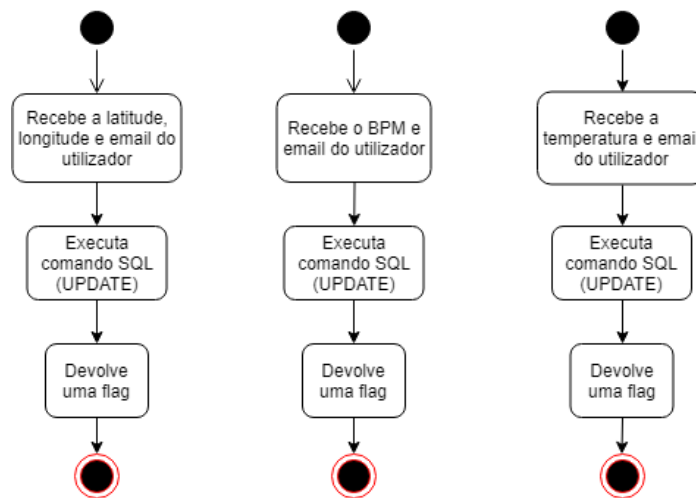


Figura 4.8: Diagramas de atividade dos serviços de inserção de dados de tempo real.

O próximo diagrama (4.9) descreve o serviço responsável por devolver informação pessoal, informação de saúde e informação captada em tempo real de todos os utilizadores registados. Como o serviço não recebe qualquer informação inicial, é apenas executado um comando SQL que seleciona todos os utilizadores bem como os seus dados associados (pessoais, saúde, e tempo real). De seguida, essas informações são guardadas no serviço e enviadas para o cliente em formato JSON. Caso não sejam encontradas ocorrências, será apenas devolvido um JSON vazio.

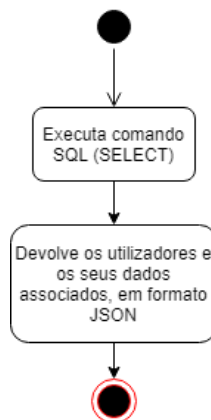


Figura 4.9: Diagrama de atividade do serviço de seleção de utilizadores e dados associados a eles.

4.2.5 Base de Dados

O uso de uma BD é essencial para a arquitetura e para todas as aplicações, uma vez que toda a informação recolhida e trocada pelas mesmas tem de ser armazenada.

Tabelas

É possível verificar na figura 4.10 que existe uma tabela principal, e quatro tabelas auxiliares que guardam diferentes informações.

- A tabela **tb_User** irá ter informação pessoal relativa aos utilizadores registados no HIA, mais especificamente o *email*, a *password* encriptada em *Secure Hash Algorithm-256* (SHA-256), o primeiro e último nome, o número de telemóvel, a data de nascimento, a morada e o género. Cada utilizador possuirá apenas uma linha nesta tabela. Caso seja pertinente mudar alguma informação, apenas a linha do utilizador em questão será alterada.
- A tabela **tb_HealthData** possui dados relacionados com a saúde de cada paciente. Cada utilizador possuirá apenas uma linha nesta tabela. Caso alguma desta informação tenha de ser alterada, a linha é apenas editada. Esta tabela possuirá informações como o peso, a altura, o tipo sanguíneo, alergias, e se possui doenças como diabetes, epilepsia ou asma. Existe também informação sobre alguma observação que o utilizador ache pertinente registar.
- A tabela **tb_RealTimeData** é responsável por guardar dados em tempo real de cada utilizador. São armazenados dados como as coordenadas em que se encontra, o BPM e a temperatura corporal. Nesta tabela existe apenas uma linha para cada utilizador, e, ao receber novas informações, o serviço apenas atualiza a linha já existente.
- A tabela **tb_SOSEvent** guardará dados relativos a um evento de emergência, tais como a data, o relatório do incidente e o nome do operador responsável pelo evento. Aqui é possível existir mais que um evento por utilizador.
- A tabela **tb_LeituraBandas** possui dados relativos à leitura de bandas corporais. Aqui é guardada informação como a data e hora em que foi efetuada a leitura, e uma matriz em formato BLOB que possui todos os dados recolhidos numa leitura, num determinado intervalo de tempo.

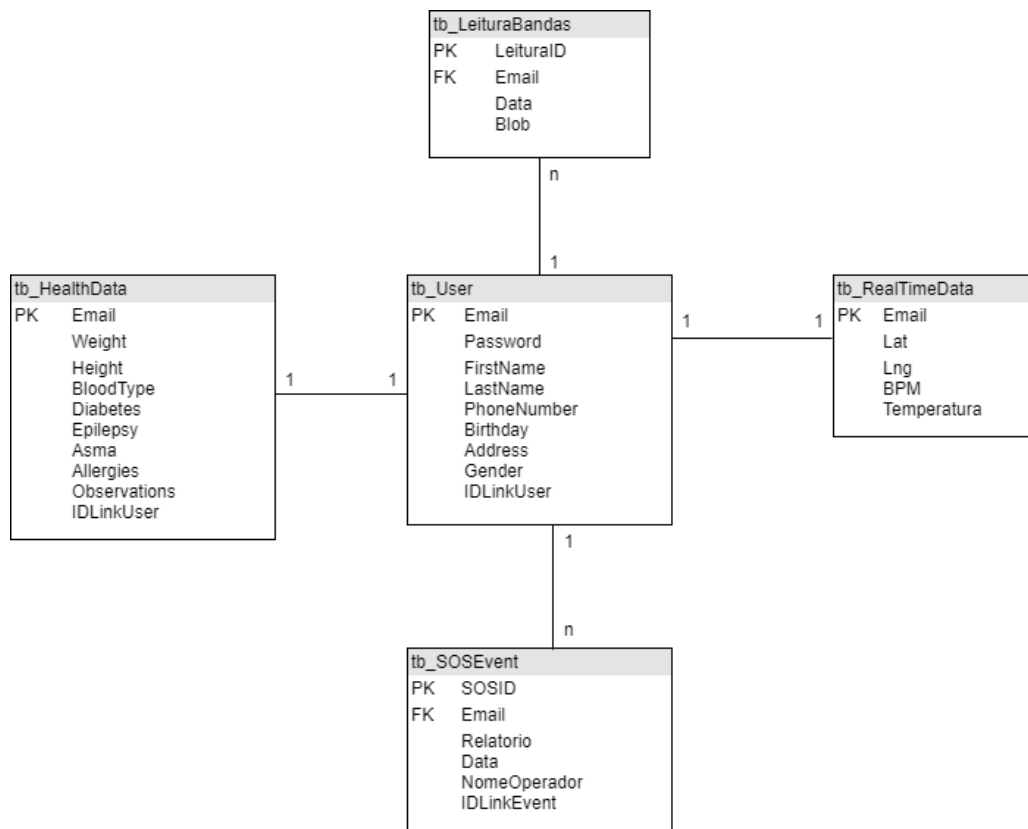


Figura 4.10: Esquema relacional da base de dados.

4.3 Engenharia de Software - Aplicação Móvel de Consulta

Será também elaborada uma aplicação móvel para consulta de dados. Esta aplicação, durante a fase de desenvolvimento do projeto, servirá essencialmente para testar alguns dos serviços apresentados anteriormente (4.2). O desenvolvimento desta aplicação abre também caminho para que prestadores de serviços de saúde como médicos de família, centros de saúde e hospitais, passam ser capazes, de uma forma eficaz e eficiente, consultar dados e eventos dos pacientes.

4.3.1 Requisitos Funcionais

- Apresentar o primeiro e último nome de todos os utilizadores registados no HIA.
- Permitir consultar dados pessoais, dados de saúde, e dados de tempo real de cada utilizador.
- Permitir consultar o histórico de eventos de emergência de cada utilizador.
- Permitir consultar o histórico de leituras das bandas corporais já realizadas.

4.3.2 Requisitos Não Funcionais

- Qualquer utilizador com conhecimento básico de como utilizar um dispositivo Android deve ser capaz de usar a aplicação.
- A aplicação pode ser usada em qualquer dispositivo Android com a versão 5.0 ou superior.
- O sistema deverá permitir que vários utilizadores acessem à aplicação ao mesmo tempo.
- A aplicação deve poder ser acessada a qualquer hora do dia, com exceção de falha na cobertura de rede.

4.3.3 Casos de Uso

- Consultar o nome de todos os utilizadores registados no sistema.
- Selecionar um utilizador, para aceder às suas informações.

- Aceder aos dados pessoais de um determinado utilizador previamente selecionado.
- Aceder aos dados de saúde de um determinado utilizador previamente selecionado.
- Aceder a dados capturados em "Tempo Real" de um determinado utilizador previamente selecionado.

Diagrama de Casos de Uso

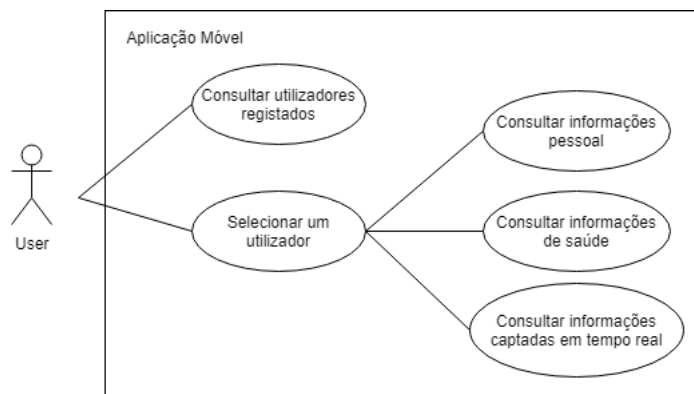


Figura 4.11: Diagrama de casos de uso da aplicação móvel.

4.3.4 Diagrama de Atividade

De seguida, é apresentado o diagrama de atividade da aplicação móvel (4.12). A aplicação começa por verificar, através de um serviço (descrito na figura 4.9 deste relatório), se existem utilizadores registados. Caso existam, serão apresentados no ecrã. De seguida, o utilizador da aplicação selecionará um utente, e a partir daí terá acesso às informações pessoais, dados de saúde e dados captados em tempo real associados ao mesmo.

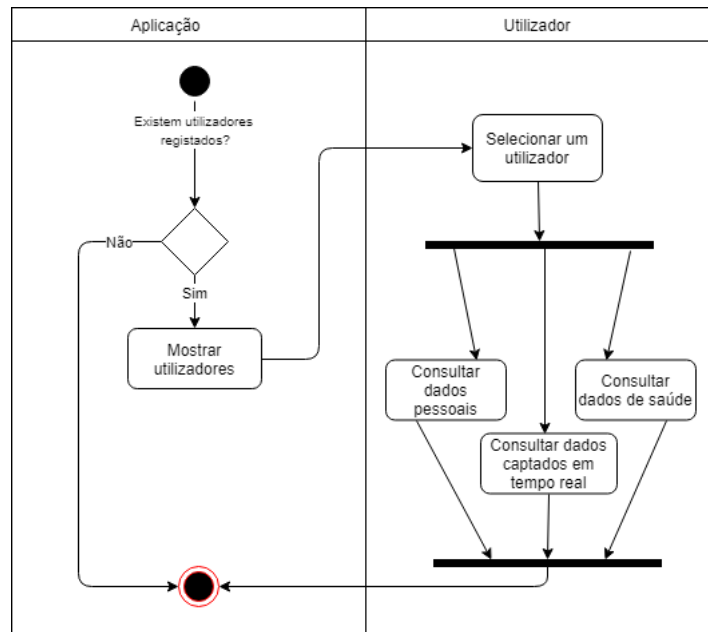


Figura 4.12: Diagrama de atividade da aplicação móvel.

4.4 Conclusões

Este capítulo descreveu os requisitos funcionais e não funcionais do sistema, bem como pormenorizou os serviços presentes no mesmo. Foi apresentada também a estrutura da BD envolvida no sistema.

Ainda neste capítulo foram descritos também os requisitos funcionais e não funcionais da aplicação móvel, bem como as atividades que ela disponibiliza.

Capítulo 5

Implementação

5.1 Introdução

Este capítulo explica detalhadamente a construção da arquitetura baseada em serviços e da aplicação móvel. Descreve também algumas decisões tomadas durante o desenvolvimento das mesmas.

5.2 Arquitetura do Sistema

Na figura seguinte (5.1) está esquematizada a arquitetura do projeto. Esta arquitetura é composta por diversos serviços, modelos de dados, e uma BD. A *layer* de registo e os restantes clientes ligados a este sistema, comunicam com a arquitetura baseada em serviços através de **HTTP requests** e **HTTP responses**.

Uma chamada *Hypertext Transfer Protocol* (HTTP) consiste no envio, por parte do cliente, de requisições (*requests*, às quais o servidor responde através de respostas (*responses*). Para que todo este sistema funcione corretamente, os serviços terão de se encontrar diretamente conectados à BD (excerto 5.1).

```
//connection string
private string conStr = "Server=tcp:hiadatabases1.database.
windows.net; Database=HIADatabase; User ID =
miguelbruno@hiadatabases1; Password=*****;
Trusted_Connection=False; Encrypt=True";
```

Excerto de Código 5.1: String de conexão à base de dados.

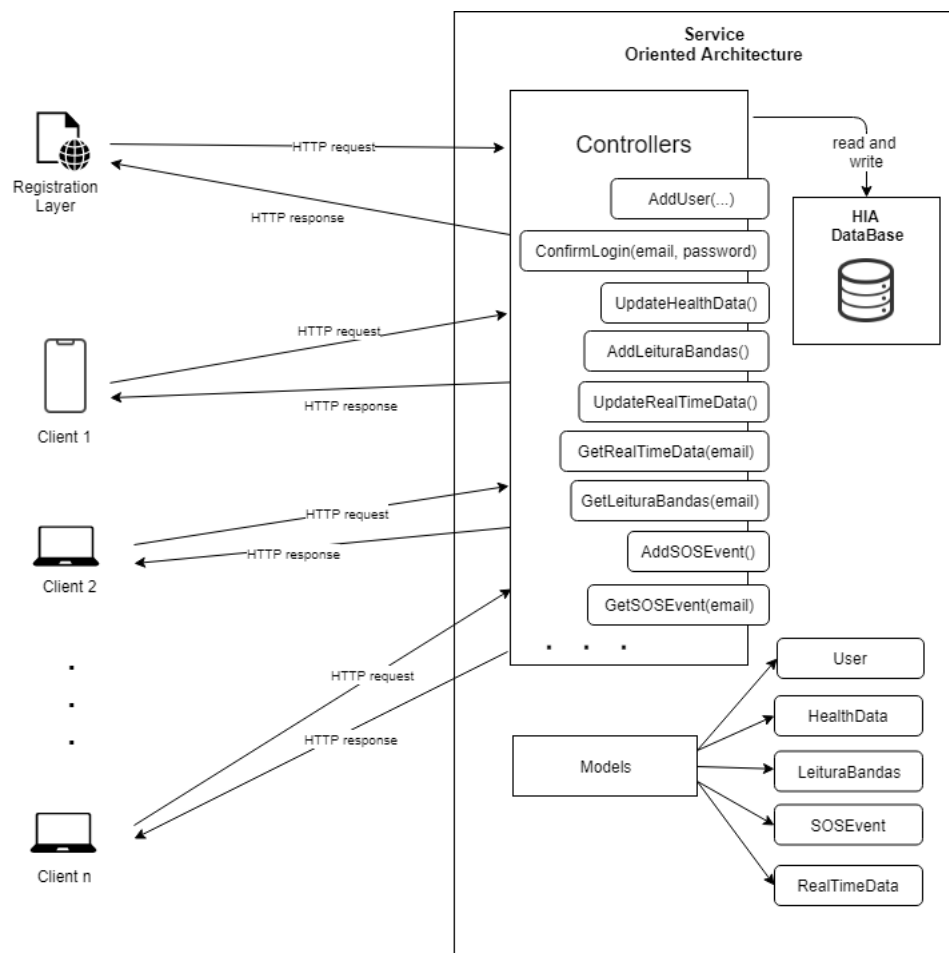


Figura 5.1: Esquema da arquitetura do sistema.

5.3 Estrutura da Arquitetura Baseada em Serviços

5.3.1 Modelos de Dados (*Models*)

Dentro desta arquitetura existem cinco modelos de dados diferentes, cada um deles com os seus dados correspondentes :

- **User:**

- Email;
- Password;
- FirstName;
- LastName;
- PhoneNumber;
- Birthday;
- Gender;
- Address.

- **HeathData:**

- Email;
- Weight;
- Height;
- BloodType;
- Diabetes;
- Epilepsy;
- Asma;
- Allergies;
- Observations.

- **SOSEvent:**

- SosID;
- Email;
- Relatorio;
- Data;
- NomeOperador.

- **LeituraBandas:**

- LeituraID;
- Email;
- Data;
- Blob.

- **RealTimeData:**

- Email;
- Latitude;
- Longitude;
- BPM;
- Temperatura.

Em seguida, é apresentado um exemplo de como é declarado um modelo de dados no projeto (excerto 5.2).

```
//modelo de dados SosEvent
public class SOSEvent{

    public string SoSID { get; set; }

    public string Email { get; set; }

    public string Relatorio { get; set; }

    public string Data { get; set; }

    public string NomeOperador { get; set; }

}
```

Excerto de Código 5.2: Declaração do modelo de dados "SOSEvent".

Neste exemplo, é declarado o modelo de dados de "SosEvent". Todos os restantes modelos são declarados de forma semelhante.

5.3.2 Serviços (*Controllers*)

Tratando-se este projeto de uma arquitetura baseada em serviços, são então os serviços a essência do mesmo. São usados para estabelecer um elo de ligação entre as aplicações incluídas no HIA e o servidor.

O HTTP (em português Protocolo de Transferencia de Hipertexto) é um protocolo de comunicação utilizado por sistemas de informação. É o protocolo usado nas comunicações efetuadas na *World Wide Web*. Este protocolo possui oito métodos que indicam a ação a ser realizada: *get*, *head*, *post*, *put*, *delete*, *trace*, *options*, e *connect* [20].

Neste projeto, optou-se por usar nos serviços, os métodos de ***HTTP Post*** e ***HTTP Get***.

Http Post

Tal como o nome indica, os métodos de ***HTTP Post*** permitem enviar dados para os serviços, onde serão processados. Neste projeto, os serviços que contêm este método são responsáveis por receber dados e enviá-los para o servidor onde ficarão alojados [13]. De seguida, é apresentado um excerto (5.3) de um dos serviços que contêm o método ***HTTP Post***:

```
[Route("/AddUser/")]
[HttpPost]
public int AddUser(string email, string password, string
    firstname, string lastname, string phonenumber,
    string birthday, string gender, string address)
{
    int status = 0;
    SqlConnection connection = new SqlConnection(this.
        conStr);
    SqlCommand cmd = new SqlCommand();
    try
    {
        if (connection.State == ConnectionState.Closed)
        {
            connection.Open();
        }
        string q = "Insert into tb_User(Email, Password,
            FirstName, LastName, PhoneNumber, Birthday,
            Gender, Address) values('" + email + "', '" +
            password + "', '" + firstname + "', '" +
            lastname + "', '" + phonenumber + "', '" +
            birthday + "', '" + gender + "', '" + address
            + "') insert into tb_HealthData (Email)
            values ('"+email+"') insert into
            tb_RealTimeData (Email) values ('" + email +
            "') ";
        cmd = new SqlCommand(q, connection);
        cmd.CommandType = CommandType.Text;

        cmd.ExecuteReader();
        status = 1;
    }
    catch (Exception ex)
    {
        throw ex;
    }
    finally
    {
        connection.Close();
        cmd.Dispose();
    }
    return status;
}
```

Excerto de Código 5.3: Serviço que adiciona um utilizador à BD.

O serviço presente no excerto 5.3 é responsável por adicionar um novo utilizador no servidor. O serviço recebe, através do método de *HTTP Post* informação como *email*, *password*, primeiro e ultimo nome, número de telemóvel, data de nascimento, género e morada. Inicialmente é efetuada a conexão ao servidor (excerto 5.1). De seguida é criado um comando SQL de inserção dos dados recebidos pelo serviço na BD. Neste comando, por uma questão de conveniência, são também criadas as linhas referentes aos dados de saúde e aos dados captados em tempo real, onde apenas é adicionado o email, ficando os restantes campos a *NULL*, para serem alterados posteriormente. Optou-se também por declarar como *int* os serviços de *post*, com o objetivo de ser devolvida uma *flag* que informa os clientes que usarão o serviço, se os dados foram ou não publicados com sucesso.

Os restantes serviços de *post* são elaborados de forma semelhante.

HTTP Get

O método de *HTTP Get* permite solicitar dados ao serviço a que está ligado. Neste projeto, os serviços que contêm este método terão de ser capazes de devolver a informação solicitada. De seguida é apresentado um excerto de um serviço *get* do projeto:

```
[Route("/{GetLeituras}/{Email}/{Data}")]
public List<LeituraBandas> GetLeituras(string Email,
    string Data)
{
    List<LeituraBandas> userList = new List<
        LeituraBandas>();
    SqlConnection connection = new SqlConnection(this.
        conStr);
    connection.Open();
    SqlCommand cmd = new SqlCommand("select * from
        tb_LeituraBandas where Email='" + Email + "' and
        Date='"+Data+"'", connection);
    cmd.CommandType = CommandType.Text;
    SqlDataReader sdr = cmd.ExecuteReader();

    while (sdr.Read())
    {
        LeituraBandas user = new LeituraBandas();
        user.LeituraID = sdr["LeituraID"].ToString();
        user.Email = sdr["Email"].ToString();
        user.Date = sdr["Date"].ToString();
        user.Blob = sdr["Blob"];

        userList.Add(user);
    }
}
```

```
        return UserList;
    }
```

Excerto de Código 5.4: Serviço que devolve informação acerca das leituras efetuadas por bandas corporais.

O serviço presente no excerto 5.4 é responsável por devolver ao cliente a informação solicitada. O serviço recebe inicialmente um *email* e uma data. Depois de estabelecer ligação com o servidor, é criado um comando SQL que irá seleccionar na *BD* as leituras efetuadas por bandas corporais correspondentes ao utilizador que possui o *email* fornecido, e na data também fornecida. Esta informação será devolvida ao utilizador em formato JSON.

Os restantes serviços de *get* são feitos de forma semelhante.

5.3.3 Base de Dados

A BD deste projeto contem cinco tabelas, cada uma com informações de saúde de cada utilizador. De seguida é apresentado um excerto que mostra como é criada a tabela de informação pessoal do utilizador:

```
if not exists (select * from dbo.sysobjects
where id = object_id(N'[dbo].[tb_User]'))
begin
CREATE TABLE tb_User (
Email varchar(150) NOT NULL,
Password varchar(MAX) NOT NULL,
FirstName varchar(MAX) NOT NULL,
LastName varchar(MAX) NOT NULL,
PhoneNumber varchar(MAX) NOT NULL,
Birthday date NOT NULL,
Address varchar(MAX) NOT NULL,
Gender varchar(MAX) NOT NULL,
CONSTRAINT PK_Email PRIMARY KEY (Email)
);
end
```

Excerto de Código 5.5: Criação da tabela *tb_User* na BD.

No comando SQL mostrado acima (excerto 5.5) é possível verificar que foi criada a tabela *tb_User* com todos os seus dados associados. Verifica-se também que foi declarado o *email* como chave primária.

As restantes tabelas do projeto foram criadas de forma semelhante à apresentada.

5.4 *Layer* de Registo

A *layer* de registo consiste numa interface de registo do utilizador. Este registo será único para todo o ecossistema HIA: o utilizador apenas terá de se registar uma vez, e com essa conta conseguirá efetuar o *login* em todas as aplicações do HIA. Esta interface é uma página HTML com instruções em *JavaScript*. Neste formulário de registo, o utilizador terá então de colocar informação básica sobre ele, bem como definir um *email* e uma *password* para mais tarde efetuar o *login*.

Em termos de código, todos os dados são inseridos em caixas de texto, com as exceções da data de nascimento, que é introduzida através de um *date picker*, e do género, que é selecionado através de um *drop down* (figura 5.2).

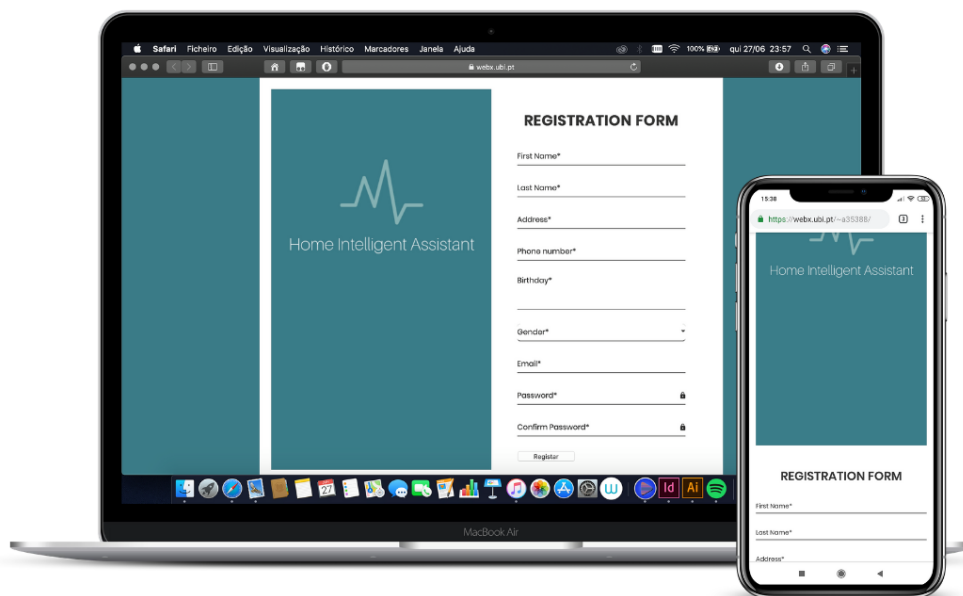


Figura 5.2: Interface de registo HIA.

No excerto seguinte, que consta do código *JavaScript* da *layer* de registo, é possível conferir um **HTTP request**, que através do método **post**, enviará os dados pessoais do utilizador para o serviço que os adicionará posteriormente ao servidor:


```
const Http = new XMLHttpRequest ();

const url = "https://hiaweb-service1.azurewebsites.net/AddUser?
    email="+Email+"&password="+hexString(digestValue)+"&firstName
    =" +Fname+"&lastName="+Lname+"&phoneNumber="+PhoneNumber+"&
    birthday="+Bday+"&gender="+Gender+"&address="+Address;

Http.open("POST", url);
Http.send();
```

Excerto de Código 5.6: Chama do serviço que adiciona um utilizador na BD na *layer* de registo.

Nesta *layer* e em todas as aplicações do universo HIA, as *passwords* são enviadas para os serviços, e posteriormente guardadas no servidor, encriptadas em **SHA-256**.

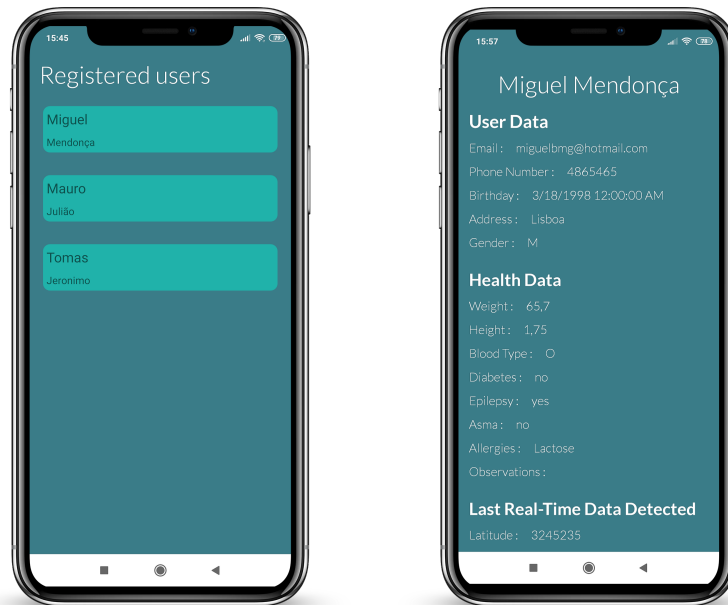
5.5 Aplicação Móvel

A aplicação móvel incluída neste projeto tem como objetivo a consulta de dados associados aos utilizadores registados no HIA.

Inicialmente, a aplicação cria uma *hash map* onde guardará toda a informação associada em cada utilizador. Essa informação resulta de um método *get* que fará uma seleção dos dados necessários.

Visualmente, no ecrã inicial, a aplicação apresenta uma *list view* onde apresenta o primeiro e ultimo nome de cada utilizador registado (figura 5.3a). De seguida, ao clicar em qualquer utilizador, é mostrada no ecrã informação relativa mesmo (figura 5.3b). Este evento implica uma mudança de classe, onde foi usado um *intent* para transferir as informações relativas a cada utilizador de uma classe para outra (excerto 5.7).

Toda a informação dos utilizadores é guardada na *hash map* no momento em que a aplicação é aberta. Isso faz com que, inicialmente a aplicação seja ligeiramente mais lenta na altura de apresentar os nomes dos utilizadores, mas de seguida, na altura de consultar os dados, a aplicação torna-se mais rápida e fluida, uma vez que já tem todos os dados guardados.



(a) Ecrã inicial.

(b) Ecrã de informações.

Figura 5.3: Interface da aplicação.

```
list.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
    @Override  
    public void onItemClick(AdapterView<?> parent, View  
        view, int i, long l) {  
        Intent intent = new Intent(MainActivity.this,  
            Info.class);  
  
        intent.putExtra("objetodata", ListaNomes.get(i)  
            );  
  
        startActivity(intent);  
    }  
});
```

Excerto de Código 5.7: *Intent* de transferência de dados.

5.6 Conclusões

Neste capítulo foram detalhados os vários componentes da implementação da arquitetura baseada em serviços e da aplicação móvel, de modo a perceber-se melhor o funcionamento da mesma.

Capítulo 6

Demonstração e Validação da Arquitetura Baseada em Serviços

6.1 Introdução

Neste capítulo será apresentada a validação de toda a arquitetura construída. Para o efeito serão apresentados vários exemplos práticos do funcionamento da arquitetura desenvolvida, e todos os seus serviços. Recorreu-se à criação de *personas* que representam diferentes pessoas e diferentes situações/ocorrências em que têm de recorrer ao uso do HIA.

6.2 *Personas*

6.2.1 *Persona 1 - Rui Marques*

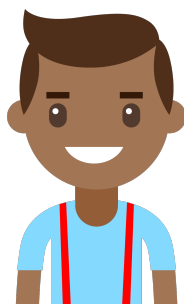
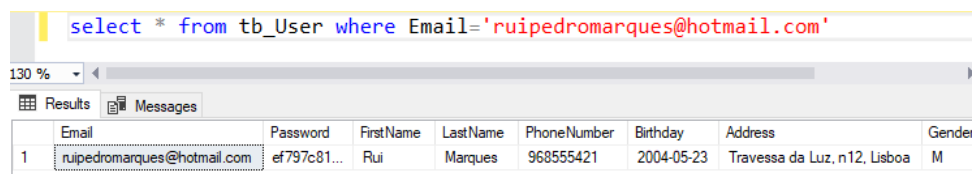
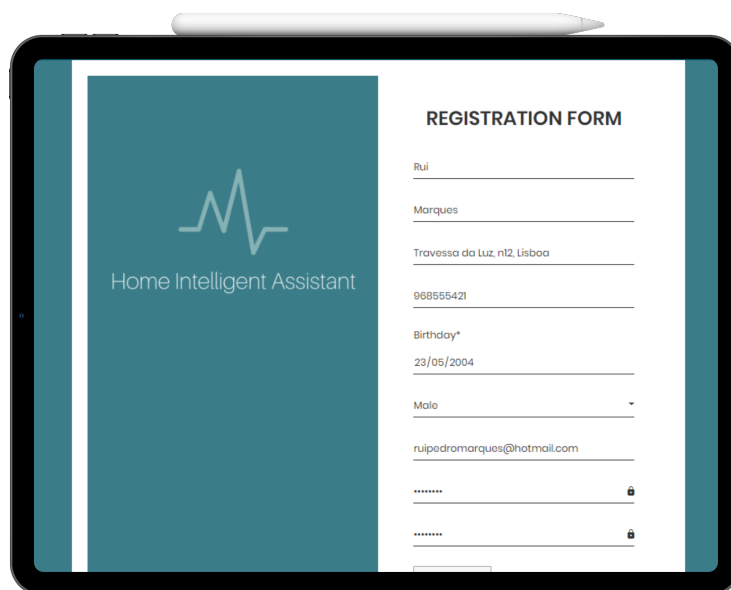


Figura 6.1: *Persona* Rui Marques.

O **Rui Marques** é um jovem de 15 anos entusiasta das novas tecnologias. A sua mãe ofereceu-lhe um *smartwatch* com capacidade para registar alguns sinais vitais e também a sua localização em tempo real. Através do registo no HIA (figura 6.2 e 6.3a) informações como: coordenadas de localização, o BPM e a temperatura corporal do Rui são transmitidas e armazenadas. A mãe do Rui, consegue assim através da aplicação de consulta instalada no seu *smartphone*, consultar estas informações referentes ao Rui (6.3b).



	Email	Password	FirstName	LastName	PhoneNumber	Birthday	Address	Gender
1	ruipedromarques@hotmail.com	ef797c81...	Rui	Marques	968555421	2004-05-23	Travessa da Luz, n12, Lisboa	M

Figura 6.2: Registo da *persona* Rui na BD.

REGISTRATION FORM

Rui

Marques

Travessa da Luz, n12, Lisboa

968555421

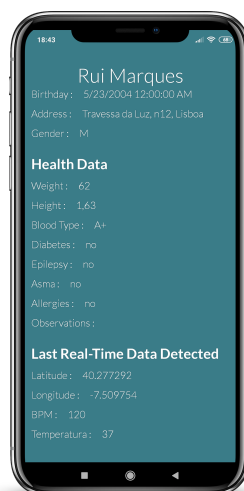
Birthday*

23/05/2004

Male

ruipedromarques@hotmail.com

(a) Registo.



Rui Marques

Birthday: 5/23/2004 12:00:00 AM

Address: Travessa da Luz, n12, Lisboa

Gender: M

Health Data

Weight: 62

Height: 1.63

Blood Type: A+

Diabetes: no

Epilepsy: no

Asma: no

Allergies: no

Observations:

Last Real-Time Data Detected

Latitude: 40.277292

Longitude: -7.509754

BPM: 120

Temperatura: 37

(b) Informações na aplicação.

Figura 6.3: Registo e informações *persona* Rui.

6.2.2 Persona 2 - Ana Ferreira

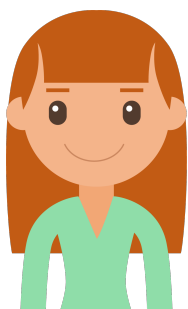


Figura 6.4: *Persona* Ana Ferreira.

A **Ana Ferreira** é uma adolescente de 19 anos. Nos últimos tempos, a Ana sentia algum desconforto e sucessivas faltas de ar quando se encontrava em casa, junto dos seus animais de estimação. Num desses recorrentes episódios, decidiu efetuar uma chamada de emergência através de uma aplicação móvel integrada no HIA. Depois de falar com o operador do outro lado da chamada, chegaram à conclusão possivelmente a Ana seria alérgica ao pêlo de gato (registo da chamada na figura 6.5). Durante a chamada, o operador obteve em tempo real todos os dados de saúde que a Ana tinha no seu histórico, bem como a sua localização em tempo real.

Depois de se dirigir a uma clínica, a Ana confirmou a sua alergia ao pêlo de gato, e o seu processo foi então alterado (registo de saúde atualizado na figura 6.6).

A Ana está agora a pensar em comprar um *smartwatch* porque leu na *Internet* que está a ser desenvolvida, para esse dispositivo, uma versão da aplicação que ela usou para efetuar a chamada de emergência.

```
select * from tb_SOSEvent where Email='ferreiraana@gmail.com'
```

	SOSID	Email	Relatorio	Data	IDLinkEvent	NomeOperador
1	3	ferreiraana@gmail.com	Sintomas de falta de ar. Possivel alergia a pêlo de gato	2019-07-02	NULL	Júlio Dias

Figura 6.5: Registo do evento de emergência da *persona* Ana na BD.

```
select * from tb_HealthData where Email='ferreiraana@gmail.com'
```

	Email	Weight	Height	BloodType	Diabetes	Epilepsy	Asma	Allergies	Observations
1	ferreiraana@gmail.com	59,9	1,71	A+	no	no	no	pêlo de gato	NULL

Figura 6.6: Registo dos dados de saúde da *persona* Ana na BD.

6.2.3 Persona 3 - Jorge Almeida



Figura 6.7: *Persona* Jorge Almeida.

O Sr. **Jorge Almeida** tem 71 anos. Há poucos anos, o Sr. Jorge sofreu um enfarte do miocárdio, e a sua saúde acabou por ficar ligeiramente debilitada. Entretanto, foi aconselhado pela sua médica de família a efetuar regularmente um Eletrocardiograma (ECG) para a sua situação ser vigiada.

Graças ao HIA, é possível que a médica de família do Sr. Jorge consulte e acompanhe, através de um *software* específico para o efeito, o histórico dos resultados dos ECGs efetuados pelo seu paciente (figura 6.8).

Sempre que for detetada alguma situação anormal nos seus ECGs, o Sr. Jorge receberá um aviso no seu *smartphone* a comunicar que se deve dirigir ao prestador de saúde mais próximo.

O Sr. Jorge acabou também por adquirir na *Play Store* a aplicação do universo HIA que realiza chamadas de emergência. Até hoje, felizmente, ainda não precisou de a utilizar (figura 6.9).

```
select * from tb_LeituraBandas where Email='jorgealmeida@iol.pt'
```

	IDM	Email	Date	Blob
1	5	jorgealmeida@iol.pt	2019-06-30 at 00:31:10 ...	0.0 0.0 1.8193 -0.1301 1.0 0.0 1.8201 -0.1301 ...
2	6	jorgealmeida@iol.pt	2019-07-01 at 11:06:43 ...	0.0 0.0 1.8193 -0.1301 1.0 0.0 1.8201 -0.1301 ...

Figura 6.8: Registo das leituras de bandas corporais da *persona* Jorge na BD.

```
select * from tb_SOSEvent where Email='jorgealmeida@iol.pt'
```

SOSID	Email	Relatorio	Data	IDLinkEvent	NomeOperador
-------	-------	-----------	------	-------------	--------------

Figura 6.9: Registo de eventos de emergência da *persona* Jorge na BD.

6.2.4 Outras *personas*

Em seguida são apresentadas algumas *personas* já mencionadas, que são essenciais para que o sistema funcione corretamente. São elas :

- A mãe do Rui Marques (figura 6.10a);
- O operador que falou com a Ana Ferreira na chamada de emergência (figura 6.10b);
- A médica de família do Sr. Jorge Almeida (figura 6.10c).

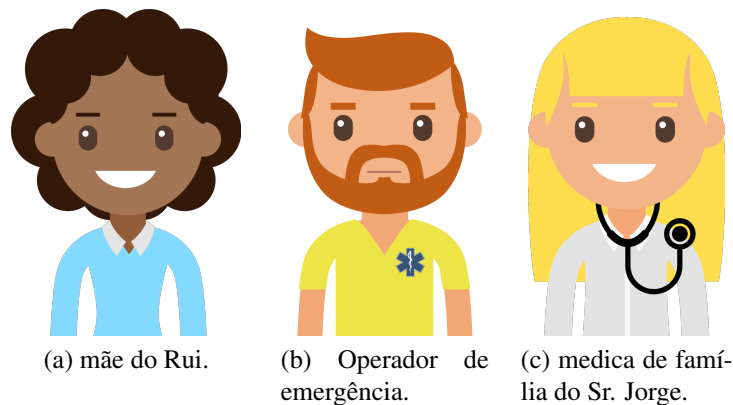


Figura 6.10: Outras *personas* presentes neste sistema.

6.3 Demonstração e Validação

Na figura seguinte (6.11) estão representados os serviços presentes na arquitetura, associando-os às *personas* que terão de os utilizar, para que todo o sistema funcione corretamente. Como podemos verificar:

- O Rui, a Ana e o Sr. Jorge fizeram o seu registo no sistema;
- Todos os utilizadores registados fazem *updates* de dados captados em tempo real, uma vez que todos têm um *smartphone* capaz de captar a sua localização. No caso do Rui, ainda tem um *smartwatch* que complementa essa captura de dados.
- A Ana conseguiu atualizar a sua ficha de saúde, adicionando a sua alergia. A aplicação que a Ana usou para efetuar a chamada de emergência encarregou-se de registar o episódio sucedido na BD;

- A aplicação do Sr. Jorge adiciona regularmente os resultados do ECG na sua ficha de saúde. Está também preparado para efetuar uma chamada de emergência através da aplicação, sempre que precisar.
- Durante a chamada de emergência da Ana, o operador teve acesso aos dados pessoais, dados de saúde e dados em tempo real da Ana. No final da chamada, o operador adicionou um relatório da ocorrência na ficha de saúde da Ana.
- A médica de família do Sr. Jorge consegue ter acesso aos dados pessoais, dados de saúde e dados em tempo real do seu paciente. Tem também acesso aos ECGs que o Sr. Jorge efetua;
- A mãe do Rui tem acesso aos dados pessoais, dados de saúde e dados em tempo real do seu filho.

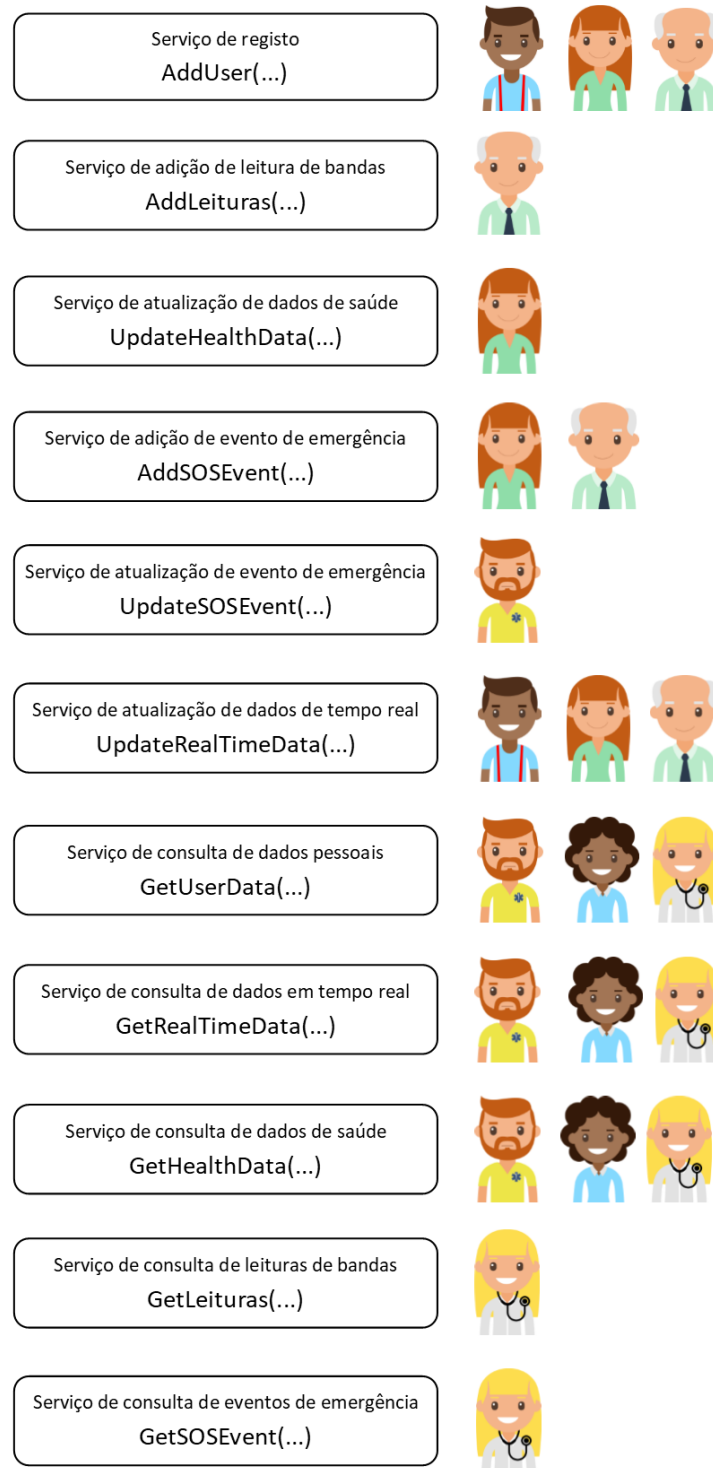


Figura 6.11: Associação dos serviços Web da arquitetura do sistema às diferentes *personas*.

6.4 conclusões

Neste capítulo foi demonstrada a validação de todo o sistema, recorrendo a *personas* representativas de diversas situações. Ficaram apresentadas várias chamadas à BD com o objetivo de demonstrar como as informações são guardadas e consultadas.

Foram também neste capítulo foram apresentados vários exemplos de uso dos serviços presentes na arquitetura do projeto.

Capítulo 7

Conclusões e Trabalho Futuro

7.1 Conclusões Principais

Com este projeto foi criada uma arquitetura baseada em serviços para ser incluída num ambiente *Smart Home*, bem como uma aplicação móvel destinada à consulta de dados dos utilizadores registados no sistema. Começou por ser feita uma análise de requisitos com o objetivo de delinear as funcionalidades que se desejava implementar. Após a construção dos vários diagramas que constituem a engenharia de *software*, foram selecionadas as tecnologias a utilizar durante o desenvolvimento do projeto. Seguidamente, foram implementadas a arquitetura baseada em serviços e a aplicação móvel. Por ultimo, foram criadas *personas* no sistema, com o objetivo de demonstrar e validar o mesmo.

Podemos assim concluir que este projeto concretizou todos os objetivos a que se propôs, e que, quando adequado, poderá ser facilmente incluído num ecossistema de *Smart Home*, propondo aos utilizadores uma experiência de controlo e segurança na sua vida.

7.2 Trabalho Futuro

Apesar do projeto se encontrar finalizado, o objetivo será a evolução do mesmo, incluindo na arquitetura um leque de outras aplicações que se enquadrem no conceito de *Smart Home* e *Smart Health*. Mesmo incluindo novas aplicações, novos sensores e novas fontes de dados, é importante que todo o ecossistema se torne cada vez mais interoperável. Por isso, um estudo e implementação de técnicas de interoperabilidade é um dos principais objetivos futuros.

Será também um objetivo futuro, a concretização de um testes piloto, com a premissa testar todo o sistema em ambiente real.

Bibliografia

- [1] 6,5 milhões de portugueses têm smartphone. [Online] <https://tek.sapo.pt/noticias/telecomunicacoes/artigos/65-milhoes-de-portugueses-tem-smartphone>.
- [2] A guide to home automation and smart home assistants. [Online] <https://www.clickatell.com/articles/technology/home-automation-smart-home-assistants/>.
- [3] Assistentes pessoais virtuais: por que eles se tornaram uma tendência? [Online] <https://blog.sonda.com/assistentes-pessoais-virtuais-por-que-eles-se-tornaram-uma-tendencia/>.
- [4] CareAngel. [Online] <https://www.careangel.com/>.
- [5] Meet Android Studio. [Online] <https://developer.android.com/studio/intro>.
- [6] SmartThings Samsung. [Online] <https://www.samsung.com/pt/apps/smartthings/>.
- [7] Assistentes pessoais virtuais: a inteligência artificial aplicada no dia a dia, 2016. [Online] <https://helabs.com/blog/assistentes-pessoais-virtuais-a-inteligencia-artificial-aplicada-no-dia-a-dia/>.
- [8] SQL Server Management Studio (SSMS), 2018. [Online] [://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-2017](https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-2017).
- [9] Visual Studio Documentation, 2019. [Online] <https://docs.microsoft.com/en-us/visualstudio/?view=vs-2019#pivot=get-started>.
- [10] What is Azure?, 2019. [Online] <https://azure.microsoft.com/en-gb/overview/what-is-azure/>.

- [11] Paulo Alves. 13 coisas que o Google Assistant pode fazer no celular Android, 2017. [Online] <https://www.tudocelular.com/android/noticias/n89161/13-coisas-Google-Assistant-pode-fazer.html>.
- [12] S. Fenves M. Gruninger-V. Kashyap B. Lide J. Nell R. Raman R. Sriram C. Bock, L. Carnahan. Healthcare Strategic Focus Area: Clinical Informatics, 2005.
- [13] Oficina da Net. O protocolo HTTP, 2007. [Online] https://www.oficinadanet.com.br/artigo/459/o_protocolo_http.
- [14] Trans Atlantic Consumer Dialogue. Resolution on Software Interoperability and Open Standards, 2008.
- [15] J. Walker J. Adler-Milstein E. Pan, D. Johnston. The Value of Healthcare Information Exchange and Interoperability, 2004.
- [16] Kevin Heubusch. Interoperability: What it Means, Why it Matters, 2006. [Online] <https://library.ahima.org/doc?oid=60942#.XQ5be-hKjIU>.
- [17] Patient Home. SMART HEALTH TECHNOLOGY THE NEXT STEP IN HEALTHCARE TECHNOLOGY, 2018. [Online] <https://path2025.dk/smart-health-technology/>.
- [18] J. Olaleke Iroju Olaronke, Ishaya Gambo. Interoperability in Healthcare: Benefits, Challenges and Resolutions, 2013.
- [19] Pedro Pinto. Afinal o que é um sistema distribuído?, 2015. [Online] <https://pplware.sapo.pt/informacao/afinal-o-que-e-um-sistema-distribuido/>.
- [20] W3schools. HTTP Request Methods. [Online] https://www.w3schools.com/tags/ref_httpmethods.asp.