INSTITUTO POLITÉCNICO
DE VIANA DO CASTELO

João Carlos da Rocha Palma

BlitzChat - Sistema de Comunicação Instantânea Intraempresa

Mestrado em Engenharia de Software

Trabalho de Projeto efetuado sob a orientação de
Doutora Sara Paiva

Fevereiro de 2019

RESUMO

A sociedade de hoje está profundamente marcada pela era digital. O avanço tecnológico substancial que se tem verificado nas últimas décadas tem vindo a transformar a vida dos seres humanos e, sobretudo, revolucionou a forma como estes comunicam entre si e com as instituições.

Neste contexto comunicacional, os telemóveis são hoje utilizados para mais do que realizar ou receber chamadas telefónicas, agregando um conjunto múltiplo de funcionalidades que lhes conferem o título de "telefones inteligentes" ou *smartphones*. É através destes *smartphones* que as pessoas estão, cada vez mais, permanentemente conectadas com o mundo.

Aliadas à evolução dos *smartphones*, as aplicações móveis têm vindo também a assumir uma preponderância enorme no mercado e tem-se verificado um incremento substancial na sua utilização. Reconhecendo a importância que os *smartphones* têm vindo a assumir na vida e na forma de comunicação interpessoal, as organizações têm vindo a privilegiar o recurso a aplicações móveis para comunicar com os seus clientes e com os seus colaboradores.

Nesta sequência, o presente trabalho tem como objetivo apresentar o desenvolvimento de uma aplicação móvel que permite o envio de mensagens envolvendo um aspeto inovador adicional face ao que atualmente existe: a possibilidade de encontrar alguém dentro de uma empresa/instituição, sem a necessidade de ter os seus contactos nem de a adicionar nas redes sociais/privadas.

A aplicação móvel tem a designação de *Letter Messenger* (Inicialmente designada de BlitzChat e nome pelo qual também é referida ao longo deste trabalho) e o seu objetivo fundamental é a comunicação interpessoal e intraempresarial, alicerçada na facilidade de utilização.

Fevereiro de 2019

ABSTRACT

Today's society is deeply marked by the digital age. The substantial technological advance that has occurred in the last decades has been transforming the lives of human beings and, above all, has revolutionized the way they communicate with each other and with institutions.

In this communication context, mobile phones are now used for more than making or receiving phone calls, offering a multiple set of features that give them the title of smartphones. It is through these smartphones that people are, increasingly, permanently connected with the world.

Allied to the evolution of smartphones, mobile applications have also assumed a huge preponderance in the market and there has been a substantial increase in their use. Recognizing the importance that smartphones have come to assume in life and in the form of interpersonal communication, organizations have been privileging the use of mobile applications to communicate with their customers and with their collaborators.

Given this starting point, this paper aims to expose the development of a mobile application that allows the sending of messages involving an additional innovative aspect: the possibility to find someone inside a large company / institution, without the need to add it in social / private networks.

To do this, the members of the institution must register in the application and have access to all the contacts already registered in the app, and can exchange instant messages that streamline intra-institutional communication, network.

The mobile application has the designation of Letter Messenger (Initially designated as BlitzChat and name by which it is also referred to throughout this work) and its fundamental objective is interpersonal and intra-corporate communication, based on ease of use.

February, 2019

Conteúdo

Índice de figuras

Índice de Tabelas

1. INTRODUÇÃO

A sociedade de hoje está profundamente marcada pela era digital. O avanço tecnológico substancial que se tem verificado nas últimas décadas tem vindo a transformar a vida dos seres humanos e, sobretudo, revolucionou a forma como estes comunicam entre si e com as instituições.

Neste contexto comunicacional, os telemóveis são hoje utilizados para mais do que realizar ou receber chamadas telefónicas, agregando um conjunto múltiplo de funcionalidades que lhes conferem o título de "telefones inteligentes" ou *smartphones*. É através destes *smartphones* que as pessoas estão, cada vez mais, permanentemente conectadas com o mundo.

Aliadas à evolução dos *smartphones*, as aplicações móveis têm vindo também a assumir uma preponderância enorme no mercado e tem-se verificado um incremento substancial na sua utilização. Reconhecendo a importância que os *smartphones* têm vindo a assumir na vida e na forma de comunicação interpessoal, as organizações têm vindo a privilegiar o recurso a aplicações móveis para comunicar com os seus clientes e com os seus colaboradores.

O resto deste capítulo apresenta uma contextualização da problemática, o plano de trabalho e a estrutura do presente projeto.

1.1. Contextualização

Em 2007, através do lançamento da primeira geração de *smartphones*, registou-se a primeira grande revolução no mercado dos telemóveis e da computação móvel, pois estes dispositivos móveis agregavam mais funcionalidades e eram dotados de uma maior capacidade de processamento que os telemóveis que os antecederam (Davis et al., 2011).

A introdução desses dispositivos e das aplicações móveis no mercado teve repercussões inquestionáveis na vida quotidiana, sobretudo pela criação de novas dinâmicas de comunicação que permitiram facilitar e intensificar as relações interpessoais. Isto é ainda mais expressivo quando se constata que o número de utilizadores de *smartphones*, a nível mundial, ultrapassou os 2 milhares de milhões no final de 2016, prevendo-se que no final ano de 2018 mais de metade da população mundial tivesse acesso a um *smartphone*. Isto é corroborado pelos dados obtidos pela

GSMA, os quais referem que, atualmente, mais de 5 mil milhões de pessoas utilizam algum tipo de dispositivo móvel no mundo, o que corresponde a cerca de 67% da população mundial (GSMA, 2018).

A par do aumento de utilizadores de *smartphones*, constata-se que a utilização de aplicações móveis tem vindo a registar um crescimento exponencial, especialmente, pela multiplicidade da sua utilização (GSMA, 2018). Efetivamente, existem inúmeras ferramentas com várias funcionalidades e que se encontram à distância de um clique, satisfazendo as mais diversas necessidades dos utilizadores. Não obstante, a questão da obsolescência é mínima pois as aplicações são atualizadas constantemente.

Comparativamente com o tempo despendido com os *browsers* dos computadores, os utilizadores de *smartphones* têm dedicado mais tempo na utilização das aplicações móveis. Deste modo, antevê-se que, em breve, as aplicações móveis poderão vir a substituir por completo os *websites*. Algumas das motivações para este predomínio das aplicações móveis são: o fácil acesso, a possibilidade de uso *offline*, a sua rapidez, a facilidade de obtenção de dados e as funcionalidades mais complexas (Bellman et. al, 2011).

O uso das aplicações desenvolvidas pelas próprias marcas de telemóveis (*iOS* e *Android*) tem um impacto persuasivo positivo nos consumidores e garantem altos níveis de envolvimento, interesse e interação, o que, consequentemente, aumenta o interesse por essas marcas e respetivas categorias de produtos (Bellman et. al, 2011). Segundo Bellman et al (2011) isto conforma uma estratégia de *marketing* inteligente, pois acabam por ser os consumidores a identificarem-se com as marcas, sem grande intervenção destas últimas. De facto, a decisão de efetuar o *download* de uma aplicação móvel de uma determinada marca cabe somente aos consumidores, tendo sempre subjacente a relevância que estes atribuem à marca escolhida. Sobre isto, Bellman et. al (2011) referem também que as aplicações móveis são consideradas mais eficazes do que as versões *Web*, precisamente, por reforçarem a eficiência da transmissão de mensagens relativas a uma marca. A aquisição das aplicações móveis é efetuada através da *AppStore,* onde é possível encontrar as aplicações organizadas por categorias. Até setembro de 2018, as aplicações de negócio confirmaram a segunda categoria mais popular, com uma participação de 9,77% de todas as aplicações ativas (Figura 1). Estas aplicações de negócio inserem-se nas categorias de comunicação,

englobando uma tríade constituída por *Business, Productivity e Social Networking* **(**Statista, 2018).

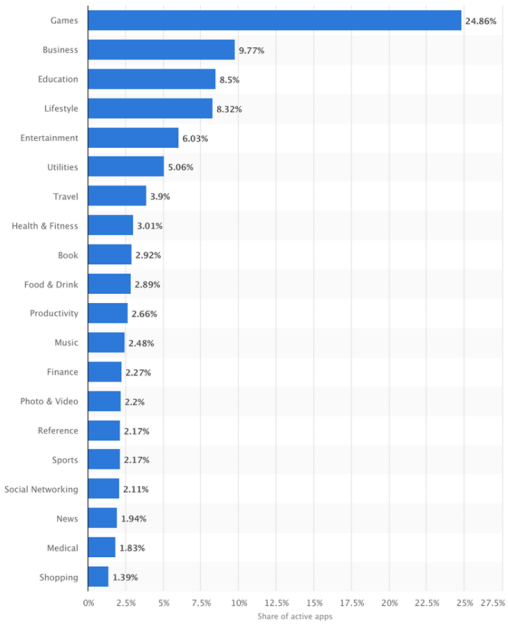

Figura 1-Percentagens das categorias mais descarregadas na AppStore

Atendendo aos dados apresentados, no próximo ponto far-se-á uma alusão à comunicação e ao impacto das aplicações móveis na criação de novos ecossistemas comunicacionais, particularmente, nas organizações.

1.2. As aplicações móveis e a comunicação intrainstitucional

As aplicações móveis têm vindo a integrar-se dentro das instituições ao longo dos últimos anos, tendendo a vir a desempenhar um papel cada vez mais importante nas mesmas (Silva & Reis, 2009). Segundo estes autores, a busca por novos métodos mais claros e eficientes para comunicar interna e externamente tem sido uma das ambições das empresas. Em função desta crescente preocupação, as empresas começaram a depender bastante dos seus sistemas de informação para tarefas de colecionar dados, tirar sentido dos mesmos e disponibilizá-los de forma acessível e consumível para os seus colaboradores. De facto, os autores admitem que estes processos são uma das bases para uma comunicação intrainstitucional eficiente (Silva & Reis, 2009).

As aplicações móveis têm vindo a ser úteis neste domínio uma vez que têm permitido hoje em dia substituir ferramentas de organização e gestão empresarial

como calendários, listas de contactos e até mesmo mapas: todos eles guardados convenientemente num *smartphone* (Ottka, 2015).

Esta mudança de paradigma de adoção de novas ferramentas digitais está também ligada a mudanças culturais: hoje em dia, grande parte da população (e em especial a geração *millenial*) é proficiente no uso de ferramentas digitais (Holicza & Kadëna, 2018). Mesmo as redes sociais têm vindo a influenciar o modo como a comunicação organizacional tem vindo a ser realizada devido à proliferação de redes como o Facebook, Twiiter e outras (Holicza & Kadëna, 2018). As empresas começaram também a utilizar esses canais para comunicar com o contexto empresarial. As aplicações de redes sociais também possuem espaços para troca de mensagens, no entanto, estas dependem do pré-registo nas mesmas e do e-mail da pessoa (Nascimento & Silveira, 2017). Desse modo, é pertinente que a comunicação, a nível organizacional, se faça de uma forma mais simples e eficaz. De facto, hoje em dia, fruto da evolução tecnológica, nas interações informais, as pessoas trocam mensagens instantâneas, através de telemóveis, constituindo esta uma forma de comunicação universal e já assumida como standard em todo o mundo.

1.3. Objetivos

Partindo das ideias de "universalidade de utilização" e "facilidade de acesso", a presente dissertação foca-se no desenvolvimento de uma aplicação móvel (*app*) que facilita encontrar alguém dentro de uma empresa/instituição, sem a necessidade de ter os seus contactos nem de a adicionar nas redes sociais/privadas, e comunicar com ela via mensagem/chat. A grande diferenciação face ao que atualmente existe é que a aplicação Blitzchat permite encontrar colaboradores de uma empresa sem necessitar de saber o seu e-mail ou número de telefone. Foca-se no uso intraempresarial, permitindo criar divisões privadas só para administradores ou divisões para cada departamento dentro da empresa em função das necessidades.

Para tal, os elementos da instituição têm de proceder ao registo na aplicação e passam a ter acesso a todos os contatos já registados na app, podendo trocar mensagens instantâneas que agilizam a comunicação intrainstitucional, em rede.

Considera-se que esta é uma ferramenta eficaz para a comunicação a nível empresarial uma vez que cada cargo é catalogado consoante a sua função na empresa, permitindo que todos possam comunicar entre si, de uma forma célere. Para maior

rigor, a pesquisa pode ser efetuada através de variadas categorias, nomeadamente: nome, função, número de telefone, ou e-mail.

A aplicação móvel tem a designação de *Letter Messenger* (Inicialmente designada de BlitzChat) e o seu objetivo fundamental é a comunicação interpessoal e intraempresarial, alicerçada na facilidade de utilização. É ainda objetivo manter os contactos pessoais privados permitindo, no entanto, a troca mútua de mensagens instantâneas com todos os contactos dos colaboradores da instituição que já se encontram registados na aplicação.

Considera-se que esta aplicação de comunicação intrainstitucional tem uma elevada pertinência nas organizações de hoje e pode constituir um importante ecossistema comunicacional.

Com as devidas adaptações, pode ser difundido a instituições de todos os setores, desde empresarial, educacional, social e até de saúde. Trata-se de uma aplicação inovadora, intuitiva e capaz de suplantar as demais formas tradicionais de comunicação institucional.

1.4. Plano de trabalho

Nesta secção apresenta-se o plano de desenvolvimento do projeto. Tal como demonstra a Tabela 1, o projeto, composto por 11 tarefas, teve início no dia 1 de março de 2018 e terminou a 30 de novembro de 2018.

Durante a execução do projeto foi feita uma reunião mensal de monitorização e análise do ponto de situação do projeto. Foram previstas e executadas as tarefas abaixo descritas.

Número	Nome da Tarefa	Início	Fim	Duração
1	Relatório da Tese	01/03/2018	01/02/2019	11 Meses
2	Mockups da Aplicação	06/03/2018	20/03/2018	15 Dias
3	Implementação da base de dados	21/03/2018	30/03/2018	10 Dias
4	Criação dos Serviços Web/RESTful API	01/04/2018	20/04/2018	20 Dias
5	Desenvolvimento da App Part 1 - Registo e Login do utilizador	21/04/2018	20/05/2018	30 Dias
6	Desenvolvimento da App Part 2 -	20/05/2018	25/05/2018	6 Dias

	Detalhes do utilizador e definições			
7	Desenvolvimento da App Part 3 - Implementação dos Contactos	26/05/2018	08/06/2018	13 Dias
8	Desenvolvimento da App Part 4 - Serviço de Chamadas	10/06/2018	20/07/2018	30 Dias
9	Desenvolvimento da App Part 5 - Serviço de Mensagens	21/07/2018	20/08/2018	30 Dias
10	Implementação de notificações	21/08/2018	24/08/2018	4 Dias
11	Teste e correção de bugs	25/08/2018	24/09/2018	30 Dias
12	**Entrega da tese**			**Fevereiro 2019**

Tabela 1 - Plano de atividades

1.5. Estrutura da Tese

Este documento encontra-se dividido em 5 capítulos principais.

No capítulo 1, foi feita uma introdução ao tema onde foram contextualizados os objetivos e o plano de trabalho seguido.

No capítulo 2, serão apresentadas as definições e conceitos de todas as plataformas e tecnologias utilizadas durante o desenvolvimento do projeto.

No capítulo 3 é realizada uma revisão às ferramentas de comunicação intrainstitucionais existentes, bem como a comparação com outras ferramentas de comunicação focadas no âmbito não-institucional.

No capítulo 4 são apresentadas as especificações, pressupostos e metodologia de desenvolvimento do trabalho realizado, onde é ainda especificada a visão global do sistema, os seus atores e o modelo relacional da aplicação. Neste capítulo é ainda explorado o inquérito inicial realizado para averiguar a pertinência do desenvolvimento da aplicação proposta neste trabalho, bem como os atores e casos de uso da mesma.

No capítulo 5, apresentam-se os detalhes de desenvolvimento da aplicação *mobile* e são expostas as suas funcionalidades.

No capítulo 6, são focados os resultados dos questionários elaborados de forma a compreender a utilidade e aplicabilidade da aplicação projetada, bem como

confrontar os resultados com aplicações existentes de forma a enriquecer e tornar concretas e concisas as conclusões.

No capítulo 7 são apresentadas as conclusões face aos resultados obtidos e sugestões de trabalho futuro para evoluir a aplicação.

2. ENQUADRAMENTO TEÓRICO

2.1. Soluções móveis

O primeiro telefone foi inventado ainda no século XIX, há mais de 135 anos atrás, e ninguém conseguiria vir a prever que o sucesso que este atingiu iria levar ao tipo de comunicações móveis que existe atualmente (Cerdeño, 2013). Segundo Redda (Redda, 2012), sensivelmente até ao início dos anos 90 o telefone servia, essencialmente, para realizar e receber chamadas. Segundo o autor, foi apenas após essa década que começou a existir alguma investigação e desenvolvimento de serviços fundamentais que viriam a ser as bases de suporte para todas as funcionalidades oferecidas pelos *smartphones* atuais. Redda (2012) afirma ainda que foi em 2002 que os *smartphones* adquiriram arquiteturas e tecnologias que vieram potenciar o como o Bluetooth, Wi-Fi e outras. Segundo Cerdeño (2013), a grande revolução ocorreu com a chegada das tecnologias 3G e 4G, que vieram permitir às pessoas comunicar rapidamente graças à Internet e aos dispositivos móveis que se encontram disseminados por quase todo o mundo, sendo utilizados por indivíduos de todas as idades. Nesse âmbito, os *smartphones* são dispositivos móveis que fornecem recursos avançados, comparativamente com um telemóvel comum (Cerdeño, 2013).

Os *smartphones* assumem um predomínio particular no universo dos dispositivos móveis, demarcando-se dos demais, uma vez que oferecem uma variedade de tecnologias num só aparelho, tal como GPS, câmara digital, telefone, *web browsers, mediaplayers*, mecanismos de pagamento, entre outros, que são disponibilizadas aos utilizadores através das já mencionadas aplicações móveis (Pires, 2016).

2.2. Ferramentas nativas versus multiplataforma

De acordo com (Ottka, 2015), o ambiente móvel é altamente fragmentado, pois existem diferentes plataformas móveis, *Android, iOS, Windows Phone*, sendo que cada uma tem várias versões em uso ativo. O autor afirma também que apesar da diversidade de sistemas operativos poder trazer novas escolhas para o mercado e para os consumidores, a mesma traz também implicações para quem desenvolve aplicações móveis para estes sistemas operativos, uma vez que os fornecedores de dispositivos tendem a desenvolver o seu próprio sistema operativo.

Em função disso, cada sistema operativo tem a sua própria linguagem de programação, interface de dispositivos, ferramentas de desenvolvimento e convenções de estilo. Isto antevê um problema pois a implementação de um conjunto de aplicações nativas significa implementar cada aplicação a partir do zero, para além de exigir conhecimento das três plataformas por parte de toda a equipa de desenvolvimento (Ottka, 2015). Cada aplicação está estritamente ligada a um ambiente de desenvolvimento específico que inclui a linguagem de programação, ciclo de vida, IDE (*Integrated Development Environment)* e API (*Appliation Programming Interface)*. O compilador é extremamente importante para o processo de desenvolvimento, auxiliando na identificação de erros de compilação e de tempo de execução, economizando tempo e esforço que depois seriam necessários para os corrigir (Pires, 2016). Assim, e de acordo com Ottka (2015), quando se pretende criar uma aplicação destinada a várias plataformas existem três soluções principais que podem ser adotadas: 1) oferecer o serviço pela internet, como uma aplicação da web; 2) usar uma ferramenta multiplataforma ou 3) usar uma ferramenta de desenvolvimento nativo.

2.2.1. Ferramentas Nativas

O mercado de *smartphones* tem vindo a ser dominado por empresas como a Google, Apple e Microsoft, sendo estas também que oferecem as melhores ferramentas para o desenvolvimento nativo. Com a sua evolução, cada uma destas empresas tem vindo a aprimorar as suas ferramentas de tal modo que o mercado se tem centrado fortemente em volta das mesmas para realizar desenvolvimento de aplicações nativas (GSMA, 2018).

Nas ferramentas nativas, o acesso aos recursos específicos da plataforma é executado através de uma API abrangente provida pelo fornecedor da plataforma e pode ser muito diferente de uma plataforma para outra. Esta API inclui acesso total a camadas de baixo e alto nível (Pires, 2016).

Segundo Gavalas & Economou (2011), camadas de baixo nível permitem desenvolver aplicações que permitem aceder facilmente ao hardware dos dispositivos móveis, o que inclui câmaras, Bluetooth e sensores como GPS, bússola, microfone, luminosidade e proximidade. Os autores afirmam também que camadas de alto nível permitem o acesso a *widgets* de interface, gestos, *multi-touch* e interação com outros aplicações como calendário, e-mail, contatos, telefone e, também, permitem uma conexão fácil a serviços na *cloud* e na web. Todos esses recursos dão acesso a várias possibilidades que podem ser usadas no desenvolvimento de aplicações criativos e inovadores (Gavalas & Economou, 2011).

O acesso a camadas de alto nível permite ainda que os programadores de aplicações possam conferir uma melhor aparência à *app* e uma melhor experiência de utilização da mesma (Pires, 2016). Segundo os mesmos autores, o tipo de *layout* e *design* aconselhado para cada aplicação depende bastante da plataforma para a qual esta se destina, devido a fatores como a existência de diferentes ciclos de vida das plataformas e componentes.

A definição destes designs prende-se com requisitos de usabilidade e *user experience* (ou UI e UX, respetivamente), que podem ser a diferença entre a adoção massiva ou o abandono de uma aplicação. Isto porque os utilizadores têm algumas noções pré-concebidas do aspeto que determinadas aplicações apresentam tipicamente e podem inclusivamente reagir mal a mudanças bruscas de *layouts* ou *designs*, sobretudo quando isso implica uma aprendizagem de manuseio da aplicação (Merz, Tuch, & Opwis, 2016). Os autores salientam também que os utilizadores de dispositivos móveis tendem a ser cada vez menos pacientes com aplicações que apresentem erros ou navegação complexa e que, por esse motivo, a qualidade das aplicações é algo cada vez mais importante.

Após a conclusão do desenvolvimento do produto, decorre a fase de exportação da aplicação. No caso da plataforma Android, trata-se de um arquivo *.apk*. No caso da plataforma iOS, denomina-se *.ipa* e, no caso da plataforma Windows, trata-se de um *.appx* (Pires, 2016).

Na fase de aquisição do produto, os utilizadores fazem o download das aplicações nas lojas geridas por cada um dos fornecedores. Procede-se, seguidamente, à descrição breve dos requisitos de cada uma das três principais ferramentas nativas.

2.2.1.1. ANDROID

As *apps* desenvolvidas para o sistema operativo Android necessitam de ser programadas em através da linguagem Java e do *Software Development Kit* (SDK) do Android (Singh, 2014). Segundo mesmo autor, este SDK inclui um conjunto de ferramentas: um depurador, bibliotecas de software, um Simulador Android, bem como um Ambiente de Desenvolvimento Integrado (IDE). Os criadores do *Android* geralmente usam o Eclipse ou o Android Studio.

Segundo Thamizharasi (2016), o *Android Studio* é um projeto do Google baseado na plataforma *IntelliJ IDEA*, o qual foi lançado em 2013. Por sua vez, o Eclipse é um IDE já foi amplamente utilizado, mas, foi-lhe adicionado um conjunto de *plugins* com o intuito de melhorar o desenvolvimento do Android (Singh, 2014). Singh (2014) e Thamizharasi (2016) afirmam também que as interfaces do Android são construídas em XML e que nos doisIDEs é possível construir interfaces arrastando e soltando componentes ou escrevendo diretamente o XML.

Segundo Pires (2016) Tanto o Eclipse como o *Android Studio* suportam a execução de aplicações integradas com o *Google Cloud Platform*. O autor afirma também que esta integração permite usufruir de funcionalidades como o backup de dados do utilizador, gerir serviços de notificação e integrar a aplicação no *Google Maps Service*.

Atualmente, existe uma grande variedade de dispositivos Android, com diferentes dimensões, densidade de pixéis, processamento, etc. que podem tornar o desenvolvimento de aplicações nesta plataforma um verdadeiro desafio (Thamizharasi, 2016).

2.2.1.2. IOS

Segundo Ottka (2015), o iOS da Apple foi originalmente lançado em 2007 junto com o primeiro iPhone, conhecido como iPhone OS, sendo posteriormente apelidado de iOS e desenvolvido de forma a suportar outros produtos Apple, como iPad e iPod.

Segundo o mesmo autor, o iOS tem um número relativamente pequeno de versões (todas elas são fornecidas pela Apple) e além disso suporta apenas um pequeno número de dispositivos quando comparado ao Android.

Outros autores como Tracy (2012) afirmam que as aplicações iOS eram desenvolvidas em Objective-C, mas, desde 2014, foi adotada uma nova linguagem de programação - Swift. O autor explica ainda que a adoção do Swift se deveu ao facto de esta linguagem se apresentar como mais robusta e menos propensa a causar inconveniências dependentes de erros de compilação apresentando-se assim como uma linguagem mais moderna que ainda suporta alguns mecanismos antigos de linguagens de programação, como macros e arquivos de *Heading*.

Por sua vez Goadrich & Rogers (2011) afirma que o Objective-C fornece muitos mecanismos que, por exemplo, o Java não possui, tais como blocos de código, categorias para estender as funcionalidades de classe e funções lambda do Objective-C (que são mais simples de usar do que as de Java). Os autores afirmam também que o SDK da iOS permite também aceder a recursos dos *smartphones* da Apple, tal como a câmara, eventos *multi-touch***,** acelerómetro, GPS e redes.

O IDE da Apple para desenvolver aplicações para iOS é chamado de *Xcode* e além de fornecer a complementação de código e uma ferramenta de construção de interface, ele possui um depurador poderoso e um simulador confiável e rápido (Goadrich & Rogers, 2011).

A interação entre o utilizador e o aplicação iOS é concretizada através de um mecanismo chamado *View Controller***,** os quais fazem a conexão entre os dados do aplicação e sua aparência visual (Tracy, 2012).

Ambas as linguagens podem coexistir no mesmo aplicação, para que as bibliotecas do Objective-C possam ser integradas no Swift (Pires, 2016).

2.2.1.3. WINDOWS

Segundo Wasserman (2010), as aplicações Windows podem ser desenvolvidas através de linguagens de programação como C#, C++, Javascript recorrendo ao software *Visual Studio*. O autor afirma também que a utilização de C# pode ser combinada com XAML UI, enquanto o C++ pode ser combinado com DirectX e JavaScript, em HTML/CSS, possibilitando que o desenvolvimento possa ser com

ferramentas mais orientadas para a programação orientada para objetos ou para a programação na Web.

Tipicamente as aplicações Windows podem ser desenvolvidas para *Windows Phone* (quase abandonado) e para o próprio ambiente Windows podendo ser igualmente distribuído na *Windows Store* e *Windows Phone Store*. Esta abordagem permite que o código possa ser partilhado entre plataformas mantendo, no entanto, a sua natividade (Gavalas & Economou, 2011).

Gronli et al (2014) afirmam também que no que diz respeito ao IDE *Visual Studio*, o que serve para desenvolver aplicações do Windows, Pires (2016) menciona que este recorre a um construtor de interfaces ou permite o desenvolvimento dos mesmos diretamente, através do arquivo XAML (*Extensible Application Markup Language*). O XAML é uma linguagem baseada em XML desenvolvida pela Microsoft que é usada para construir interfaces de utilizador e definir os valores de suas propriedades.

Os mesmo autores afirmam que nas aplicações Windows, cada arquivo de página XAML tem um arquivo *.cs* associado, onde fica a lógica dessa mesma página. Essa combinação é traduzida no compilador, combinando a saída do arquivo XAML e o arquivo *.cs* em uma única classe o que "forma" a uma janela da aplicação.

2.2.2. Ferramentas Multiplataforma

Segundo Pires (2016), a principal vantagem e objetivo de realizar o desenvolvimento de software recorrendo a IDEs multiplataforma reside na reutilização do código entre as plataformas a que a aplicação se destina: deste modo, torna-se possível evitar a repetição de desenvolvimentos ao longo de novas plataformas, bem como otimizar o tempo e esforço de desenvolvimento e reduzir custos de desenvolvimentos associados. Ottka (2015) defende o mesmo ponto de vista, referindo que a utilização de ferramentas, evidenciando ainda que, desta forma, as aplicações desenvolvidas podem ser distribuídas e instaladas nos *smartphones* destino como se se tratasse de uma *app* nativa.

Existem atualmente várias ferramentas multiplataforma no mercado como a Ionic, Phonegap, Xamarin, entre outras. A maioria delas permite o uso de uma única linguagem de desenvolvimento para conseguir exportar aplicações para várias outras plataformas como *iOS, Android* e *Windows Phone***.** Neste contexto, a ferramenta

Xamarin merece especial atenção devido não só a possibilidade de exportar para múltiplas plataformas, como também pelo facto de conseguir que as aplicações sejam mais robustas. Isto é, como *Xamarin* permite o desenvolvimento para qualquer plataforma (*Android, iOS, Windows Phone*), o *workflow* de desenvolvimento é o mesmo e evita que seja necessário replicar o código para desenvolver a mesma aplicação para diferentes plataformas. Além disso, o comportamento, fluidez e facilidade de alterar código das aplicações desenvolvidas em *Xamarin* assemelha-se às aplicações nativas das plataformas para onde a aplicação é exportada. De facto, estes são fortes fatores de diferenciação do *Xamarin* face a outras ferramentas.

Para o presente trabalho foi utilizada a ferramenta Xamarin, pelas razões atrás mencionadas e ao contacto profissional prévio com essa plataforma assim como com a linguagem C#, uma das possíveis de ser usada com esta plataforma. A opção pelo C# sustentou-se no facto de se tratar de uma linguagem de programação amplamente difundida, usada por milhões de programadores em todo o mundo, conformando uma linguagem muito simples e fácil de aprender. Para além disso, tem sido amplamente utilizada na indústria por grandes empresas, tornando-a uma das linguagens de programação mais populares no desenvolvimento de *software* comercial e de negócios. O Xamarin, além de ser uma ferramenta com uma curva de aprendizagem relativamente reduzida e de fácil compreensão, possui também funcionalidades embutidas como o *Intellisense* que tornam possível corrigir eventuais erros e criar funções através de recursos disponíveis na internet. Para além disso, não se pode descurar o facto de estar já bastante enraizada no mercado, concretamente, por uma das maiores empresas de software – a Microsoft.

2.3. Xamarin

O Xamarin é uma plataforma de aplicação comercial baseada no projeto de código aberto, lançado em 2001 pela Ximian. A Ximian foi adquirida pela Novell em 2003 e pela Microsoft em 2016. O Xamarin atualmente suporta o desenvolvimento de aplicações para iOS, Android e Windows Phone. A principal linguagem de programação do Xamarin é o C#, mas também é possível fazer o desenvolvimento usando F#

O Xamarin até 2016 oferecia aos desenvolvedores o seu próprio IDE (Xamarin Studio). Com a aquisição da Microsoft, o Xamarin foi integrado no Visual Studio, um IDE bastante robusto da Microsoft.

Para desenvolver aplicações com o Xamarin é necessário recorrer a controlos de interface nativos e standard. De destacar que as aplicações desenvolvidas não se limitam a ser semelhantes às aplicações nativas. De facto, antes pelo contrário elas comportam-se efetivamente como aplicações nativas (Figura 2).

Figura 2-SDK, API e Performance Nativas

No que respeita às *API's* de cada plataforma, o Xamarin suporta o acesso total a todas elas. Todas as funcionalidades específicas de cada plataforma são expostas pelo Xamarin de maneira a poderem ser usadas em C#. Em termos de performance o Xamarin tira partido da aceleração de *hardware* de cada plataforma e as aplicações são compiladas para garantir um desempenho nativo, o que não pode ser garantido por soluções que interpretam código em tempo de execução.

Com o Xamarin é possível utilizar a mesma linguagem, *API's* e estruturadas de dados para partilhar em média 75% do código das aplicações nas várias plataformas móveis, através de *Portable Class Libraries* e *.NET Standard***.** A Figura 3 ilustra a partilha de código entre plataformas conseguida com o C# e Xamarin. Existe uma camada comum onde código C# é partilhado ao longo de todas as plataformas. O código partilhado é, essencialmente, o comportamento da aplicação face aos eventos e pedidos que o utilizador realiza. Dentro do código partilhado existem serviços como autenticação, envio de mensagens, gestão da base de dados local, entre outros.

Existe depois uma camada específica a cada plataforma. Neste caso, o código C# corresponde essencialmente a serviços e protocolos que são necessários implementar para aceder a recursos da plataforma como, por exemplo, acesso à lista de contactos, acesso à galeria de fotografias, comportamento dos ecrãs, etc.

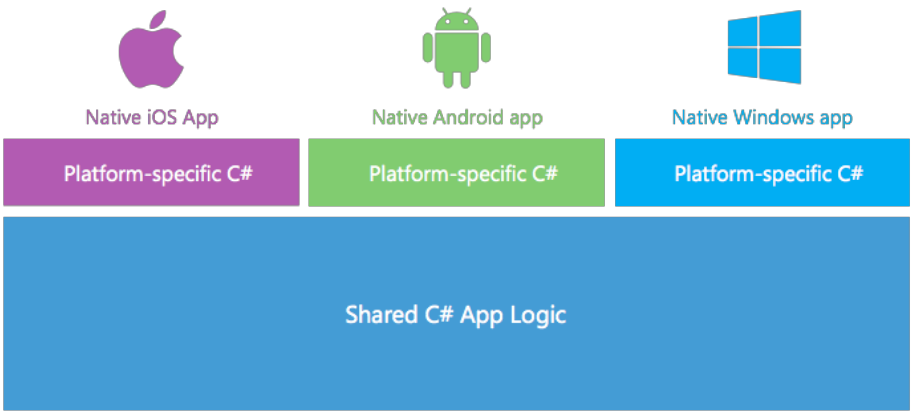

Figura 3-Partilha de código entre plataformas

O desenvolvimento em Xamarin pode ser realizado através do *Visual Studio* que existe para as versões Windows e Mac OSX. É um IDE focado na tecnologia .NET *framework* e suporta linguagens como C, C++, J# e C#. É composto por uma família de produtos, ferramentas e tecnologias que podem ser utilizadas para criar aplicações de alto desempenho. Esta versão pode criar aplicações para Windows, Web, Jogos, e aplicações para telemóveis *iOS, Windows Phone e Android*. Uma representação do ambiente de desenvolvimento encontra-se ilustrada na Figura 4.

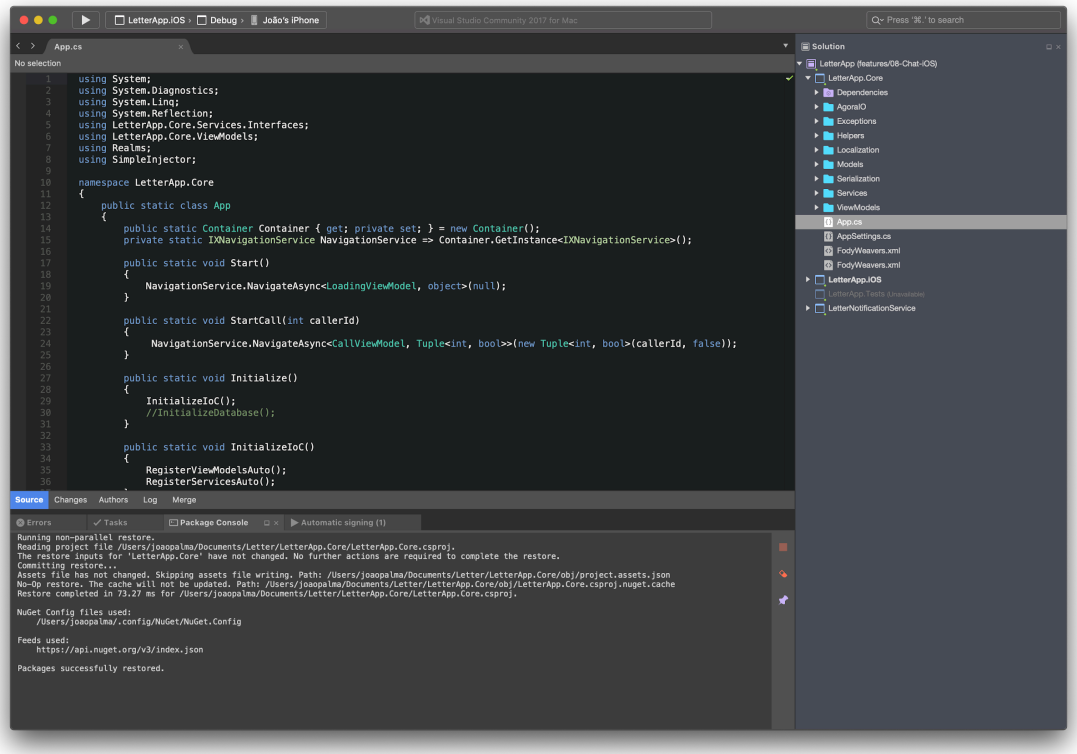

Figura 4-Visual Studio para Mac OSX

2.4. Linguagem de Programação C#

C# é uma linguagem de programação moderna desenvolvida pela Microsoft. Tal como a plataforma .NET, este tipo de linguagem favorece a resolução de problemas da integração de diferentes aplicações entre *mobile*, *desktop* e web. Por conseguinte, através desta plataforma os programadores obtiveram um ambiente que suporta os padrões da *web* emergentes e que fornecem fácil integração com outras aplicações e que, simultaneamente, permite o desenvolvimento de diferentes aplicações: aplicações Web, aplicações para computadores, telemóveis, criação de jogos, entre outros. C# é também uma linguagem de alto nível, semelhante a Java e Swift. A maioria dos programas e aplicações C# são orientados a objetos, ou seja, este tipo de programação é um modelo onde existem diversas classes, que possuem características de um objeto da vida real. De forma sintética, as principais características do C# são:

- **Simplicidade**: pois foi elaborado para ser facilmente aprendida e usada de forma intuitiva e eficaz.
- **Modernidade**: permite gerir a memória do telemóvel de forma automática, não havendo necessidade de o programador implementar esta tarefa. Essa gestão passa por limpar recursos que o telemóvel não está a utilizar de momento, de forma rápida e eficiente. Outra das vantagens é que permite manter em segurança a confidencialidade de dados pessoais de cada utilizador pois possui um modelo de segurança sólido.
- **Orientação a Objetos:** facilita a alteração e implementação de novas funcionalidades, permitindo manter o software atualizado.
- **Versatilidade**: pode ser usada para uma variedade de aplicações que são suportados pela plataforma .NET, sejam eles aplicações *mobile*, web ou até mesmo jogos.

2.5. Metodologias e tecnologias utilizadas

Ao longo deste capítulo, são descritas as várias *frameworks*, tecnologias e linguagens adotadas para o desenvolvimento desta aplicação, cuja arquitetura e interligação de componentes se representa na Figura 5.

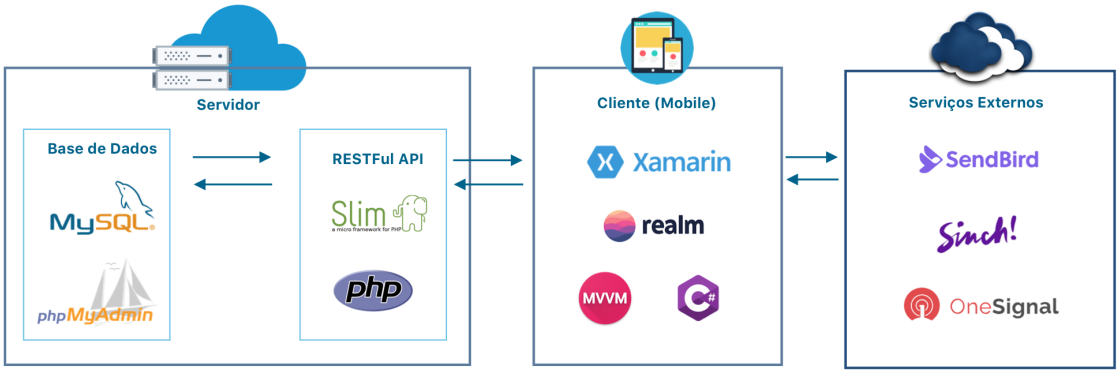

Figura 5 - Arquitetura geral da aplicação Letter Messenger

Na solução desenvolvida, existem três componentes principais: o servidor, a *app* mobile (cliente) e serviços externos. Para cada uma, será feita uma análise global das tecnologias utilizadas e explicada a razão da sua utilização. A componente de servidor está a correr com uma base de dados MySQL, para armazenar dados das organizações e utilizadores, juntamente com uma API *RESTful* em PHP, recorrendo à *framework* Slim, para que sejam gerios os pedidos feitos pelo cliente. A componente de cliente (*app)* foi desenvolvida em C#, através do IDE Visual Studio e da plataforma Xamarin, segundo o padrão de design MVVM (*Model-View-ViewModel*), é utilizado ainda a ferramenta Realm para armazenar os dados da organização e do utilizador no telemóvel. A *app* recorre ainda a serviços externos como o SendBird, Sinch e OneSignal. O serviço SendBird é utilizado para enviar e receber mensagens, sendo um serviço de *chat* que é utilizado para agilizar a componente chat da aplicação. Por sua vez, o serviço Sinch é utilizado para a realização de chamadas na aplicação enquanto que o serviço OneSignal é utilizado para receber notificações dentro da aplicação.

2.5.1. MYSQL & PHPMYADMIN

Para a componente de servidor, foi necessária a criação de uma base de dados tendo sido escolhido o MYSQL, um sistema de gestão de bases de dados (SGBD) relacional que utiliza a linguagem SQL como interface. A sua principal função é armazenar e disponibilizar dados de outras aplicações, assim como realizar a gestão de acessos e permissões aos mesmos. As bases de dados MySQL têm compatibilidade com módulos de diversas linguagens como PHP, C# e Java. Uma das suas principais características é a sua portabilidade, ou seja, a capacidade de suportar qualquer plataforma atual. Além disso, possui ainda um excelente desempenho e estabilidade, sustentando, igualmente, o controlo transacional, *triggers* e funções. Para a gestão da base de dados, foi utilizada a ferramenta phpMyAdmin.

2.5.2. REST

Na sequência da necessidade acima mencionada de usar serviços, usamos REST porque apresenta a agilidade e flexibilidade necessárias para conseguir o processamento rápido de pedidos. Além disso, permite também o uso de sintaxes distintas como XML ou JSON (sendo esta última a escolhida) e é também um protocolo bastante utilizado. O REST é uma técnica de engenharia de software para sistemas *hipermédia* distribuídos. Caracteriza-se como um conjunto de restrições na arquitetura dos componentes, conectores e elementos de dados dentro destes sistemas. O REST foca-se no propósito do componente, nas restrições da sua interação com outros componentes e na sua interpretação de elementos de dados significativos, ignorando os detalhes da implementação do componente e a sintaxe do protocolo. O REST pode ser descrito como um "mapa comportamental" da aplicação: uma rede de caminhos dentro de aplicações ou websites onde o utilizador navega, selecionando as ligações que pretende e tendo como resultado a página seguinte que está a ser transferida e que é posteriormente apresentada (Adamczyk *et al,* 2011). Segundo os mesmos autores, o *REST* permite o desenvolvimento de serviços web. Estes caracterizam-se como *RESTful* sempre que respeitem as restrições existentes à sua arquitetura. Algumas das propriedades da arquitetura *REST* incluem:

- Performance.
- Escalabilidade.

- Simplicidade de interfaces.
- Adaptação à mudança.
- Visibilidade na comunicação.
- Portabilidade.
- Fiabilidade e resistência a falhas.

2.5.3. APIs RESTful

Tendo em conta a escolha de REST para os serviços, foi necessária a criação de uma API RESTful para que o acesso à base de dados seja mais rápido, simples e seguro. O termo "*RESTful API*" é usado para descrever uma arquitetura de comunicação que permite interligar diferentes dispositivos como telemóvel, computador ou outro dispositivo eletrónico, de forma a inserir, extrair ou apagar informação da base de dados. Na prática, uma API possibilita maior rapidez e segurança da aplicação tornando o sistema mais eficiente, permitindo adicionar facilmente novas funcionalidades e modificar outras pré-existentes sem grande esforço por parte do programador. Sendo uma ferramenta de fácil uso, tanto para os utilizadores finais como para os gestores na tomada de decisão, provando que a qualidade dessa API é essencial para que a interação do utilizador com a aplicação desenvolvida seja eficaz.

2.5.4. Web Services

Os *Web Services* são soluções utilizadas para integrar e promover a comunicação entre diferentes sistemas/plataformas e outras aplicações. Estes são identificados por um *Uniform Resource Identifier* (URI) e permitem às aplicações o envio e receção de dados nos formatos XML ou JSON. Como tal, possibilitam que novas aplicações possam interagir com as existentes, assim como, a interação entre sistemas desenvolvidos em plataformas diferentes. Com esta tecnologia, uma aplicação pode recorrer a outra para efetuar determinada tarefa, independentemente do sistema ou linguagem serem ou não comuns entre ambas. Os *Web Services* promovem a dinamização da comunicação entre sistemas, tornando-a mais eficaz, simples e segura. Para este projeto, os *Web Services* foram necessários para conectar a base de dados que está alocada no servidor e a aplicação *mobile*, para que seja possível fazer transferência de dados entre as duas plataformas.

2.5.5. JSON

Para que exista uma estrutura de dados de modo a que seja mais simples compreender o tipo de dados que estão a ser transferidos entre a base de dados e a aplicação, foi necessária a utilização de *Java Script Object Notation* (JSON). O JSON é uma formatação simples e leve utilizada na troca de dados. Esta formatação é bastante prática devido à facilidade de escrita e leitura. A linguagem baseia-se num subconjunto da linguagem de programação Java Script. É principalmente utilizado na transmissão de dados entre servidores e aplicações web, sendo que é em formato texto e completamente independente de linguagem, existindo código para gerar e realizar o Parse de dados JSON em diversas linguagens de programação. São estas características que fazem de JSON o formato ideal para troca de dados.

O JSON é constituído por duas estruturas:

- Uma coleção de pares nome/valor que, em várias linguagens, caracteriza-se como um objeto, record, estrutura, dicionário, *hash table, keyed list* ou *array* associativo.
- Uma lista ordenada de valores. Na maioria das linguagens caracteriza-se por *array*, vetor, lista ou sequência.

  Estas estruturas de dados são universais, sendo que de uma forma ou de outra, são suportadas por todas as linguagens de programação moderna.

Qualquer formato de troca de dados independente de linguagens de programação pode basear-se nestas estruturas.

2.5.6. PHP

A linguagem PHP foi a escolhida para gerir a ligação entre a bases de dados e a aplicação *mobile*. PHP, ou "*PHP: Hypertext Processor*" é uma linguagem de programação que *open source* que é maioritariamente utilizada para realizar o desenvolvimento de aplicações e serviços web (PHP, 2018). Esta linguagem é conhecida pela sua versatilidade, facilidade de uso e de integração, bem como devido à sua rápida curva de aprendizagem. O PHP utiliza também influências de outras linguagens como C, Java ou Perl e pode ser facilmente embebida em páginas construídas sobre HTML.

2.5.7. Slim

Esta framework foi utilizada para desenvolver a RESTful API em que é feito o mapeamento dos pedidos vindos da aplicação mobile com a base de dados. Slim é uma framework PHP que permite o desenvolvimento de aplicações web e APIs para os mais diversos fins (Slim, 2018). Segundo o seu website oficial, esta framework encontra-se preparada com várias funcionalidades úteis para engenheiros de Software. Além disso permite também a implementação de mensagens HTTP PSR-7, as quais permitem inspecionar e manipular parâmetros como *status, URI, headers***,** entre outros. O website oficial afirma ainda que esta *framework* permite utilizar funcionalidades de *dependency injection* bem como a utilização de *middleware***.**

2.5.8. Realm

Realm é uma plataforma de desenvolvimento de aplicações que é descrita no seu website oficial como uma combinação de um servidor NoSQL e componentes de desenvolvimento *client side* (Realm, 2018)**.** Segundo a mesma fonte, esta plataforma permite a integração com sistemas *backend* de terceiros como SQL, Kafka (entre outros) e subdivide-se em dois componentes: a *Realm Database* e o *Realm Object Server*. Esta separação permite que a plataforma seja suficientemente flexível para conseguir dar resposta a vários tipos de aplicações e usos distintos. No seu website oficial são citadas algumas como aplicações offline, aplicações de coleção de dados, serviços móveis onde é necessário ter elevado grau de atenção à responsividade de utilizadores e a fluxo de dados, entre outras.

Neste trabalho, foi utilizado a componente *Realm Database* como base de dados local que grava dados no *smartphone* para um acesso mais rápido a informações e também para que seja possível o uso da aplicação quando não existe acesso à internet.

2.6. Serviços Externos

2.6.1. Sendbird

SendBird é uma API que permite implementar a serviços de *messaging-as-a-service* (SendBird, 2018). Segundo o webisite oficial, o Sendbird disponibiliza também todas as funcionalidades de *backend* necessárias para que qualquer aplicação consiga implementar serviços de mensagem em tempo real, disponibilizando-os aos utilizadores das mesmas. Além disso, esta API é atualmente suportada por *iOS (Objective-C and Swift), Android,* Web *(JavaScript),* Xamarin**,** entre várias outras. Neste trabalho, o SendBird foi utilizado para implementar os serviços de mensagem instantânea na aplicação.

2.6.2. Sinch

A Sinch é uma API que permite implementar várias funcionalidades distintas desde funcionalidades de chamadas VoIP (*Voice over IP)*, mecanismos de verificação de utilizadores via SMS (*Shot Message Service)*, chamadas de vídeo, SMS e máscaras de números de telefone (Sinch, 2018). A API de voz é suportada em plataformas *iOS, Android* e foi utilizada com esse exato propósito na aplicação Blitzchat, para que seja possível fazer chamadas telefónicas gratuitas entre utilizadores da aplicação.

2.6.3. OneSignal

O OneSignal é uma API *RESTful* que oferece serviços de *push notifications* e e-mail, integráveis em qualquer tipo de aplicação web ou móvel. A OneSignal oferece também ferramentas de marketing que permitem realizar o teste A/B, *segment targeting, tracking* de conversões, entre outras (OneSignal, 2018). Além disso, esta API oferece SDKs para os maiores IDE multiplataforma como Unity, PhoneGap, Cordova, Ionic, React Native, Intel XDK, Corona, Xamarin, Marmalade, Adobe Air e Web Push. Não obstante, é também suportada por várias plataformas como *iOS, Android*, *Windows Phone*, entre outras.

Durante este trabalho, o OneSignal foi utilizado para implementar *push notifications* na aplicação Blitzchat, para que quando uma nova mensagem seja enviada, o utilizador receba uma notificação no ecrã do seu telemóvel.

2.7. MVVM

O *Model-View-ViewModel* ou MVVM é um padrão de arquitetura para aplicações que visa estabelecer uma clara separação de responsabilidades e tornar uma aplicação *mobile* mais fácil de atualizar. O MVVM assemelha-se em alguns aspetos a outros padrões de *design*, nomeadamente o *Model View Controler* (MVC) e o *Model View Presenter* (MVP). Pode-se dizer que o MVVM é uma especificação do MVC adaptado para a arquitetura mobile. Este padrão de arquitetura foi adotado para este projeto devido à versatilidade que apresenta face a outras arquiteturas como o MVC, permitindo uma melhor e organização e estruturação dos comportamentos e responsividade da aplicação durante a sua utilização. Além disso, o MVVM apresenta maior facilidade de aplicação com a ferramenta Xamarin que as restantes arquiteturas.

A Figura 6 ilustra o funcionamento do padrão MVVM (Fonte: Lauren Bugnion, Messenger and View Model Services in MVVM – abril, 2015).

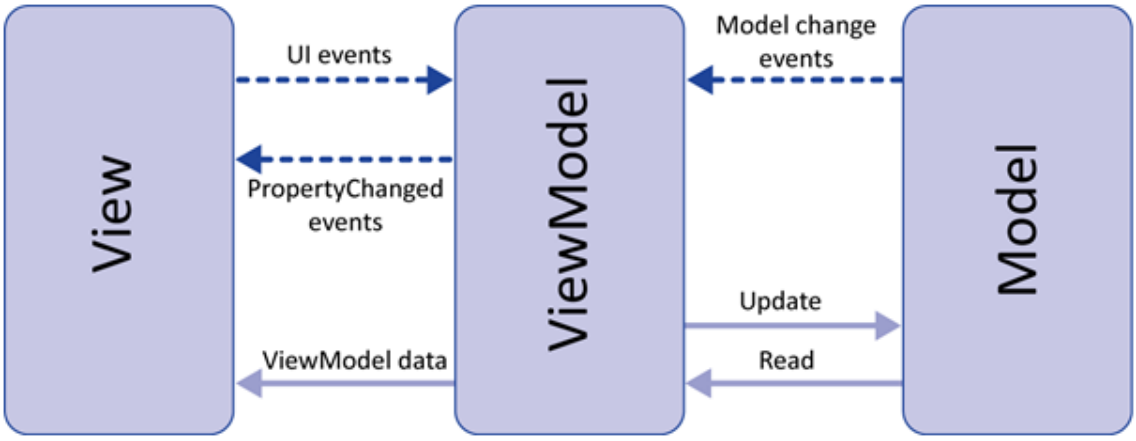

Figura 6- Arquitetura MVVM

Este padrão permite ter uma visão da clara separação da interface com o utilizador (*View)*, lógica de apresentação (*ViewModel*) e os dados (*Model)*. Esta filosofia de desenvolvimento permite a separação de responsabilidades e é possível evoluir e manter as aplicações de forma mais eficaz. A *View* interage com o *ViewModel* através

de *data-binding,* notificando a ocorrência de eventos e o disparo de comandos. O *ViewModel* responde a esta notificação realizando uma ação modelo, como por exemplo obter dados, atualizar ou inserir informação no modelo. A responsabilidade da *View* é definir a aparência ou estrutura daquilo que o utilizador vê no ecrã do seu *smartphone*. A *View* referencia o *ViewModel* através da propriedade *DataContext*. Os controlos da *View* são preenchidos com propriedades ou comandos expostos pela *ViewModel*. Já o *Model* encapsula a lógica de negócios e os dados. E nada mais é do que o modelo de domínio de uma aplicação, representando as classes de negócio que serão utilizadas numa determinada aplicação. O *Model* também contém os papéis e a validação dos dados de acordo com o negócio cuja aplicação visa atender. Por fim, a responsabilidade da *ViewModel* é disponibilizar para a *View* uma lógica de apresentação. O *ViewModel* não tem nenhum conhecimento específico sobre a *View*. Esta implementa propriedades e comandos para que a *View* possa preencher os seus controlos e notifica a mesma, seja através de eventos ou notificação de alterações. O *ViewModel* é uma peça fundamental no MVVM, uma vez que é esta que vai coordenar as interações da *View* com o *Model*. Para além disto, pode também implementar lógicas e validações para garantir a consistência de dados.

3. FERRAMENTAS DE COMUNICAÇÃO INTRAINSTITUCIONAL: UMA REVISÃO

As ferramentas de comunicação apresentam diferentes funcionalidades dependendo do propósito para as quais foram idealizadas. Por exemplo, uma aplicação pensada para ser utilizada em âmbito empresarial (ex.: uma universidade) apresenta restrições de privacidade mais complexas que uma ferramenta de comunicação social. Para a análise referente a ferramentas de comunicação intrainstitucional existentes, e cujo objetivo/funcionalidades possam ser comparadas com o objetivo proposto nesta tese, escolhemos quatro aplicações: *HipChat, Slack, Skype for business, Ring Central Glip, Whatsapp* e *Facebook Messenger*. As quatro primeiras são maioritariamente utilizadas em contexto empresarial e as duas últimas são maioritariamente utilizadas em contexto social. Foram avaliadas quanto às suas semelhanças, diferenças e mais-valias, com o intuito de fazer uma comparação com aquilo que é o nosso propósito.

3.1. Hipchat

O *Hipchat* é uma aplicação de chat instantâneo que facilita o trabalho em grupo. Concretamente, os utilizadores podem criar grupos e falar entre si (Atlassian, 2018). Esta aplicação desenvolvida pela Atlassian tem integração com outros produtos desta empresa, tal como o JIRA ou Bitbucket, o que facilita a partilha de código, e permite planear os próximos passos a dar no projeto. Além disso, esta aplicação permite o envio de mensagens privadas, partilhas de ecrã, ficheiros e mensagens de vídeo, tendo sido desenvolvida para o contexto empresarial. O *HipChat* é uma ferramenta de trabalho, específico para grupos de trabalho, mas não é uma aplicação ideal para grandes empresas ou instituições, tal como universidades uma vez que apresenta uma curva de aprendizagem moderada e é pouco ***user friendly*** para pessoas que não estejam habituadas a utilizar este tipo de aplicações. O *HipChat* é uma aplicação paga que é disponibilizada para *Mac, Linux, iOS* e *Android* mas foi recentemente descontinuada e substituída pelo *Slack.*

3.2. Slack

O *Slack* é uma outra aplicação de comunicação instantânea, a qual possui várias semelhanças com o *HipChat*, concretamente, ambas são ferramentas de trabalho específicas para grupos de utilizadores que estão a trabalhar, em conjunto, em projetos (Slack, 2018), o que representa uma noção que não é a pretendida com a *app* que nos propomos desenvolver. Tal como o *HipChat*, o *Slack* centraliza as comunicações entre colaboradores de empresas e permite a criação de diferentes conversas para assuntos distintos, bem como o *upload/download* de ficheiros e a manipulação de ficheiros *spreadsheet*. Qualquer pessoa pode inscrever-se na plataforma utilizando o seu e-mail, sendo esta acessível via web. Ao inscrever, é possível convidar outros colegas ou juntar-se a outras salas de chat de aplicação, bem como realizar a procura de pessoas via e-mail. Esta aplicação tem o mesmo objetivo que a *app* BlitzChat, ou seja, reduzir o tempo que se necessita para encontrar o contacto eletrónico de pessoas que trabalhem na mesma empresa, embora não permita encontrar colaboradores de uma empresa de forma simples como fazer uma pesquisa por nome, ou e-mail ou número de telefone. Também não permite criar divisões para cada departamento dentro da empresa.

3.3. Skype for Business

O Skype for Business é um software da Microsoft de troca de mensagens e vídeo instantâneo para empresas (Skype, 2019). Este software permite a realização de reuniões com outros utilizadores que possuam contas Skype, realizar convites via URL, partilhas de ecrã, *white boards* para desenhar esquemas em tempo real e até carregamento de ficheiros PowerPoint. O software está disponível para várias plataformas como PC/Mac, iPhone, iPad e Android. Apesar de todas as suas funcionalidades, o Skype For Business exige também que os utilizadores cedam o seu e-mail e até número de telefone para utilizar o sistema. Além disso, apesar de ser direcionado para empresas, é maioritariamente focado para o contacto com utilizadores que não pertençam à empresa do utilizador. A aplicação Blitzchat pretende organizar as comunicações entre elementos intraempresariais, representando um âmbito de utilização diferente.

3.4. Ring Central Glip

O *Ring Central Glip* é um software de gestão de equipas intraempresariais que permite realizar a troca de mensagens instantâneas pessoais e em grupo, bem como a partilha de ecrã, partilha de ficheiros, e *video chatting* (RingCentral, 2019)**.** Permite ainda outras funcionalidades como um gestor de tarefas e calendário para equipas. Este software também está disponível para PC/Mac, iPhone, iPad e Android. Este é talvez a solução no mercado que se assemelha mais ao que se pretende desenvolver com a aplicação Blitzchat. Apesar disso, esta aplicação não permite a definição de papéis de utilizador dentro da mesma (administrador, funcionário, etc) e também não permite encontrar colaboradores de uma empresa fazendo pesquisas avançadas e não permite criar divisões privadas só para administradores ou divisões para cada departamento dentro da empresa em função das necessidades.

3.5. Whatsapp

O Whatsapp é agora uma aplicação da empresa Facebook que permite aos seus utilizadores conversarem entre si através de serviços de mensagens instantâneas, bem como o envio de ficheiros como áudio ou imagens, e ainda a realização de chamadas de voz ou de vídeo via internet (Whatsapp, 2018). Esta aplicação é muito similar à aplicação de mensagens instantâneas do Facebook tendo, no entanto, sido a pioneira no uso do número de telefone do utilizador como o indicador principal na aplicação. Isto é, ao instalar a aplicação, o utilizador tem imediatamente como contactos todos os utilizadores que também utilizem Whatsapp e cujo utilizador possua o número de telefone. O registo necessita, portanto, do número de telefone, além do nome de utilizador. Esta aplicação está também disponível para várias plataformas como a Web, *iOS, Android* e *Windows Phone* e é vocacionada para o uso social, ao contrário da aplicação Blitzchat.

3.6. Facebook Messenger

O *Facebook Messenger*, ou simplesmente *Messenger*, é uma aplicação em tudo similar à aplicação *Whatsapp*, excetuando o facto que o registo não assenta maioritariamente sobre a necessidade de introduzir o número de telefone do utilizador, embora o permita para ceder serviços de SMS (Facebook, Inc., 2018). O *Messenger* é uma aplicação que pode ser, de certa forma, vista como complemento da rede social Facebook, permitindo os utilizadores desta interagir entre si através de

serviços de mensagens e vídeo instantâneos, envio de ficheiros, entre outros. Além disto, o Messenger permite a partilha de contactos entre o Facebook e a rede social Instagram uma vez que ambas pertencem à mesma empresa.O foco desta aplicação é também o uso no âmbito social e não profissional, pelo que não se compara diretamente com a aplicação Blitzchat.

A Tabela 2 apresenta uma análise comparativa entre as várias aplicações, nos seguintes aspetos:

- Envio de mensagens instantâneas;
- Chamadas Gratuitas;
- Detalhes do Utilizador (a aplicação exige detalhes pessoais de utilizadores como e-mail, telefone ou outros para pesquisar utilizadores);
- Procura de utilizadores avançada
- Envio de Ficheiros
- Partilha de Ecrã
- Pertencer a diferentes organizações
- Aplicação intuitiva (aplicação apresenta navegação intuitiva até para pessoas que não estão muito familiarizadas com tecnologia);
- Permite configurações
- Custos

Funções	**Slack / HipChat**	**Skype for business**	**RingCentral Glip**	**Facebook Messenger / Whatsapp**	**Blitzchat**
Envio de mensagens instantâneas	SIM	SIM	SIM	SIM	SIM
Chamadas Gratuitas	SIM	SIM	SIM	SIM	SIM
Detalhes do Utilizador	NÃO	NÃO	NÃO	SIM	SIM
Procura avançada de	NÃO	NÃO	NÃO	NÃO	SIM

utilizadores avançada					
Envio de Ficheiros	SIM	SIM	SIM	SIM	SIM
Partilha de Ecrã	SIM	SIM	SIM	SIM	NÃO
Pertencer a diferentes organizações	SIM	NÃO	NÃO	NÃO	SIM
Aplicação intuitiva	NÃO	SIM	NÃO	SIM	SIM
Permite configuração administrativa a partir de um backoffice	SIM	NÃO	NÃO	NÃO	SIM
Custo	Grátis com restrições	Grátis com restrições	Pago	Grátis	Grátis

Tabela 2 - Benchmarking de soluções

A principal conclusão retirada desta análise comparativa é que existe uma lacuna que pode ser preenchida com a aplicação Blitzchat**,** proposta neste trabalho. De momento, existem várias aplicações para comunicação instantânea, como acima foi apresentado. No entanto, nenhuma preenche os vários requisitos necessários para ser utilizada por grandes instituições. Salientamos as seguintes:

- Inexistência de um *backoffice* onde administradores das instituições podem configurar diversas funcionalidades para permitir uma utilização da aplicação mais completa e uma procura avançada dos utilizadores.
- A dependência de dados pessoais como e-mail ou números de telefone para registo e pesquisa de utilizadores dentro de empresas, ignorando a privacidade dos mesmos. Esta é uma desvantagem, uma vez que o utilizador

está sempre obrigado a ceder um e-mail ou número de telefone que será público e pesquisável;

- A falta de melhores métodos de pesquisa dentro de empresas devido à exigência de e-mail ou número de telefone. Por exemplo, atualmente um utilizador não conseguirá pesquisar por outro utilizador da mesma empresa apenas sabendo elementos como o seu nome e departamento.

Quando comparada a aplicações como o *Whatsapp* ou *Messenger*, o Blitzchat não necessita de saber ou utilizar o número de telefone de colaboradores da empresa para permitir as comunicações (requisito por nós pretendido para a facilidade de utilização da *app*), bastando pesquisar por alguém através de e-mail, contacto telefónico ou pesquisar através de junção de elementos (ex.: parte do número de telefone e do nome do contato " 96 João", que permite encontrar todas as pessoas cujo nome é "joão" e cujo contacto comece por "96"). Este último tipo de pesquisa também não é possível no *Hipchat*, *Slack, Skype for Business* ou *RingCentral Glip.* O *Blitzchat* adiciona automaticamente todos os contactos de uma empresa à lista de contactos do utilizador, desde que ele se assuma como colaborador da mesma. Pelo simples facto de ter em conta qual a empresa a que o utilizador pretende e a disponibilizar automaticamente contactos e ferramentas para promover comunicações entre o mesmo e os restantes colaboradores da empresa, o Blitzchat desmarca-se de todas as soluções apresentadas.

4. ESPECIFICAÇÃO DE REQUISITOS

Este capítulo descreve a especificação de requisitos do sistema. Apresenta a visão global do mesmo, bem como os seus pressupostos de utilização e metodologia de desenvolvimento. É ainda apresentado o questionário realizado para determinar a necessidade da aplicação sob a perspetiva dos utilizadores, bem como a exposição dos vários tipos de utilizadores que a plataforma pode agregar e o modelo relacional da aplicação.

4.1. Visão Global

O funcionamento e mais-valia fundamental que se pretende com esta aplicação é que pessoas dentro de uma organização sejam capazes de comunicar de forma rápida, mesmo não possuindo contatos (email nem telefone) da outra pessoa. A aplicação foi pensada para poder ser utilizada por qualquer instituição embora, para esta tese, tenha sido focada a realidade de uma instituição de ensino superior. Foi também decidido, como requisito, que a aplicação poderia permitir a comunicação gratuita através de dois meios distintos: mensagens escritas e chamadas de voz.

4.2. Pressupostos de utilização

Inicialmente, foram definidos os seguintes pressupostos de utilização do sistema:

- existência de ligação à internet por parte da *app.*
- no que diz respeito ao servidor aplicacional e à base de dados subjacente, é também necessário que o desempenho destes esteja dimensionado em função do número de utilizadores da aplicação.
- para os utilizadores que utilizam a aplicação *mobile* IOS, é condição essencial ter a versão iOS 10 ou mais recente.
- Para poder haver comunicação por voz, é imprescindível que o *smartphone* possua microfone e que o utilizador autorize a aplicação a utilizar o mesmo. O mesmo cenário ocorre para o utilizador receber notificações.
- O *smartphone* tem ainda de conter memória suficiente para instalar a aplicação e respetivas atualizações e é condição base que o utilizador mantenha sempre a sua aplicação atualizada para um desempenho melhorado.

- necessidade de definição dos vários perfis, de forma a criar uma lista de contactos que discriminasse os papéis que cada utilizador tem dentro da organização que integra.

4.3. Metodologia de desenvolvimento

A metodologia para o desenvolvimento deste trabalho iniciou-se com o perceber as reais necessidades de um grupo de potenciais utilizadores, neste caso, numa instituição de ensino superior - a Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Viana do Castelo. Pretendeu perceber-se a pertinência e utilidade de uma aplicação como a proposta neste trabalho, para esta instituição em particular, e as suas necessidades ao nível das comunicações interinstitucionais. Para tal, foi elaborado um inquérito que teve por objetivo perceber estas necessidades. De seguida, foram definidos os requisitos da aplicação, definida a arquitetura, implementada a base de dados de suporte, criada a API RESTful e finalmente desenvolvida a aplicação móvel, com os seus vários casos de uso.

4.4. Inquérito inicial

Este inquérito teve como objetivo perceber a real necessidade de uma aplicação que permitisse a comunicação instantânea intrainstitucional, na realidade de uma Instituição de Ensino Superior. Foi feito a 85 alunos e docentes da ESTG. Do grupo de pessoas que responderam ao inquérito, a grande maioria foi alunos (85%), dos quais 45% ligados à área de informática ou afins (Figura 7 e discriminação na Tabela 3). Esta separação por área é feita para poder haver uma maior predisposição destes utilizadores para usar este tipo de solução.

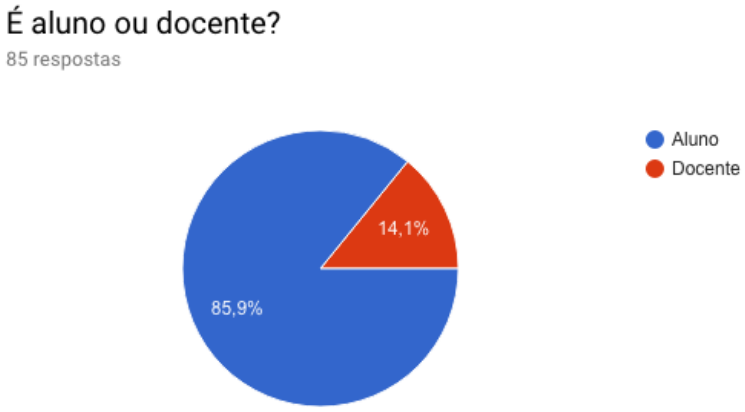

Figura 7- Rácio de alunos/docentes questionados.

Curso	Nº de Pessoas
Engenharia Informática	33
Alimentar	4
Turismo	5
Engenharia Software	5
Engenharia Civil	3
Mestrado de Logística	1
Mecânica	4
Gestão	9
Engenharia Computação Gráfica e Multimédia	2
Tecnologia e Programação de Sistemas de Informação	2
Engenharia de Redes e Sistemas de Computadores	2
Engenharia Mecatrónica	1
Engenharia Eletrónica e Redes de Computadores	1
Design do Produto	1
Mestrado em Contabilidade e Finanças	1
Engenharia Civil e do Ambiente	1
Gestão Hoteleira	1

Mestrado de Gestão das Organizações	1
Outros Cursos	7

Tabela 3 - Descriminação de pessoas inquiridas por curso.

Como é possível observar na Figura 7, a grande maioria dos inquiridos corresponderam a alunos do IPVC, sendo apenas 14.1% dos entrevistados pertencentes ao corpo de docentes da instituição. A grande maioria dos inquiridos pertencia ao curso de Engenharia Informática, como demonstra a Tabela 3.

Outra questão colocada teve a ver com a necessidade que, em algum momento, já sentiu de necessitar contactar de forma imediata alguém da instituição, verificando-se respostas positivas em aproximadamente 84% dos casos (Figura 8).

Figura 8 – Gráfico de análise de necessidades de comunicação I.

Nos casos em que houve necessidade, uma grande maioria nunca conseguiu (13%) ou apenas às vezes (75%) conseguiu o contacto (Figura 9). Quase 50% dos inquiridos afirma que já teve dificuldades em comunicar com outros colegas ou docentes dentro da instituição, como demonstra a Figura 8. Apesar disso, cerca de 73% afirma (pela Figura 9) que estes constrangimentos da comunicação aconteciam "às vezes", ou seja, que ainda existe espaço para aprimorar o modo como as comunicações intrainstitucionais são realizadas.

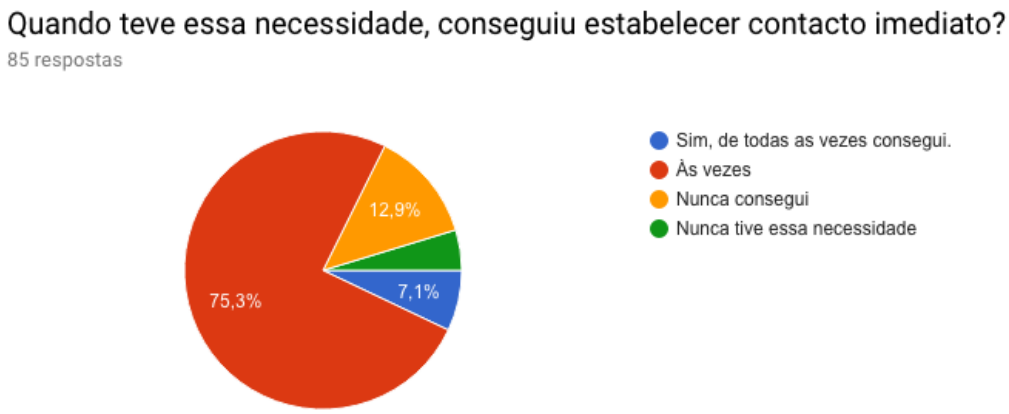

Figura 9 - Gráfico de análise de necessidades de comunicação II.

Quando interrogados como tentaram estabelecer o contacto (Figura 10), pouco mais de 50% referiram ter escolhido o envio de email e ter tido a sorte da pessoa ter respondido de imediato; em 25% dos casos foi usado o contacto telefónico e noutros 25% não houve forma de contacto.

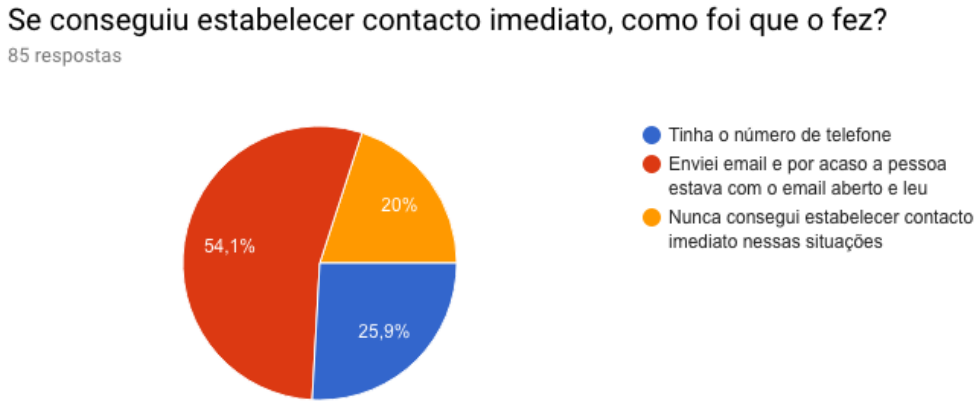

Figura 10 - Gráfico de análise de necessidades de comunicação III.

Além disso, a Figura 10 evidencia também que mais de metade das situações onde o contacto imediato foi estabelecido deveu-se essencialmente ao acaso (54.1% das respostas afirmam que "enviei email e por acaso a pessoa estava com ele aberto e leu") e que 20% dos inquiridos nunca conseguiu estabelecer a comunicação.

Aplicações	**Nº de Utilizadores**
Messenger	3
Jabber	1

Telegram	1
WhatsApp	5
Slack	1
Skype	1

Tabela 4 - Tabela de respostas para a questão "Se conhecer alguma aplicação que permita este tipo de comunicação, indique o nome.".

Em relação às aplicações mencionadas como uma opção para estabelecer esta comunicação, as indicadas são mencionadas na Tabela 4. Finalmente, mais de 90% dos inquiridos referiram a grande relevância e utilidade de uma aplicação destas neste contexto (Figura 11).

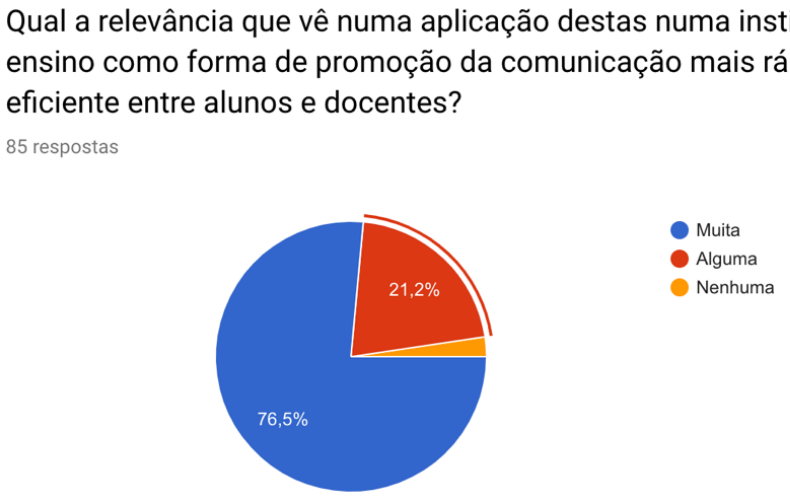

Figura 11 - Gráfico de análise da relevância da aplicação.

Dentro das aplicações alternativas percecionadas pelos inquiridos, salienta-se a maioria nas escolhas das aplicações *Messenger* e *Whatsapp*, que não são indicadas para uso profissional/intrainstitucional, como demonstrado na Tabela 4. Por fim, 76.5% dos inquiridos admitiu que uma aplicação como o Blitzchat apresenta-se como uma abordagem muito pertinente para o meio intrainstitucional, como demonstra a Figura 11. Após a análise das respostas obtidas, conclui-se que existe interesse e espaço de inovação ao nível do processo para o desenvolvimento de uma aplicação como a proposta no presente trabalho.

4.5. Atores e casos de uso modelo relacional da aplicação

Esta secção descreve os atores do sistema, bem como os casos de uso de cada um. A aplicação subdivide-se em dois grandes atores: funcionário e administrador.

4.5.1. Funcionário

O funcionário representa o utilizador base que usará a aplicação Blitzchat. Este terá acesso aos casos de uso ilustrados na Figura 12 e que se encontram detalhados nas Tabela 5, 6, 7, 8, 9 e 10.

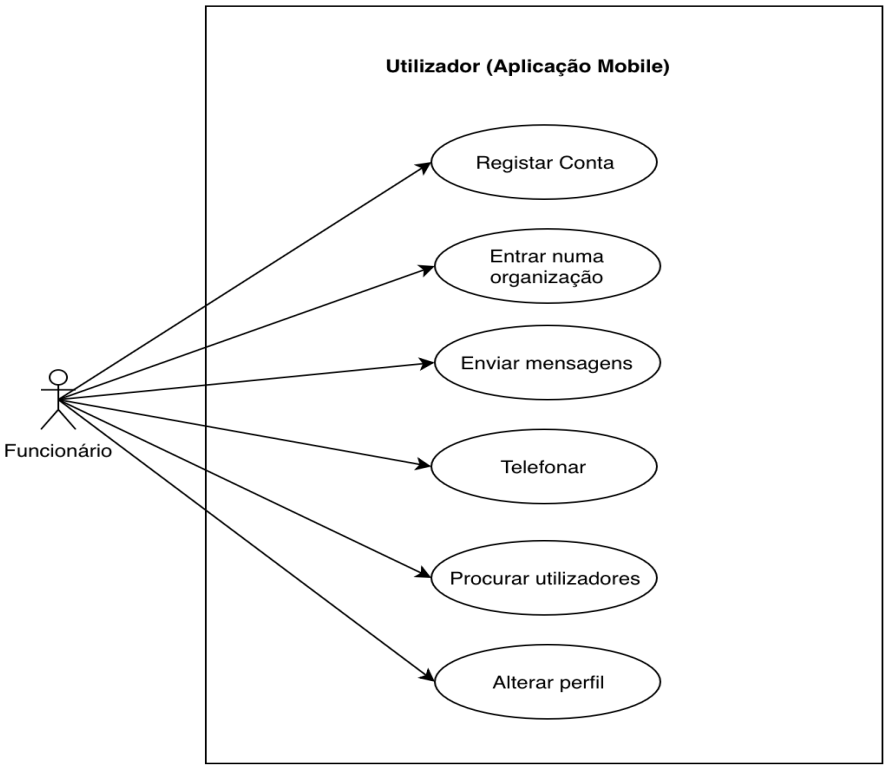

Figura 12 - Funcionalidades acessíveis ao ator "Funcionário".

Identificador	Caso de Uso (uc_letter_funcionario_01)
Nome	Criar conta
Descrição Sumário	Este caso de uso consiste na criação da conta do funcionário
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator é capaz de utilizar um telemóvel
Pré-condições	O telemóvel encontra-se ligado à internet
Pós-condições	A conta fica registada e ativa para utilização posterior

Tabela 5 - Funcionalidade do ator funcionário: "criar conta".

Identificador	uc_letter_funcionario_02
Nome	Entrar numa organização
Descrição Sumário	Este caso de uso consiste no registo numa organização
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator já realizou *o log in* na aplicação; O ator conhece o nome da organização que quer entrar;
Pré-condições	O telemóvel encontra-se ligado à internet
Pós-condições	O ator fica registado na organização

Tabela 6- Funcionalidade do ator funcionário: "entrar numa organização".

Identificador	uc_letter_funcionario_03
Nome	Enviar mensagens
Descrição Sumário	Este caso de uso consiste no envio de uma mensagem instantânea.
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator já realizou *o log in* na aplicação; O ator está registado numa organização; O ator já escolheu a quem quer enviar a mensagem;
Pré-condições	O telemóvel encontra-se ligado à internet
Pós-condições	A mensagem é enviada com sucesso

Tabela 7- Funcionalidade do ator funcionário: "enviar mensagens".

Identificador	uc_letter_funcionario_04
Nome	Telefonar
Descrição Sumário	Este caso de uso consiste em fazer uma chamada de voz
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator já realizou o *log in* na aplicação; O ator está registado numa organização; O ator já escolheu a quem quer telefonar;
Pré-condições	O telemóvel encontra-se ligado à internet.
Pós-condições	O telefonema é feito com sucesso.

Tabela 8- Funcionalidade do ator funcionário: "telefonar".

Identificador	uc_letter_funcionario_05
Nome	Procurar utilizadores
Descrição Sumário	Este caso de uso consiste em procurar um utilizador
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator já realizou *o log in* na aplicação; O ator está registado numa organização;
Pré-condições	O ator não está sozinho na organização
Pós-condições	Um utilizador é encontrado e é possível ver informação sobre ele e contactá-lo.

Tabela 9 - Funcionalidade do ator funcionário: "procurar utilizadores".

Identificador	uc_letter_funcionario_06
Nome	Alterar perfil
Descrição Sumário	Este caso de uso consiste em ver e alterar perfil do utilizador
Ator	Funcionário
Prioridade	Essencial
Pressupostos	O ator já realizou *log in* na aplicação; O ator está registado numa organização;
Pré-condições	
Pós-condições	É possível visualizar a alteração feita.

Tabela 10 - Funcionalidade do ator funcionário: "alterar perfil".

4.5.2. Administradores

Este ator representa o administrador do sistema. Pretende-se que este seja um perfil de gestão de outros utilizadores, de permissões e de atribuição de tarefas. Na Figura 13 encontram-se esquematizados os casos de uso deste ator, sendo os mesmos detalhados nas tabelas 11, 12, 13 e 14.

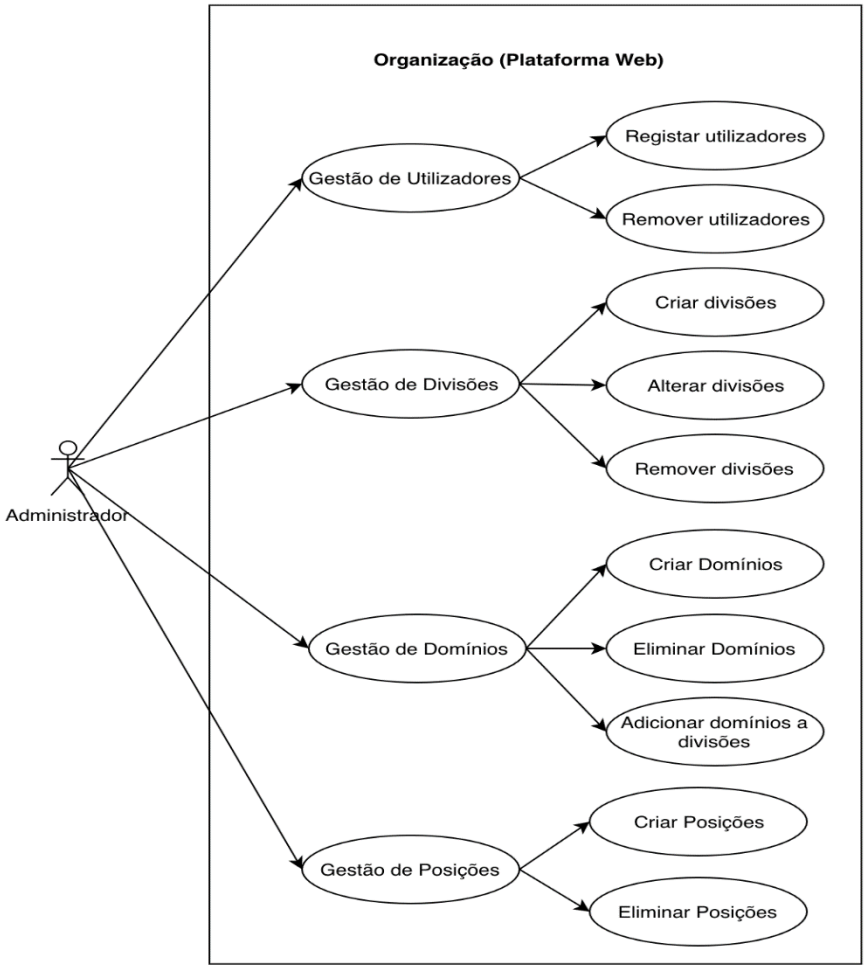

Figura 13 - Funcionalidades acessíveis ao ator "Administrador".

Identificador	**uc_letter_administrador_01**
Nome	Gestão de utilizadores
Descrição Sumário	Este caso de uso em criar utilizadores e registar utilizadores em divisões.
Ator	Administrador
Prioridade	Essencial
Pressupostos	O ator realizou *o log in* no sistema
Pré-condições	O computador encontra-se ligado à internet

Pós-condições	As alterações feitas ficam disponíveis ao administrador e funcionários.

Tabela 11 - Funcionalidade do ator Administrador: "gestão de utilizadores".

Identificador	**uc_letter_administrador_02**
Nome	Gestão de divisões
Descrição Sumário	Este caso de uso consiste em criar ou alterar informações de divisões.
Ator	Administrador
Prioridade	Essencial
Pressupostos	O ator realizou *o log in* no sistema
Pré-condições	O computador encontra-se ligado à internet
Pós-condições	As alterações feitas ficam disponíveis ao administrador e funcionários.

Tabela 12 - Funcionalidade do ator Administrador: "gestão de divisões".

Identificador	**uc_letter_administrador_03**
Nome	Gestão de domínios
Descrição Sumário	Este caso de uso consiste em adicionar domínios de email da organização a divisões.
Ator	Administrador
Prioridade	Essencial
Pressupostos	O ator realizou *o log in* no sistema
Pré-condições	O computador encontra-se ligado à internet
Pós-condições	As alterações feitas ficam disponíveis ao administrador e funcionários.

Tabela 13 - Funcionalidade do ator Administrador: "gestão de domínios".

Identificador	**uc_letter_administrador_04**

Nome	Gestão de posições
Descrição Sumário	Este caso de uso consiste em adicionar posições que existam dentro da organização.
Ator	Administrador
Prioridade	Essencial
Pressupostos	O ator realizou *o log in* no sistema
Pré-condições	O computador encontra-se ligado à internet
Pós-condições	As alterações feitas ficam disponíveis ao administrador e funcionários.

Tabela 14 - Funcionalidade do ator Administrador: "gestão de posições".

4.6. Modelo relacional

O modelo relacional de suporte ao sistema, está representado na Figura 14.

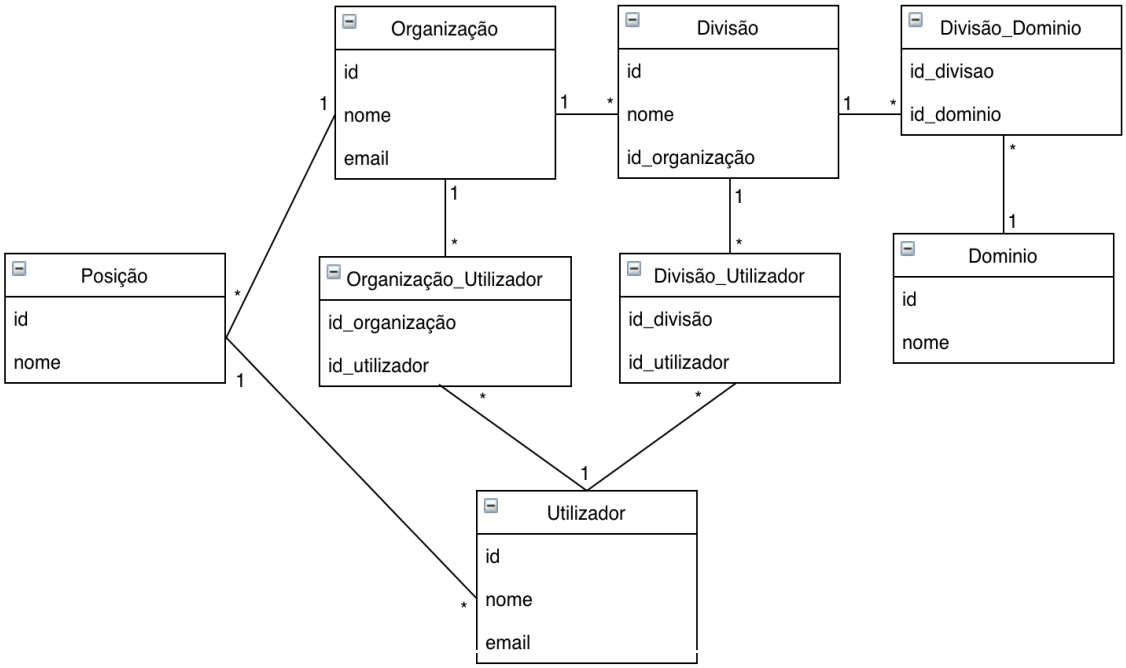

Figura 14 - Modelo relacional da aplicação Blitzchat.

Uma organização pode ter várias divisões ou departamentos (financeiro, recursos humanos, etc.). Para a presente solução, considerou-se que, para um utilizador se associar a um determinado departamento, terá que possuir um domínio

e-mail igual ao nome desse departamento dentro da organização. Considerou-se também que uma organização pode ter também várias posições (que correspondem a um cargo que o funcionário ocupa). Uma divisão/departamento pode ter vários utilizadores, sendo que um utilizador pode estar inserido em diferentes divisões/departamentos. Por fim, cada utilizador tem uma posição (cargo).

5. DESENVOLVIMENTO DA APLICAÇÃO

Nesta secção serão apresentados os desenvolvimentos realizados com base nas especificações, pressupostos e metodologias escolhidas. Será também feita a demonstração do conceito e apresentado um caso de uso prático da solução. A Figura 15 relembra a arquitetura da solução desenvolvida ao longo do presente trabalho. Os desenvolvimentos iniciaram-se com o desenvolvimento da API, bem como da lógica (core) e dos elementos de UI (user interface).

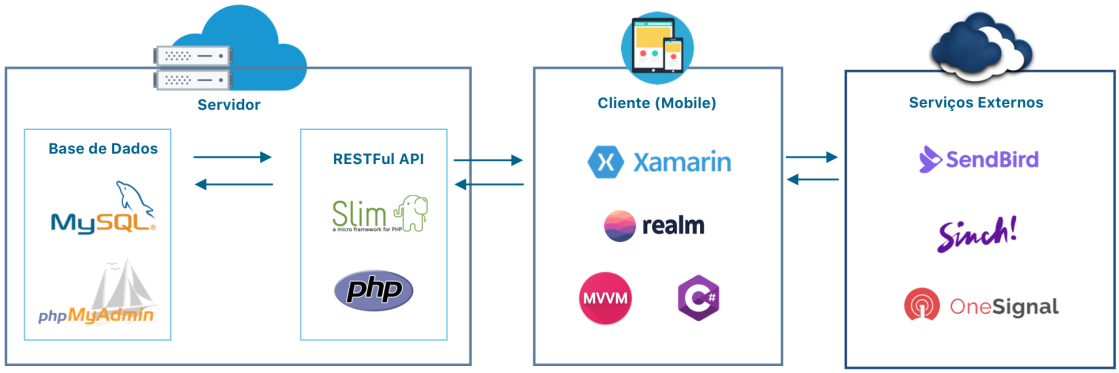

Figura 15 - Arquitetura geral da aplicação Letter Messenger

5.1. API

A aplicação Blitzchat começou a ser desenvolvida pela API. Esta apresenta uma topologia REST que funciona de forma a fazer a conexão entre a aplicação móvel e a base de dados alojada num servidor online. Após a realização e uma chamada ao serviço a partir do telemóvel, a API realiza a ligação à base de dados e retorna (em formato JSON) a informação que foi pedida. Existem inúmeras chamadas possíveis à API e maioria delas são a pedido de utilizadores na organização, nomeadamente: observar detalhes de um funcionário, o registo de um novo utilizador, guardar informação sobre definições, consultar informações sobre o utilizador, alterar a palavra-passe, entre muitos outros. A segurança destas invocações está assegurada se o utilizador tiver um *token* de segurança válido. Este *token* é gerado quando o utilizador é registado, sendo um código único de 155 caracteres que contém informações de acesso do utilizador. Quando a API recebe uma chamada, esse t*oken* é descodificado e verificado. Se for válido, a informação pedida pelo utilizador é enviada.

Caso contrário, é enviado um erro com a informação que o *token* está expirado e que é necessário renovar o mesmo. O *token* expira de 3 em 3 meses e, para renová-lo, basta que o utilizador esteja ligado à aplicação com as suas credenciais. Foram criadas mais de 40 rotas para suportar todas as funcionalidades definidas. No anexo A encontra-se a listagem completa das funcionalidades.

5.2. Lógica da Aplicação

Aquilo que designamos de *core* é a lógica da aplicação, ou seja, engloba todas as potenciais respostas e comportamentos que a aplicação pode apresentar quando recebe informação de uma API (o que ocorre quando o utilizador clica em algum botão ou entra numa nova janela). O *core* é também o responsável pela forma como é tratada a informação apresentada ao utilizador. Esta foi a segunda parte de desenvolvimento da aplicação, pertencendo já à solução móvel. Apesar disso, continua a ser um projeto distinto da camada de desenvolvimento *iOS* e *Android*. Como referido anteriormente, é realizada uma separação da camada *core* (Lógica) e da *user interface iOS/Android* para que não seja necessário replicar código. Uma vez que a lógica da aplicação é semelhante tanto para iOS ou Android, é feita uma separação para que seja possível fazer uma ligação entre o *core* e a parte *UI* (*user interface*) (*iOS/Android*). Para conseguir esta estrutura, foi utilizada a arquitetura de software MVVM previamente abordado. O *core* e as lógicas subjacentes alojadas no mesmo permitem que o utilizador possa definir critérios para utilizar algumas funcionalidades da aplicação como, por exemplo, as pesquisas de utilizadores (ex.: apresentar funcionários que têm o nome começado pela letra "J" é um exemplo de uma lógica do *core* que despoleta o apresentar de contactos que se encaixem nestes critérios dentro da camada de UI.

5.3. User Interface

O *user interface,* ou UI, é a camada que gere a informação visual que é apresentada ao utilizador através de uma ligação entre o UI e o *Core*. Para este projeto, foi desenvolvido um ambiente gráfico para *iOS*, com versão mínima v10.0 (o que significa que a aplicação está disponível a partir do *iphone 5)*. O desenvolvimento da aplicação foi dividido em 5 partes, sendo elas:

- Gestão de Utilizador
- Perfil de utilizador & Definições
- Lista de Contactos

- Chamadas e histórico de chamadas
- Lista de conversas/Janela de chat.

A implementação do *core* também seguiu esta ordem de implementação, e foi elaborada paralelamente.

5.3.1. Gestão de utilizador

Para a gestão do utilizador e associação do mesmo a uma organização, foram utilizados serviços da API como os serviços "registar utilizador", "recuperar palavra-passe", "procura de uma organização por nome ou e-mail" e "mostrar lista de divisões e posições que uma empresa tem para o utilizador selecionar". Todas estas informações estão guardadas na base de dados alocada no servidor. A Figura 16 mostra a interface de *log in* (imagem à esquerda) e a interface de registo de utilizador (imagem à direita).

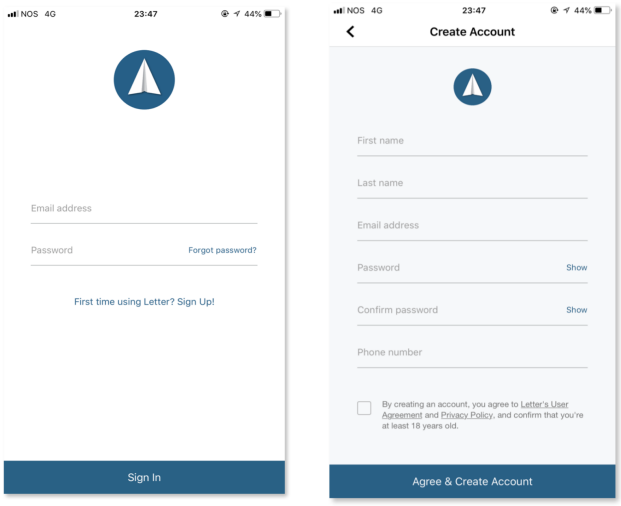

Figura 16 - Interface de log in e interface de criação de conta, respetivamente.

5.3.2. Perfil de utilizador e definições

No que diz respeito ao tratamento de dados do perfil de utilizador ficou definido que estes deveriam ser guardados tanto na base de dados do servidor como na base de dados do *smartphone*, o que torna possível o acesso à aplicação quando não houver conexão à internet, permitindo um acesso mais rápido quando o utilizador entra na aplicação. Esta condição também se verifica para os restantes pontos apresentados nesta secção. A Figura 17 mostra a interface de perfil de utilizador (imagem à esquerda) e o menu de definições (imagem à direita).

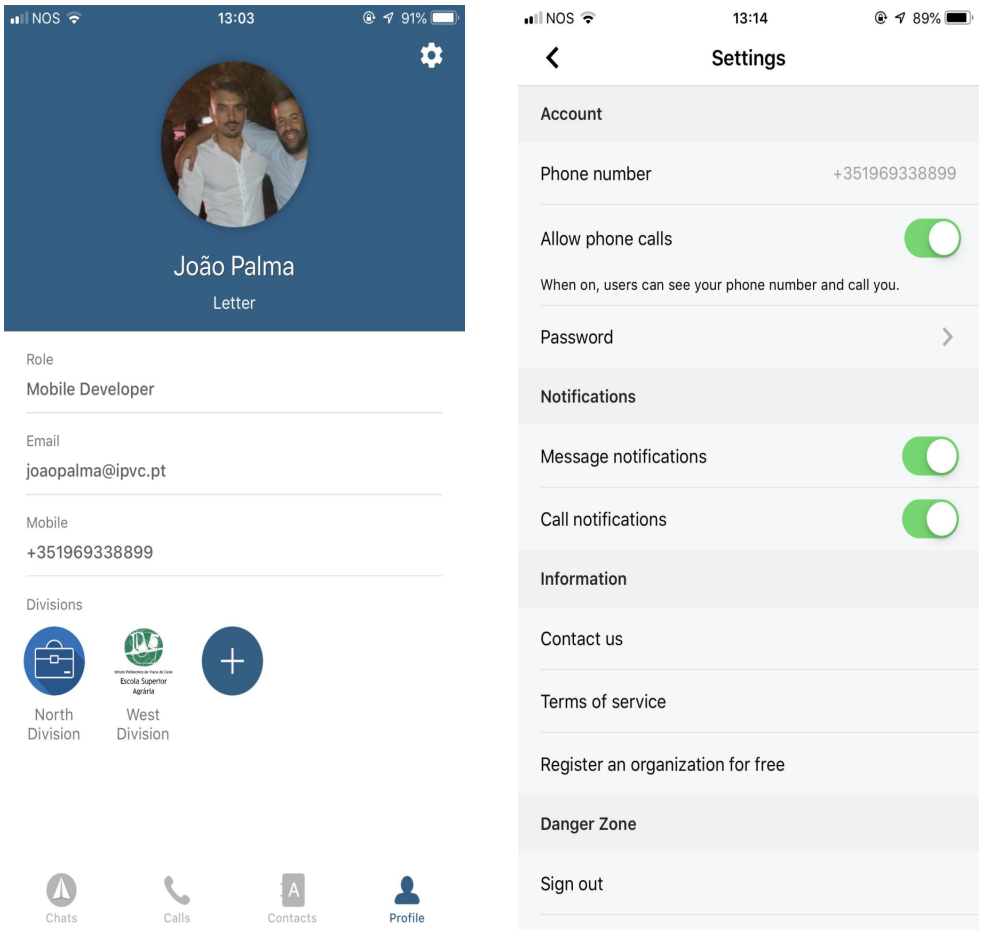

Figura 17 - Interface de perfil de utilizador e menu de definições, respetivamente.

5.3.3. Lista de contactos

Na lista de Contactos são apresentados todos os contactos de forma estratificada. Definiu-se que, por base, os contactos são estratificados segundo as divisões/departamentos que cada contacto integra dentro de uma organização. Aqui o utilizador tem a possibilidade de ver o perfil de funcionários, abrir uma conversa ou fazer uma chamada. Ainda é possível organizar os contactos por nome, ou cargo que

os funcionários que ocupam na empresa. A Figura 18 mostra as interfaces das listas de contactos da aplicação Blitzchat.

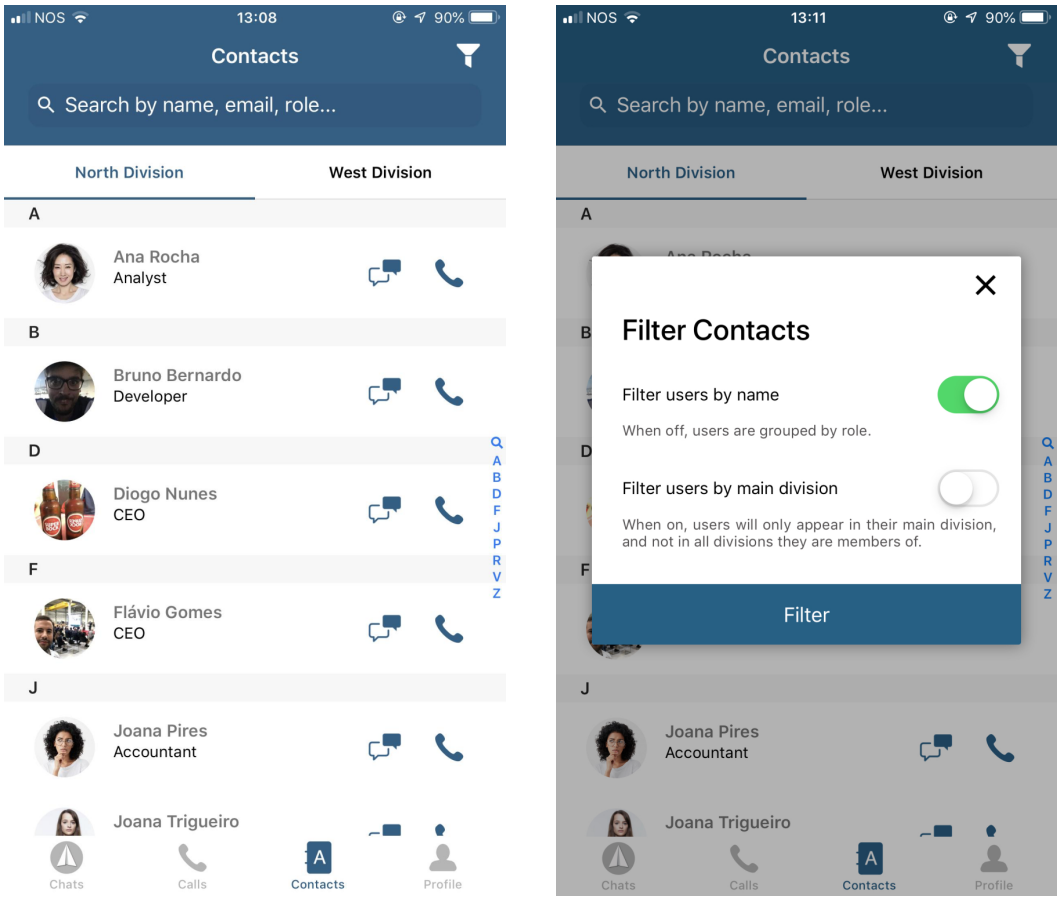

Figura 18 - Interfaces da lista de contactos.

5.3.4. Chamadas e histórico de chamadas

No que diz respeito à realização de chamadas telefónicas foram desenvolvidos dois modos distintos (chamada via aplicação Blitzchat e chamada via rede GSM). Pretendeu-se tornar possível estes dois modos de conexão para salvaguardar situações onde não existisse conexão internet. O primeiro modo utiliza o protocolo VoIP para a realização de chamadas e requer conexão à internet sendo, também, gratuito. O segundo modo de realizar chamadas recorre ao operador de rede GSM do utilizador, necessitando que os utilizadores tenham ativa a permissão de receber chamadas telefónicas.

No que diz respeito à consulta de histórico foi desenvolvido uma interface de consulta chamadas realizadas, perdidas e atendidas. A Figura 19 apresenta a interface de realização e consulta de histórico de chamadas.

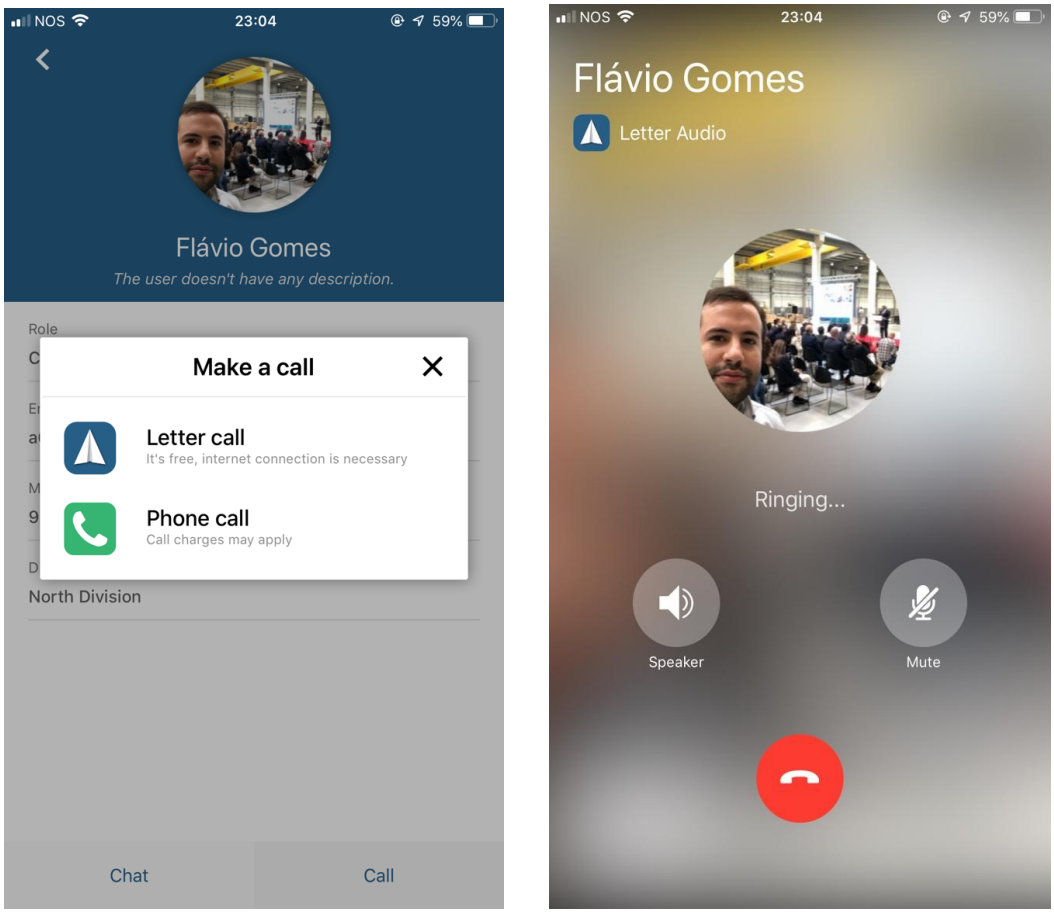

Figura 19 - Interface de realização e consulta de histórico de chamadas.

5.3.5. Janela de Chat

A janela de chat subdivide-se em 3 componentes: navegação; interação de mensagens; visualização das mensagens como exposto na Figura 20.

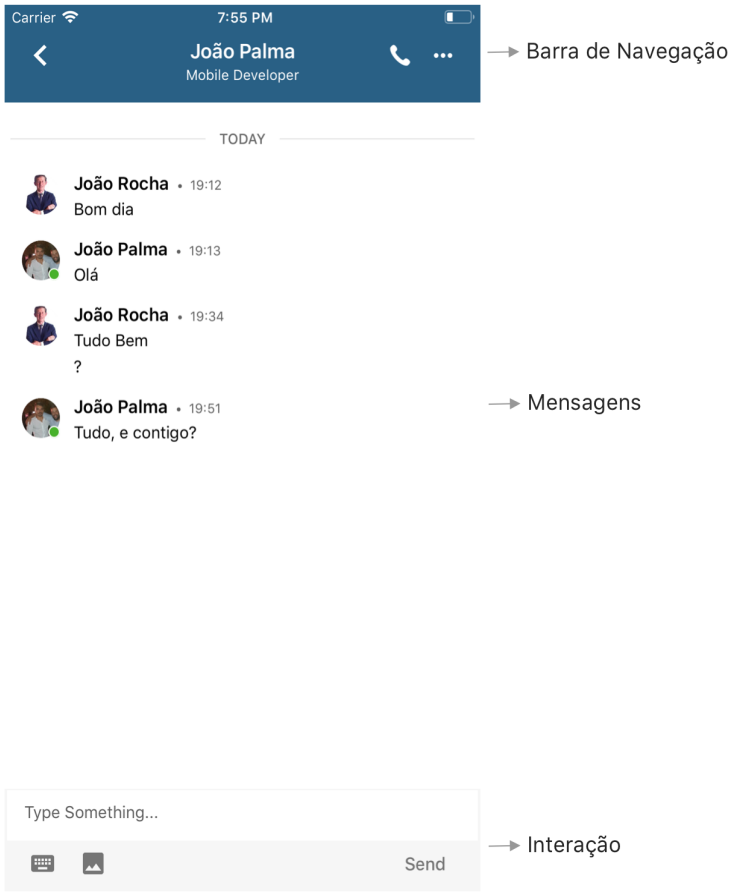

Figura 20 - Janela de chat da aplicação Blitzchat.

A barra de navegação contém um título e um subtítulo, sendo que o título corresponde ao nome do colaborador e o subtítulo ao cargo que ocupa. Esta barra pode conter ainda mais um campo que corresponde à divisão/departamento a que o utilizador pertence, estando esta variável dependente de se o colaborador está em mais do que uma divisão/departamento dentro da organização. A barra de navegação contém ainda dois botões. Um deles permite iniciar chamadas telefónicas e o outro permite aceder ao menu de definições desta janela de chat. No menu de definições o utilizador pode inspecionar informações do colaborador e realizar ações como arquivar conversa e silenciar as notificações de novas mensagens. Por fim, a barra de navegação possui ainda um botão para fechar a conversa. Outra componente de extrema importância é a interação de mensagens, pois é o meio onde se propaga a troca de

informações entre utilizadores, permitindo o envio de texto ou imagens. Existe também lógica de programação desenvolvida para o envio de mensagens, tendo-se definido que só é possível enviar mensagens caso exista algum texto escrito (ou seja, não é possível enviar uma página em branco). O mesmo tipo de lógica foi aplicada para o uso de espaçamentos, impedindo que o utilizador envie vários espaçamentos sem que estes sejam seguidos de uma palavra ou frase. Foi também desenvolvida lógica para salvaguardar o envio de mensagens e notificar o utilizador de problemas. No caso de falha de rede é apresentado ao utilizador um alerta de erro de envio de mensagem e a mensagem falhada aparece listada a vermelho. Para a resolução deste erro, basta carregar na mensagem falhada ou conectar-se a internet para a mensagem ser reenviada automaticamente. Quando se pretende escrever uma mensagem, é apresentado um teclado no ecrã que divide a janela de chat a meio, de forma a ser possível visualizar as mensagens mais recentes, mas também escrever uma nova mensagem. Para além disso, ambos os utilizadores ainda conseguem observar quando o utilizador está a escrever uma nova mensagem através de uma indicação no ecrã. Quando um utilizador entra numa janela de conversação, é possível observar as últimas 30 mensagens da conversa. Para além disso, as mensagens são guardadas de duas formas distintas, nomeadamente, na base de dados online e na base de dados local do *smartphone*. Isto permite que o utilizador possa consultar as últimas mensagens transmitidas sem que este esteja conectado à internet. Se existirem novas mensagens que estejam pendentes ou a descarregar, o utilizador é alertado através de uma animação desenvolvida para esse efeito. Após essa descarga é possível ler a mensagem. Assim que a mensagem for lida, o outro interveniente da conversa recebe um alerta de que a mensagem foi lida. No que diz respeito aos tipos de mensagem existentes foram desenvolvidas funcionalidades para suportar o envio de mensagens de texto e de mensagens de imagem. Estas são apresentadas consoante algumas regras que foram implementadas, tais como parâmetros como “quem enviou”, “quantas mensagens enviou”, “qual foi o intervalo de tempo que passou entre cada mensagem”. Consoante estes indicadores são apresentadas as mensagens com o nome do utilizador que as enviou, a sua fotografia do perfil, hora e data, o status da pessoa (*online/offline*) e há quanto tempo esteve. Quanto às notificações de mensagens, existem dois tipos de notificação: as notificações remotas e as notificações locais. As notificações remotas ocorrem sempre que é enviada uma mensagem para

um utilizador que tem a aplicação fechada. Este recebe um alerta no smartphone, indicando que existe uma nova mensagem por ler na aplicação. Além de exibir a mensagem é também adicionado um alerta em imagem com a quantidade de mensagens que tem por ler por cima do ícone da aplicação. Esse alerta é limpo quando o utilizador abre a aplicação. No que diz respeito às notificações locais, estas ocorrem quando o utilizador se encontra com a aplicação aberta, mas fora da conversa onde foi gerada a notificação. A Figura 21 demonstra o sistema de notificações desenvolvido.

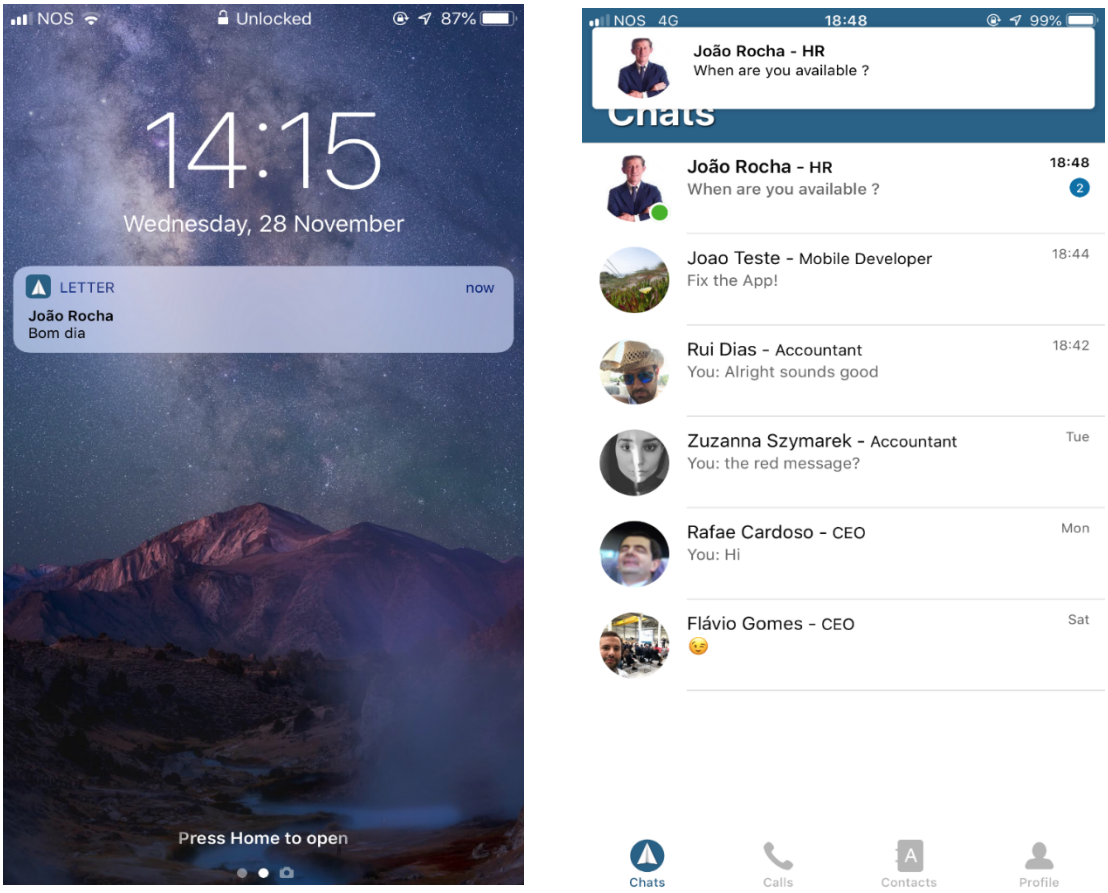

Figura 21 - Sistema de notificações.

6. RESULTADOS

Após o desenvolvimento da aplicação, foi pedido aos mesmos utilizadores do primeiro questionário para avaliarem a aplicação desenvolvida e mencionar de que forma respondia aos objetivos iniciais. Estas respostas seriam classificadas de 1 até 5, onde os valores classificam o quanto o entrevistado concorda ou discorda com a questão realizada. Foram obtidas 30 respostas. De seguida apresentam-se as questões feitas e respostas obtidas, bem como uma análise das mesmas, ilustradas nas Figuras 22, 23, 24 e 25.

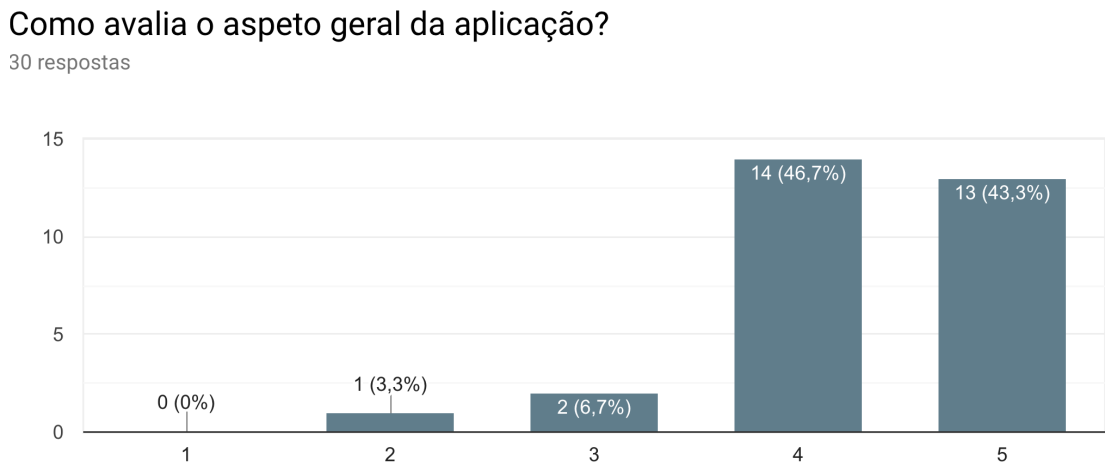

Figura 22 - Avaliação do aspeto geral da aplicação.

O primeiro aspeto sobre o qual as 30 pessoas foram inicialmente questionadas teve a ver com o aspeto da aplicação. Dentro das respostas, os valores mais obtidos focaram-se no valor 4 (com 13 respostas, correspondendo a 46,7% das mesmas) e no valor 5 (com 13 respostas, correspondendo a 43,3% das mesmas) como demonstra a Figura 22. De modo geral, o aspeto da aplicação foi bem aceite pela amostra de entrevistados.

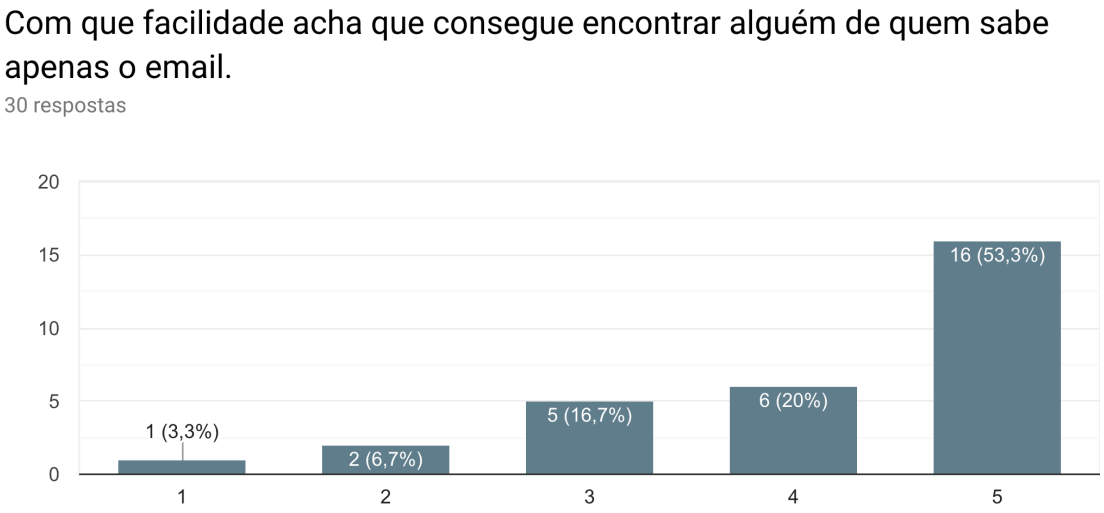

Figura 23 - Avaliação da facilidade de busca de contactos I.

No que diz respeito a aspetos relacionados com a usabilidade, questionou-se também a amostra de pessoas sobre a facilidade em encontrar um contacto sabendo apenas o seu e-mail. Cerca de 53% dos votos (16 respostas) avaliaram a facilidade com valor 5 (ou seja, muito fácil), e cerca de 20% (6 respostas) e 16% (5 respostas) avaliaram a facilidade com os valores 4 e 3, respetivamente, como demonstrado na Figura 23. Mencione-se também que apenas existiram 2 respostas com o valor 2 (cerca de 7% das respostas) e apenas uma com o valor 1 (cerca de 3% das respostas).

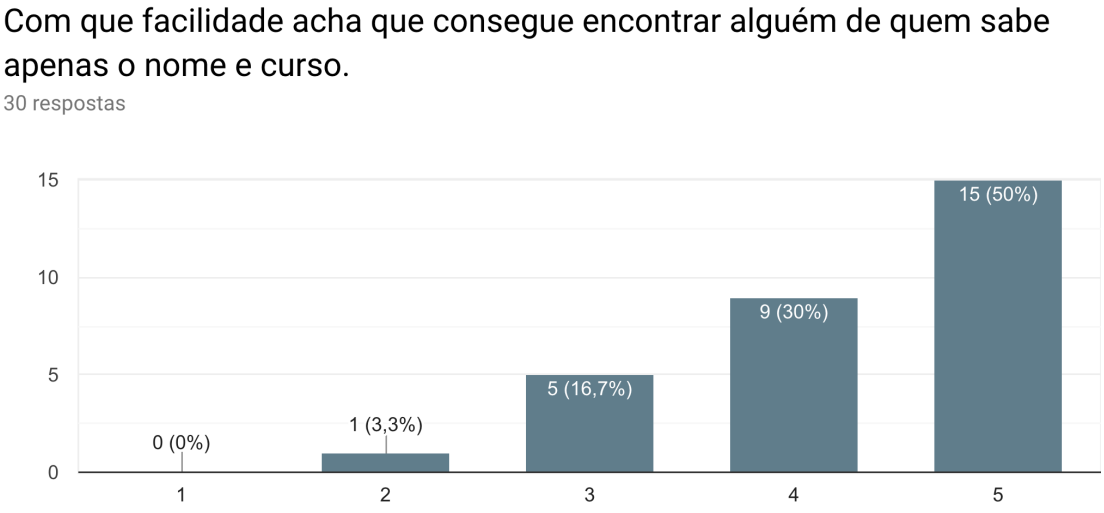

Figura 24 - Avaliação da facilidade de busca de contactos II.

No mesmo sentido, questionou-se também a amostra sobre a facilidade em encontrar outros contactos na aplicação cujo utilizador saiba apenas o nome ou curso. Cerca de 15 pessoas (50% das respostas) atribuíram o valor 5 a este mecanismo de pesquisa. Dentro das restantes, 9 pessoas (30% das respostas) atribuíram o valor 4. Por fim, saliente-se também que 5 pessoas (cerca de 17% das respostas) foram

classificadas com o valor 3, e que apenas 1 resposta foi classificada com o valor 2. As respostas para esta questão foram significativamente melhores, revelando que existe maior facilidade neste tipo de pesquisa. Além disso, é possível afirmar que o objetivo de tornar a procura de contactos simples e intuitiva foi conseguida com sucesso.

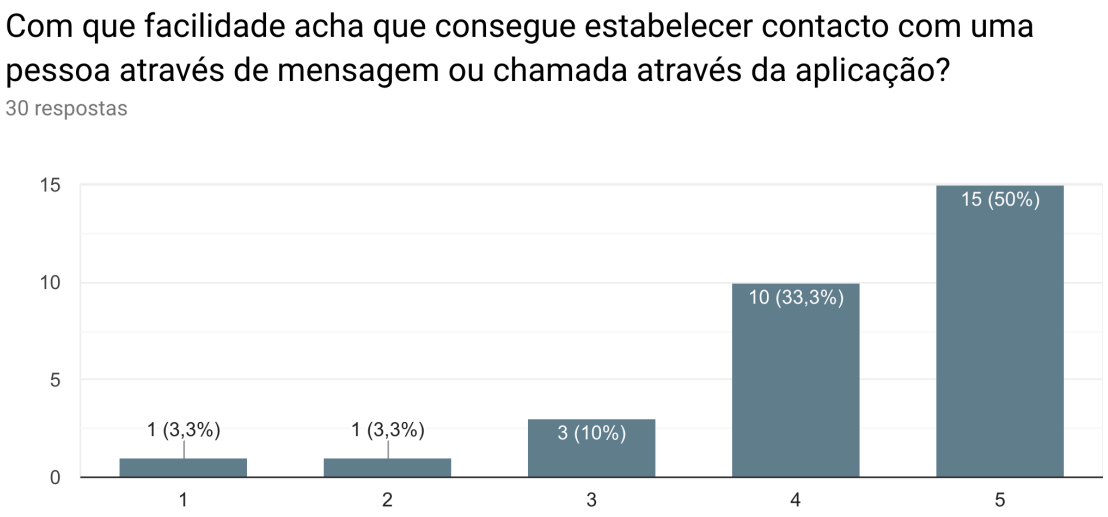

Figura 25 - Avaliação da facilidade de estabelecimento de comunicações com outros contactos.

Por fim, a amostra foi ainda questionada sobre o quão fácil é estabelecer contacto com outros utilizadores através de mensagens e chamadas dentro da aplicação Blitzchat. 50% das respostas a esta questão incidiram no valor 5, correspondendo a 15 pessoas, e 33% das mesmas sobre o valor 4, comprovando que é efetivamente fácil de realizar as ações questionadas. Apenas 3 dos entrevistados classificaram a dificuldade como mediana (valor 3) e apenas duas pessoas consideraram difícil realizar estas ações (uma resposta com valor 2 e outra com valor 1).

7. CONCLUSÕES E TRABALHO FUTURO

Este trabalho teve como principal objetivo desenvolver uma aplicação que propõe dar resposta a algumas necessidades de agilização e facilidade de comunicação dentro das instituições. O objetivo inicialmente definido era desenvolver uma aplicação móvel (*app*) que tornasse fácil e rápida a descoberta de utilizadores dentro de uma empresa sem a necessidade de ter os seus contactos nem de a adicionar nas redes sociais/privadas para comunicar.

A metodologia adotada iniciou-se com a elaboração de um questionário inicial de perceção das reais necessidades dos utilizadores, comparação com algumas ferramentas, realização da especificação de requisitos, especificação dos vários atores da plataforma, desenvolvimento do modelo relacional de suporte, a implementação propriamente dita e testes finais.

De modo a aferir a necessidade da solução foi realizado um questionário inicial com várias questões-chave que tencionavam revelar a necessidade da comunidade em adotar uma aplicação com as funcionalidades chave que eram oferecidas pela app Blitzchat. A principal conclusão foi de que grande parte dos inquiridos acharam relevante a existência de uma aplicação com este propósito dentro da Escola Superior de Tecnologia e Gestão (onde o questionário foi realizado).

Foi também efetuada a comparação com algumas ferramentas do mercado (Slack, Skyper For Business, RingCentral Glip e outras sociais como Facebook e Whatsapp) e as principais conclusões obtidas incluem o facto de nenhuma das aplicações possuir um *backoffice* - onde administradores das instituições podem configurar diversas funcionalidades para permitir uma utilização da aplicação mais completa - e também a inexistência de uma procura avançada dos utilizadores com o uso de dados como e-mail e número de telefone que não permitia criar métodos de pesquisa mais eficientes dentro das empresas (por exemplo, pesquisar utilizadores apenas sabendo elementos como o seu nome e departamento).

Posteriormente realizou-se a especificação dos requisitos da solução, onde se incluiu a visão global da mesma, os pressupostos de utilização e a metodologia de desenvolvimento que seria seguida.

O desenvolvimento da aplicação decorreu com base nas definições especificadas e iniciou-se pela API, seguindo as camadas core da aplicação e finalizando-se no desenvolvimento dos componentes da interface de utilizador e respetivos casos de uso.

Posteriormente a aplicação foi disponibilizada a uma amostra de pessoas juntamente com um questionário de avaliação da aplicação. Através das respostas obtidas foi possível aperceber-se de que a amostra questionada avaliou positivamente a solução nos aspetos chave da mesma: usabilidade, facilidade de utilização e simplicidade.

Relativamente a trabalhos futuros, existe espaço para atualizar a aplicação ao nível do *design* à medida que este sofra evoluções que são ditadas pelos padrões de usabilidade impostos por aplicações líder no mercado como os já referidos *Whatsapp, Messenger* ou *Slack.* Por fim, existe ainda espaço para idealizar novas funcionalidades que aprimorem ainda mais a experiência de comunicação entre várias entidades dentro de uma organização. Uma vez que o objetivo final seria comunicar dentro de âmbito profissional, no futuro poderia ser interessante desenvolver novas funcionalidades que promovessem a atribuição de tarefas entre contactos para execução de projetos ou mesmo a criação de *workflows* para projetos que são levados a cabo individualmente ou entre equipas de desenvolvimento.

8. REFERÊNCIAS

Adamczyk, P., Smith, P., Johnson, R., & Hafiz, M. (2011). *REST and Web Services: In Theory and in Practice.* Springer, New York, NY: Wilde E., Pautasso C. (eds) REST: From Research to Practice.

Atlassian. (2018). *HipChat*. Obtido de https://www.atlassian.com/software/hipchat/downloads

Cerdeño, E. (2013). Phone evolution and Revolution. Madrid: MAPFRE RE.

Davis, A., Guimarães, D., Arcanjo, F., & Brunoro, G. (2011). Software livre em plataformas móveis: um futuro incerto? In Anais do Congresso Nacional Universidade, EAD e Software Livre.

Facebook, Inc. (2018). *Messenger for Desktop*. Obtido de https://messengerfordesktop.com/?fbclid=IwAR0AZ6ULzQWVLKtC356ou_rEscGBrEV7Oit7FwW3138Vf2wm-FZZRYYK2DA

Gartner Inc. (2017). *Leading through digital disruption.* Gartner, Inc.

Gavalas, D., & Economou, D. (2011). *Development Platforms for Mobile Applications: Status and Trends. I.* IEEE Software, 28(1), 77–86. doi:10.1109/ms.2010.155.

Goadrich, M., & Rogers, M. (2011). *Smart smartphone development: iOS vs Android.* Proceedings of the 42nd ACM Technical Symposium on Computer Science Education - SIGCSE '11. doi:10.1145/1953163.1953330.

Gronli, M., & Younas, M. (2014). *Mobile Application Platform Heterogeneity: Android vs Windows Phone vs iOS vs Firefox OS.* IEEE 28th International Conference on Advanced Information Networking and Applications.

GSMA. (2018). *The Mobile Economy 2018.* GSM Association.

Holicza, P., & Kadëna, E. (2018). *Smart and Secure? Millennials on Mobile Devices.* Interdisciplinary Description of Complex Systems, Vol. 16, pp (376-383).

Merz, B., Tuch, A., & Opwis, K. (2016). *Perceived User Experience of Animated Transitions in Mobile User Interfaces.* Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16. doi:10.1145/2851581.2892.

Nascimento, A., & Silveira, D. (2017). *A systematic mapping study on using social media for business process improvement.* Nascimento, A. M., & Silveira, D. S. da. (2017). A systematic mapping study on using social media for business process improvement. Computers in Human Behavior, 73, 670–675.

OneSignal. (2018). *Product Overview*. Obtido de https://documentation.onesignal.com/docs

Ottka, S. (2015). *Comparison of mobile application development tools for multi-platform industrial applications.* Aalto University, Degree Programme in Computer Science and Engineering (Master's Thesis).

PHP.(2018). *PHP Manual*. Obtido de PHP: http://php.net/manual/en/intro-whatis.php

Pires, P. (2016). *Cross-Platform Development for Enterprise Solutions.* (Tese de Mestrado) Faculdade de Ciências e Tecnologias, Universidade de Lisboa.

RingCentral. (2019). *RingCentral Glip*. Obtido de Glip: https://glip.com/hp_b11

Realm. (2018). *What is Realm Platform*. Obtido de https://docs.realm.io/sync/what-is-realm-platform

Redda, Y. A. (2012). *Cross platform Mobile Applications Development.* Norwegian University of Science and Technology (Master's Thesis).

SendBird. (2018). Obtido de https://sendbird.com

SILVA, L. D., & REIS, G. (2009). *Comunicação empresarial e sua influência no cotidiano das organizações.* Revista, Uberaba, n. 6, p. 121-192, 2009.

Sinch. (2018). Obtido de https://www.sinch.com

Singh, R. (2014). *An Overview of Android Operating System and Its Security Features.* Journal of Engineering Research and Applications, Vol. 4, Issue 2(Version 1), pp.519-521.

Slack. (2018). Obtido de https://slack.com

Slim. (2018). *Slim Framework*. Obtido de https://www.slimframework.com

Skype. (2019). *Skype for Business*. Obtido de Skype for Business: https://www.skype.com/pt/business/

Statista. (2018). *Delivering business applications on the cloud: barriers in U.S. and Germany 2018*. Obtido de Statista: https://www.statista.com/statistics/875898/share-mobile-app-reinstalls-category/

Thamizharasi, R. (2016). *Android Mobile Application Build on Android studio.* International Journal of Modern Computer Science (IJMCS), Volume 4, Issue 1, February, 2016.

Tracy, K. (2012). *Mobile Application Development Experiences on Apple iOS and Android OS.* IEEE Potentials, 31(4), 30–34. doi:10.1109/mpot.2011.2182571.

Wasserman, A. I. (2010). *Software Engineering Issues for Mobile Application development.* Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research - FoSER '10. doi:10.1145/1882362.1882443.

Whatsapp. (2018). *Features*. Obtido de https://www.whatsapp.com/features/

9. ANEXO A

Este anexo contem todas as rotas de suporte às funcionalidades da aplicação. para suportar todas as funcionalidades definidas.

$app->get('/api/users/division/{id}', 'GetUsersFromDivision');

Esta rota pede o id de uma divisão, e o cliente recebe todos os utilizadores que estão nessa divisão.

$app->get('/api/users/divisions', 'GetUsersFromAllDivisions');

Esta rota o cliente recebe todas as divisão a que ele pertence

$app->get('/api/users/division/verify/{code}', 'VerifyDivisionCode');

Esta rota o cliente envia o código de verificação da divisão, se esse código estiver correto ele é automaticamente inserido nessa divisão, se não recebe um erro como o código inserido está errado.

$app->get('/api/users/division/leave/{id}', 'UserLeaveDivision');

Nesta rota o utilizador envia o id da divisão que quer abandonar.

$app->post('/api/users/forgotpassword', 'ForgotPassword');

Esta rota serve para caso o utilizador se esqueça da palavra-passe, possa receber um código no email para alterar para uma nova palavra-passe

$app->post('/api/users/login', 'UserLogin');

Para verificar o login, o utilizador envia para esta rota as suas credenciais, se estiverem corretas, é automaticamente enviado para dentro da aplicação.

$app->post('/api/registration/add', 'UserRegistration');

É registado uma nova conta de utilizador no sistema.

$app->post('/api/registration/activate', 'ActivateAccount');

Verifica se o Código de activação da conta do utilizador está certa.

$app->get('/api/registration/resendcode/{email}/{bool}', 'ResendCode');

Envia o código de ativação para o email do utilizador.

$app>get('/api/organization/verifyname/{name}', 'VerifyOrganizationName);

É verificado o nome da organização que o utilizador procura, se existir é enviado a informação sobre a organização.

$app->post('/api/organization/verifycode', 'VerifyOrganizationCode');

Verifica se o código para entrar na organização está correto.

$app->get('/api/organization/getdivisions/{id}', 'GetDivisionsOrg');

É enviado informação de todas as divisões da organização que foi pedida.

$app->get('/api/organization/getpositions/{id}', 'GetPositions');

É enviado todas as profissões que existem na organização.

$app->get('/api/organization/leaveorganization/{id}', 'LeaveOrganization');

O utilizador é removido da organização.

$app->get('/api/profiles/user/{id}', 'GetUserProfile');

Envia informações sobre um utilizador.

$app->get('/api/profiles/division/{id}', 'GetDivisionProfile');

Envia informações sobre uma divisão.

$app->get('/api/profiles/organization/{id}', 'GetOrganizationProfile');

Envia informações sobre a organização.

$app->get('/api/usercheck/user', 'UserCheck');

Quando o utilizador entra na aplicação é verificado em que organização está e a que divisões pertence.

$app->post('/api/usercheck/tokens', 'RenewAuthToken');

É renovado o token do utilizador.

$app->get('/api/users/update/position/{id}', 'UpdateUserPosition');

É atualizado a profissão do utilizador.

$app->post('/api/users/update/cellphone', 'UpdateCellphone');

É atualizado a número de telemóvel do utilizador.

$app->post('/api/users/update/description', 'UpdateDescription');

É atualizado a descrição do utilizador.

$app->post('/api/users/update/picture', 'UpdatePicture');

É atualizado a fotografia do utilizador.

$app->post('/api/users/update/password', 'ChangePassword');

É atualizado a palavra passe do utilizador.

$app->post('/api/users/deleteaccount', 'DeleteUserAccount');

A conta do utilizador é apagada.

$app->get('/api/users/registertoken/{id}', 'RegisterToken');

É registado um token do utilizador.

$app->get('/api/users/gettoken/{id}', 'GetToken');

É enviado o token de utilizador.