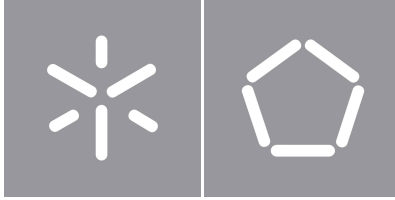


Universidade do Minho
Escola de Engenharia

André Filipe da Rocha Mendes

Consola de Atendimento para Gestão de Chamadas Telefónicas



Universidade do Minho
Escola de Engenharia

André Filipe da Rocha Mendes

Consola de Atendimento para Gestão de Chamadas Telefónicas

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho efetuado sob a orientação de
Professor Paulo Martins Carvalho
Supervisão na empresa
Célio Gomes de Abreu

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros



CC BY-NC-ND <https://creativecommons.org/licenses/by-nc-nd/4.0>

[Esta é a mais restritiva das nossas seis licenças principais, só permitindo que outros façam download dos seus trabalhos e os compartilhem desde que lhe sejam atribuídos a si os devidos créditos, mas sem que possam alterá-los de nenhuma forma ou utilizá-los para fins comerciais.]

Agradecimentos

Gostaria de expressar a minha sincera gratidão a todas as pessoas que contribuíram para a realização deste projeto de dissertação.

Esta jornada foi desafiante e recompensadora, e não teria sido possível sem o apoio, orientação e incentivo de várias pessoas e recursos. Primeiramente, quero agradecer aos meus orientadores da universidade e da empresa, Paulo Manuel Martins Carvalho e Célio Abreu respetivamente, cujas orientações se revelaram como fundamentais para o trabalho desenvolvido. A paciência, disponibilidade e esforço dedicado ao esclarecimento e explicação das minhas dúvidas e problemas, assim como os vários conselhos e críticas durante todo o processo revelaram-se como bastante valiosas e tiveram um forte impacto na qualidade desta dissertação.

Não posso deixar de mencionar toda a minha família e amigos, que sempre foram capazes de me encorajar e apoiar não só neste projeto, mas durante toda a minha jornada académica.

Gostaria também de incluir os vários colegas de turma e professores que tiveram um impacto positivo no decorrer de toda a minha formação, contribuindo para o meu crescimento como pessoa e como estudante.

À Universidade do Minho queria expressar a minha gratidão pela excelência académica e pelas inúmeras experiências positivas que foi capaz de me proporcionar.

Por fim, queria agradecer à Altice Labs pela oportunidade. Colaborar com os vários membros e equipas foi uma experiência enriquecedora, que me permitiu crescer como pessoa e como profissional, e por terem sempre os meus interesses em mente.

Mais uma vez, o meu obrigado a todos que fizeram parte desta jornada. André Mendes

Declaração de Integridade

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Universidade do Minho, Braga, novembro 2023

André Filipe da Rocha Mendes

Resumo

Esta dissertação apresenta um estudo compreensivo e extensivo de todo o processo de planeamento, desenvolvimento e implementação de um protótipo de uma consola de operador com o objetivo de auxiliar os seus utilizadores a gerir um elevado número de chamadas, de forma a ir de encontro às necessidades de várias organizações, face aos dias de hoje. Devido ao ritmo acelerado associado à evolução do software, é do interesse destas organizações melhorarem a eficiência dos locais de trabalho, algo que o protótipo desenvolvido pretende alcançar, agilizando e otimizando as várias operações de rececionistas e operadores.

Este documento inicia com um estudo aprofundado de várias soluções de consolas de operador presentes no mercado, com o intuito de analisar o que estas oferecem, incluindo a solução da Altice Labs, com o objetivo final de estabelecer uma comparação entre estas, e analisar possíveis aspetos a melhorar desta última, assim como a sua posição e papel no mercado atual.

A fase de planeamento inicia com um processo de tomada de decisão relativo às diferentes abordagens possíveis face ao trabalho a desenvolver. Após uma longa e rigorosa ponderação, que inclui uma fase de testes englobando outras soluções oferecidas por outras organizações, a abordagem relativa ao desenvolvimento de um protótipo de raiz, focado num cliente web, foi selecionada, uma vez que foi a única, na altura, viável.

Por este motivo, a fase de planeamento foi bastante extensa, envolvendo uma seleção de funcionalidades e requisitos com uma abordagem centrada ao utilizador, assegurando uma compreensão das suas necessidades e preferências, tendo sempre em mente as operações suportadas pelos serviços a integrar, que se revelaram como um tanto restritivos, passando também pelo desenho de uma interface modular e intuitiva, com foco nas várias funcionalidades planeadas.

A fase de desenvolvimento focou-se principalmente na implementação daquilo planeado na fase anterior, esclarecendo os motivos que levaram à escolha das tecnologias escolhidas e tendo sido tomado o cuidado de abordar a arquitetura e o funcionamento da aplicação, explicando os motivos que levaram às decisões tomadas. Para terminar esta fase, foi também exposto o resultado final e as várias alterações

sofridas face ao planeamento, na tentativa de melhorar o produto final.

Para terminar o documento, a eficácia da plataforma desenvolvida é avaliada através de um teste de usabilidade, com o objetivo de analisar o comportamento dos utilizadores na utilização da aplicação e coletar a opinião dos participantes, contribuindo para o levantamento de aspetos a melhorar e para tirar conclusões relativas ao produto final.

Os resultados obtidos neste teste de usabilidade foram positivos, indicando que a plataforma desenvolvida desempenhou um bom papel ao auxiliar os utilizadores a lidarem com um elevado fluxo de chamadas, aumentando a sua produtividade e eficiência, mas também servindo como fundação para trabalhos futuros na área.

Palavras-chave consola de operador, gestão de chamadas, desenvolvimento front-end, desenvolvimento web, interface do utilizador, arquitetura do cliente

Abstract

This thesis presents a comprehensive and extensive study regarding all the planning, developing and implementation process of an attendant console prototype, with the goal of aiding its users in handling a higher volume of calls, in order to meet daily workplace demands. Due to the high pace of software evolution, its in the organization's best interest to improve workplace efficiency, which is what the proposed prototype aims to achieve, enhancing and optimizing daily receptionist's and operator's tasks.

This dissertation begins with an in-depth study of existing attendant console solutions available in the market, aiming to analyze what they offer, including Altice Labs' own solution, culminating in a comparison among all the studied platforms, analyzing possible improvements and its role in the current market.

The planning phase starts with a decision-making process regarding the multiple available approaches to the problem at hand. After careful consideration, including a testing phase, where other solutions were tested, it was decided that the only viable approach referred to the development of a prototype from scratch, with focus on the web client.

For this reason, there was a long planning phase regarding the selection of all the requirements, through a user-centric approach, ensuring a broad understanding of their needs and preferences, with the integrated services supported tasks always in mind, which later revealed to be quite restrictive, including the design of an intuitive and modular user interface. The development phase was mainly implemented according to the previous planning phase, expanding on the rationale used when considering the technology utilized and explaining thoroughly all the platform's architecture and behaviour, as well as the thought process behind the decisions. To finish this chapter, the final product was also displayed, discussing the changes applied from the planning state.

The effectiveness of the developed software was also evaluated through a usability test, aiming to gather user feedback and to observe user behaviour while conducting the test, contributing for collecting various points to further improve the platform, as well as to draw conclusions regarding the final product.

The obtained results from the usability test were mainly positive, indicating the developed platform was able to successfully fulfill its role in helping its users in handling a higher influx of phone calls, increasing

their productivity and efficiency, but also in becoming a foundation for future work and improvement in the area.

Keywords attendant console, call handling, front-end development, web development, user interface, client architecture

Conteúdo

1	Introdução	1
1.1	Enquadramento e Motivação	1
1.2	Objetivos	2
1.3	Estrutura da dissertação	3
2	Estado da arte	4
2.1	Definição	4
2.2	A consola da Altice Labs	6
2.2.1	Funcionamento	7
2.2.2	Funcionalidades	8
2.2.3	Problemas	11
2.3	A concorrência	11
2.3.1	Vodafone	12
2.3.2	8x8 e Ring Central	15
2.3.3	Landis Technologies	20
2.3.4	Comparação	23
2.4	Sumário	25
3	Planeamento	26
3.1	Âmbito do Projeto	26
3.1.1	Possíveis Integrações	27
3.1.2	Integração com Bitrix24	27
3.1.3	Integração com Voice Operator Panel	32
3.1.4	Conclusões	35
3.1.5	Abordagem escolhida	36
3.2	Planeamento e Decisões	36

3.2.1	Requisitos	38
3.2.2	Tecnologias a usar	39
3.3	Arquitetura e Funcionamento	42
3.3.1	Autenticação	43
3.3.2	Chamadas	45
3.3.3	Contactos	50
3.4	Desenho da Interface do Utilizador	52
3.4.1	Página de Autenticação	53
3.4.2	Overlay da Aplicação	54
3.4.3	Área de Help Online	62
3.5	Conclusão	64
3.6	Sumário	64
4	Implementação	66
4.1	Considerações preliminares	66
4.2	Arquitetura do Front-end	67
4.3	Interface	77
4.4	Sumário	93
5	Testes de usabilidade	94
5.1	Contexto	94
5.2	Tarefas	95
5.3	Observações	97
5.4	Questionário	99
5.5	Sumário	103
6	Conclusões e trabalho futuro	104
6.1	Conclusões	104
6.2	Perspetiva de trabalho futuro	105

Lista de Figuras

1	Interface da consola da Altice Labs	10
2	Interface da Consola de Operador da Vodafone	14
3	Interface de uma aplicação da plataforma 8x8 [1]	19
4	Interface de uma aplicação da plataforma RingCentral	19
5	Integração Ring Central com Teams	20
6	Interface da Landis Attendant Console for Microsoft Teams [2]	22
7	Imagem contendo algumas integrações (obtida em novembro de 2022)	29
8	Imagem contendo a chamada enviada pela REST API pela Bitrix para o PBX integrado .	30
9	Formulário apresentado ao utilizador na Bitrix24	31
10	Interface da consola <i>Voice Operator Panel</i> , depois de instalada antes do registo no IP PBX	33
11	Chamadas efetuadas com sucesso na consola Voice Operator Panel	34
12	Erros encontrados em certas funcionalidades na consola <i>Voice Operator Panel</i>	35
13	Taxa de utilização das <i>frameworks</i> Javascript	40
14	Taxa de retenção das <i>frameworks</i> Javascript	41
15	Arquitetura de alto nível da Consola de Operador	43
16	Diagrama de fluxo associado à Autenticação	44
17	Diagrama de fluxo associado a iniciar chamada	46
18	Diagrama de fluxo associado a Chamada Atendida	48
19	Diagrama de fluxo associado a Chamada Recebida	49
20	Diagrama de fluxo associado a terminar chamada	50
21	Diagrama de fluxo associado a uma operação nos contactos	51
22	Mockup da Página de Autenticação	53
23	Mockup geral associada à disposição dos menus	54
24	Mockup associada a área de Chamadas a Tocar	56

25	Mockup associada a área de Chamadas a Tocar com o Histórico selecionado	57
26	Mockup associada a área de Chamadas em Progresso	58
27	Mockup associada ao teclado de marcação	59
28	Mockup associada à chamada em conversação no estado A Chamar	59
29	Mockup associada à área de Contactos	60
30	Mockup associada à área de Contactos, com os vários números associados a um contacto	61
31	Mockup associada à área de Contactos com contacto expandido	62
32	Mockup associada ao <i>Help</i> Online	63
33	Imagem utilizada para descrever <i>prop drilling</i> , retirada da documentação oficial do React [3]	68
34	Imagem utilizada para descrever React Context, retirada da documentação oficial do React [3]	69
35	Funcionamento da gestão de estado com <i>reducer</i> e <i>context</i> [4]	70
36	Camada aplicacional do protótipo	71
37	Arquitetura de alto nível da aplicação <i>frontend</i>	72
38	Lógica associada ao tratamento das notificações	75
39	Parte superior da árvore de componentes	78
40	Árvore de componentes relativa à Página de Autenticação (<i>Login Page</i>)	78
41	Árvore de componentes relativa à Página da consola (<i>Main Page</i>)	79
42	Imagem da interface da <i>Main Page</i>	79
43	Imagem da interface da zona do perfil do utilizador	80
44	Árvore de componentes relativa ao componente <i>Ringling Call Tab</i>	80
45	Imagem da interface relativa à área de chamadas a tocar	81
46	Imagem da interface relativa à área do histórico de chamadas	81
47	Imagem da interface relativa à área de chamadas pausadas	83
48	Árvore de componentes relativa ao componente <i>Conversation Call Tab</i>	83
49	Resultado final da implementação do teclado	84
50	Resultado final da implementação da área de chamada em conversação	84
51	Árvore de componentes relativa ao componente <i>Contacts Tab</i>	85
52	Imagem da interface relativa à área dos contactos	86
53	Imagem da interface relativa à área dos contactos monitorados	87
54	Imagem da interface relativa à área dos contactos com os filtros expandidos	88

55	Imagem da interface relativa ao formulário de edição e criação de contactos	89
56	Árvore de componentes relativa ao <i>Help Online</i>	89
57	Imagem da interface relativa ao <i>Help Online</i>	90
58	Imagem da interface relativa à aplicação num dispositivo móvel	91
59	Imagem da interface relativa à aplicação num dispositivo móvel com contactos expandidos	92
60	Respostas obtidas as perguntas de escolha múltipla	101

Lista de Tabelas

1	Comparação entre as plataformas estudadas	23
2	Lista de requisitos ordenados pela priorização Moscow	38
3	Observações retiradas dos testes efetuados	98

Acrónimos

ABC Advanced Business Communication.

API Application Programming Interface.

CRM Customer Relationship Management.

CTI Computer Telephony Integration.

DOM Document Object Model.

FAQ Frequently Asked Questions.

HD High Definition.

IT Information Technology.

JWT JSON Web Token.

PBX Private Branch Exchange.

SaaS Software as a Service.

SIP Session Initiation Protocol.

SMS Short Message Service.

SPA Single Page Application.

UC Unified Communications.

UI User Interface.

URL Uniform Resource Locator.

UX User Experience.

Capítulo 1

Introdução

1.1 Enquadramento e Motivação

A realidade do mundo das comunicações de hoje em dia é pautada pelo constante aparecimento de novos serviços e tecnologias, cada vez mais complexas e com mudanças mais aceleradas, levando a que nunca como antes fosse tão importante e necessária uma visão e gestão centralizada, rápida, eficaz e inteligente da comunicação entre as empresas e os seus clientes.

De acordo com um relatório publicado pela Grammarly [5], 76% de líderes de negócios afirmam que as suas equipas tem enfrentado dificuldades no que diz respeito em manter os processos de comunicação eficazes, principalmente com modelos de trabalho remotos ou híbridos. Estima-se que cada funcionário utilize 7.47 horas semanais na resolução de problemas resultantes de falta de comunicação ou comunicação deficiente / não eficaz. Estas horas equivalem a um dia de trabalho inteiro por semana ou, tendo em conta médias de negócios americanos, 12 506 dólares por ano.

É importante também notar o aumento na necessidade de melhorar os processos de comunicação nos últimos anos. A pandemia que deu início em 2020 levou a que várias medidas de prevenção fossem implementadas. Por este motivo, registou-se um número mais elevado de comunicação à distância, nomeadamente por telefone, para certas organizações, como o Serviço Nacional de Saúde, que lidava com chamadas relativas a este tema e prestava ajuda às pessoas. Este aumento no volume de chamadas deu origem à necessidade de as gerir, o que causou que organizações com esta necessidade procurassem software de auxílio nestas operações [6].

Efetivamente, é óbvio que processos de comunicação eficazes entre equipas e clientes são cruciais. O aumento dos recursos humanos e dos clientes associados às empresas, assim como a constante tentativa destas em crescerem como negócio e aumentar lucros são fatores que levam a que estas procurem elevar os processos de comunicação e à tentativa de acompanhar o surgimento de novos serviços e tecnologias perto, de modo a melhorar significativamente esses processos, o que resulta numa maior produtividade

e uma melhor experiência quer dos funcionários quer dos clientes finais.

Esta dissertação surge na oportunidade de trabalho académico na Altice Labs que, pelos motivos estipulados nos parágrafos acima, viu o seu produto de consola de operador, anteriormente caído no esquecimento, novamente a ser utilizado por diferentes clientes. No entanto, por ser um produto que, até à data, não tinha sido utilizado frequentemente, trata-se de software que se encontra desatualizado e com vários problemas, sendo este produto alvo de melhorias.

1.2 Objetivos

O principal objetivo deste trabalho de dissertação é o desenvolvimento de um protótipo de uma aplicação web de consola de atendimento, focada nas funcionalidades essenciais de uma rececionista ou operadora moderna.

De acordo com isto, é imperativo que a solução ofereça ferramentas para gerir eficazmente um grande volume de chamadas recebidas, bem como encaminhá-las para os destinos mais apropriados. Além disso, deve proporcionar funcionalidades adicionais, como histórico e registos, que serão definidas em fases posteriores.

É importante salientar que a Altice Labs já possui um produto de consola de atendimento/operador em produção. No entanto, devido a ser um produto com diversos problemas de implementação e documentação associados, o objetivo é que o protótipo e o estudo desenvolvido neste trabalho sirvam como base para a sua substituição.

Para alcançar este objetivo, é pretendido que se realizem várias etapas, nomeadamente:

- pesquisa relativa ao estado de arte, incluindo do produto da Altice Labs em produção, de forma a traçar metas específicas e realistas para o âmbito da implementação do software, contribuindo também para a definição de arquitetura e das interfaces;
- fase de planeamento de forma a definir com critério os requisitos, tecnologias, arquitetura e interface gráfica;
- fase de desenvolvimento com objetivo de implementar o protótipo planeado na fase anterior;
- testes de usabilidade com intuito de identificar eventuais falhas ou defeitos da solução implementada.

Além de todas estas etapas, com o intuito de resultarem num protótipo funcional de uma consola de operador, é também pretendido que todo o processo seja documentado internamente.

1.3 Estrutura da dissertação

Neste Capítulo 1, foi abordado o contexto em que o trabalho se insere, assim como foi exposta a motivação e os objetivos a atingir durante todo o processo de desenvolvimento do trabalho e da dissertação.

No Capítulo 2, é explicada a abordagem tomada para o desenvolvimento do estado de arte. Primeiramente é abordado o conceito de consola de atendimento, sendo explicada a diferença entre este conceito e outros semelhantes, seguido do estudo realizado acerca da plataforma da Altice e de outras plataformas da concorrência, onde são explicados os conceitos base relevantes para o projeto em questão. Este estudo termina com uma comparação entre estas.

No capítulo 3 está contido todo o processo de desenvolvimento, nomeadamente planeamento da abordagem a ser utilizada, definição de arquiteturas e *mockups*, um breve estudo acerca de quais tecnologias são as mais indicadas para o projeto em questão e a sua escolha, sendo que todo o processo de tomada de decisão é explicado em detalhe, indicando também qual o produto final obtido.

Relativamente ao capítulo 4, este reflete os resultados obtidos, assim como indica os diferentes testes a que foram submetidos, e se no geral, estes vão de encontro aos objetivos definidos do capítulo 1.

Para terminar, o capítulo 5 diz respeito à conclusão desta dissertação e do trabalho desenvolvido, sendo feita uma retrospectiva acerca deste, incluindo também uma secção relativa ao trabalho futuro.

Capítulo 2

Estado da arte

Neste capítulo são abordados e explicados diferentes conceitos que estão relacionados com software de consola de operador, incluindo a consola já existente, as suas funcionalidades e o seu modo de funcionamento. Além disso, é incluído também o estudo realizado relativamente a outras consolas de operador e software idêntico, que serviu de base de comparação com a consola em produção para identificar falhas nesta e também para sustentar o planeamento, auxiliando na definição de arquiteturas/interfaces, ajudando a traçar objetivos realistas. No final, é estabelecida uma comparação entre o produto da Altice Labs e a concorrência.

2.1 Definição

Uma consola de operador é uma solução de software que fornece aos seus utilizadores ferramentas para que possam fazer uma gestão de um elevado número de chamadas. Geralmente, são utilizados por rececionistas ou operadores que pretendam lidar com um elevado volume de chamadas, permitindo que estes atendam e reencaminhem todas as chamadas de acordo com as necessidades [7].

É habitual existir alguma confusão de conceitos quando se trata de software nesta área. Geralmente, apesar de ser possível fazer mais divisões em diferentes categorias, software nesta área pertence a [8] :

- **Call Center Software** - Um centro de chamadas tem um principal objetivo relativamente semelhante ao de serviços de atendimento. Trata-se de um departamento que lida com chamadas de clientes. Estes focam-se principalmente num único canal de comunicação, usando as chamadas como o único método de comunicação. De acordo com o seu foco, estes *Call Centers* podem ser *inbound* - focam-se em chamadas que são feitas do cliente para o *Call Center*, geralmente relacionadas com suporte, apoio técnico, alterações contratuais, entre outros, ou então *outbound* - que lida com chamadas do *Contact Center* para os clientes, como o objetivo de realizar inquéritos,

eventos promocionais ou vendas, sendo que um centro de chamadas pode ter como objetivo estas duas vertentes, tornando-se uma versão misturada [9];

Como tal, e de acordo com o seu foco, estes departamentos têm diferentes necessidades, como uma distribuição de chamadas, muitas vezes referido como *skill-based routing*, de modo a tornar este processo mais efetivo, ou *auto-attendants*, que automatizam estes métodos. De forma semelhante, estas soluções também usufruem de *dialers* automáticos, que automatiza também os processos de chamadas *outbound* para aumentar a produtividade dos operadores [10];

- **Contact Center Software** - Um centro de contactos pode ser visto como uma extensão de um *Call Center*, na medida em que expande as suas funcionalidades para mais canais de comunicação, nomeadamente *e-mail*, *fax*, *Short Message Service (SMS)*, cartas, redes sociais, mensagens instantâneas, entre outros. Devido a isto, todas as funcionalidades de um centro de contactos são cruciais, sendo incrementadas outras que permitam a utilização com todos os meios de comunicação mencionados [11];

A principal diferença de um *Contact Center* com serviços de atendimento está relacionada com o tratamento das chamadas em si, uma vez que nestes cenários, a intenção é que sejam os operadores que recebem ou fazem as chamadas a tratar destas, ao invés de fazer a sua transferência ou reencaminhamento [12];

- **Attendant Software** - Uma consola de atendimento tem como objetivo principal receber, transmitir e gerir um elevado volume de chamadas, sendo que estas, frequentemente, serão reencaminhadas para as linhas necessárias. Como foi dito previamente, a diferença deste software, para um de *Contact Center* está relacionada com a forma que estas chamadas são tratadas. Neste cenário, ao invés de estas serem imediatamente tratadas, é frequente serem transferidas para outros operadores de acordo com as necessidades destas [7].

Desta forma, o funcionamento normal de um rececionista que utiliza uma consola de operador pode ser resumida nos seguintes passos [13]:

- o operador atende a chamada do utilizador;
- o utilizador indica qual o motivo da chamada;
- o operador identifica a pessoa ou departamento mais adequado para lidar com o problema do utilizador;

- o operador verifica o estado da pessoa ou do departamento (se está disponível ou não) para o qual pretende transferir a chamada, e caso esteja disponível, transfere, ou então coloca o utilizador em espera.

Ou seja, uma solução de consola de operador deve ser capaz de fornecer as ferramentas essenciais para as atividades de uma rececionista ou operadora, estabelecidas acima. Como mais à frente será abordado, é comum que estas sejam parte de um sistema *Unified Communications (UC)* mais complexo, ou que usufruam de outras funcionalidades, como um serviço de gestão de filas de espera, capaz de distribuir automaticamente as chamadas pelos operadores, ou um serviço capaz de atender a chamada pelo utilizador, de forma a agilizar e automatizar este processo na tentativa de aumentar a sua efetividade.

De modo a construir um estado de arte completo, foram estudadas diferentes soluções de diferentes empresas que possuem produtos que encaixem na definição estabelecida de consola de operador, tendo como base a documentação fornecida online, quer seja através de documentos ou vídeos de demonstração para obter um conhecimento geral do seu funcionamento. Além disso, com o objetivo de aprofundar este estudo, foi explorada a possibilidade de testar uma demonstração gratuita dos produtos, no entanto, devido a múltiplas complicações, isto nem sempre se revelou possível.

É importante notar que os conceitos anteriormente explicados não tem fronteiras explícitas. Certo software estudado, apesar de pertencer a uma categoria, possui funcionalidades que vão de encontro a outras, sendo que o observado em produtos mais complexos era uma solução que englobava as três categorias de acordo com o plano de compra selecionado [14].

Além disso, foi também estudada em detalhe a consola já existente na Altice Labs, de modo a compreender em detalhe o seu funcionamento para que, futuramente, fosse possível traçar objetivos realistas. Este estudo permite também a comparação com as outras estudadas para identificar áreas em que esta pudesse ser melhorada, de forma a que a nova solução conseguisse proporcionar uma melhoria significativa, tendo em conta o âmbito desta dissertação.

2.2 A consola da Altice Labs

Esta consola trata-se de uma aplicação web, desenvolvida pela Altice Labs que, no momento desta escrita, se encontra em produção e, por isso, é utilizada diariamente por vários clientes. Tal como foi indicado previamente, é um produto que apesar de ter estado na fase final do seu ciclo de vida, recentemente voltou a ser bastante utilizado, face às necessidades enfrentadas nos últimos anos. No entanto, devido ao facto de não ser um produto muito utilizado antes da pandemia, o mesmo encontra-se desatualizado, assim

como a documentação associada que é praticamente inexistente e incompleta, e portanto pretende-se que seja melhorado. Devido à falta de documentação, e por ter sido um software desenvolvido há bastante tempo, este estudo foi bastante importante uma vez que os responsáveis que inicialmente conceberam, analisaram e desenvolveram este produto já não fazem parte da equipa há algum tempo. Como a documentação relativa a este produto é escassa, a transferência de conhecimento entre os colaboradores não foi efetuada corretamente, levando a que parte da equipa da Altice Labs envolvida neste processo não possuísse um conhecimento aprofundado acerca do produto, pelo que a aquisição e a passagem deste conhecimento para estes membros foi de enorme importância. Efetivamente, este produto foi a última consola testada, de todas as estudadas, devido a complicações existentes tanto na criação de contas a utilizar, como na estabilidade e preparação do ambiente de desenvolvimento. Apesar disso, foi possível inicialmente fazer um levantamento de funcionalidades desta consola através de diferente documentação disponível online e internamente, que foram posteriormente verificadas e aprofundadas quando esta foi finalmente utilizada.

2.2.1 Funcionamento

A primeira impressão do utilizador quando utiliza este software é bastante negativa. Comparativamente à competição estudada, esta consola é a única que não possui um *softphone* integrado e que portanto necessita de utilizar um externo.

O que é um softphone?

Geralmente, um *softphone* é uma aplicação de software que permite simular o comportamento e funcionalidades de um telefone físico tradicional, através da Internet, permitindo que o utilizador faça chamadas telefónicas sem a necessidade de hardware adicional (como um telefone fixo ou um telemóvel), uma vez que este software pode ser instalado em qualquer dispositivo [15].

Isto obviamente tem várias consequências e impactos negativos na utilização deste software. Devido a isto, a consola possui três estados possíveis:

- Online - Estado da consola que está associado à sua utilização normal, com esta totalmente operacional;
- Terminal - Estado da consola associado a um modo não totalmente operacional, com a limitação de, apesar de apresentar ao utilizador a lista completa de chamadas em que este se encontra, não lhe é permitido realizar operações nestas chamadas;

- Offline - Estado inicial da consola, onde é apenas possível visualizar a lista de contactos.

O estado da consola é obtido através da ligação desta com o *softphone*. Como, para a utilizar, é necessário conectar a um *softphone* externo, é necessário efetuar a autenticação neste sistema com as credenciais da conta do servidor *Session Initiation Protocol (SIP)* do utilizador autenticado na consola.

Assim que esta autenticação for efetuada com sucesso, o acesso à consola inicia com esta no modo *offline*, acionando uma chamada a tocar no *softphone* que é responsável por manter os dois conectados. Assim que esta chamada é estabelecida, a consola passa para o modo online, e assim permanece até que esta termine, ficando totalmente operacional. Neste modo de operação, todas as chamadas destinadas para o utilizador são direcionadas para a consola, servindo o *softphone* apenas para manter a conexão estabelecida. Desta forma, o utilizador apenas necessita de o utilizar uma vez para estabelecer a ligação. Assim, é argumentável indicar que este modo de funcionamento seja um transtorno para os utilizadores.

No entanto, os problemas surgem quando a chamada da ligação é desligada. Em cenários de término da chamada, assim como em cenários em que a chamada de ligação no *softphone* não é atendida, a consola fica em modo terminal que restringe, de um certo modo, as funcionalidade desta. Neste modo, o comportamento registado nos testes foi incoerente, mas era possível ver todas as chamadas a tocar no *softphone*, sendo apenas possível rejeitar essas chamadas. Caso o utilizador pretendesse atender qualquer chamada, teria que recorrer ao *softphone* para o fazer, sendo que a consola passa para modo online apenas enquanto essa mesma chamada se encontrar estabelecida. Ou seja, neste cenário, o terminal funciona como um terminal padrão não associado a uma consola. Apesar disso, o maior problema surge na passagem do modo online para o modo terminal.

Nestas ocorrências, notadas quando a chamada de conexão, por algum motivo, era terminada, o comportamento observado é que todas as chamadas que se encontravam pausadas pelo utilizador eram entregues sequencialmente ao terminal, permitindo ao utilizador recuperar essas chamadas. No entanto, qualquer chamada em conversação ou em toque são perdidas, não podendo ser recuperadas, o que está longe do ideal.

Por este motivo, é possível concluir que a necessidade da utilização de uma ferramenta externa que sirva como *softphone* causa algum transtorno ao utilizador, podendo este perder progresso na eventualidade de algum erro acontecer com a chamada de conexão entre a consola e a ferramenta.

2.2.2 Funcionalidades

A consola da Altice Labs apresenta as seguintes funcionalidades :

- Gestão de chamadas – É possível gerir um elevado número de chamadas simultaneamente, sendo

possível efetuar as operações básicas como atender, rejeitar e colocar em pausa. Ao utilizador é apresentada a lista de chamadas por atender, a chamada em conversação e as chamadas em espera;

- Reencaminhamento – Reencaminhamento de uma chamada quando esta ainda está a tocar, sendo possível inserir o número para o qual reencaminhar ou então escolher dos contactos, transferindo a chamada para esse utilizador. É ainda dada a hipótese de configurar um reencaminhamento incondicional, fazendo com que as chamadas direcionadas para a linha do operador sejam automaticamente reencaminhadas ou para o *voicemail* ou então para um número introduzido;
- *Do Not Disturb* – Serviço que o utilizador pode configurar de modo a rejeitar todas as chamadas direcionadas para a linha automaticamente;
- Transferir Chamadas - É possível transferir a chamada quando esta já se encontra a decorrer. Esta pode ser feita de duas maneiras - sem consulta, sendo apenas introduzido o número do utilizador para o qual se pretende transferir ou seleccionando da lista, ou então com consulta, o que inclui colocar a chamada que se pretende transferir em pausa, ligar para o número para o qual se pretende transferir para fazer a consulta, e só depois transferir a chamada. Em qualquer destas duas situações, enquanto a transferência está a decorrer, a chamada fica colocada na zona de chamadas em espera, e aí permanece caso o utilizador rejeite a transferência, podendo posteriormente ser transferida novamente ou continuada;
- Diretório de Contactos – O diretório de contactos possui contactos pessoais, que o utilizador tem total liberdade de criar, editar e eliminar e contactos corporativos, aos quais o utilizador pode apenas adicionar notas. É possível também pesquisar e filtrar todos os contactos. Existem ainda os contactos monitorados, que consistem numa lista limitada de contactos internos ao cliente com estado de presença com a hipótese de capturar chamadas;
- Estado de presença (*Busy Lamp Field*) e Capturar Chamadas – É possível ver o status de presença de todos os contactos monitorados :
 - Livre - o utilizador não está em chamada nem tem chamadas a tocar;
 - A tocar - o utilizador tem pelo menos uma chamada a tocar, mas não está em chamada;
 - Ocupado - o utilizador está em chamada.

Caso o utilizador esteja no estado A tocar, é possível capturar a chamada, ou seja, “roubar” a chamada que ainda não foi atendida. Por motivos de privacidade, não é possível ver o número de origem das chamadas a capturar nem escolher qual chamada capturar caso o utilizador tenha múltiplas chamadas a tocar;

- Histórico de chamadas – É possível visualizar o histórico de todas as chamadas que ocorreram durante a sessão ativa na consola. Assim, caso a sessão seja encerrada, o histórico é limpo. No entanto, a zona dedicada a este histórico possui uma opção que redireciona o utilizador para a ferramenta de administração onde é possível analisar o histórico completo de chamadas;
- Toque simultâneo – É possível configurar, através de grupos de atendimento, uma entrega paralela de chamadas em várias consolas, podendo todos os membros do grupo visualizar e as chamadas direcionadas para a linha desse mesmo grupo. No entanto, apenas um utilizador poderá atender à chamada.

Interface Gráfica

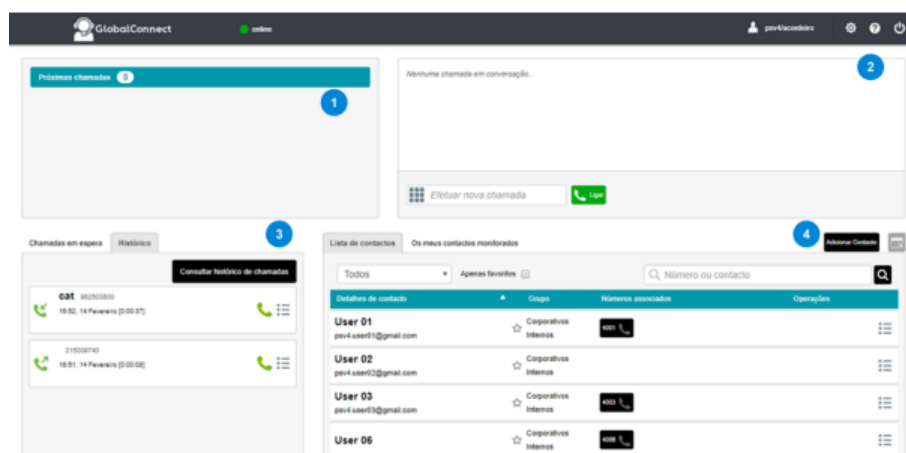


Figura 1: Interface da consola da Altice Labs

A interface gráfica, ilustrada na Figura 1, está dividida em várias áreas: área dedicada às chamadas de entrada, área para a chamada em curso, área para os contactos e uma pequena secção onde é apresentado ao utilizador as chamadas em pausa e o histórico. O desenho é relativamente moderno e intuitivo, sendo no geral, amigável para o utilizador. No entanto, é plausível argumentar que o espaço não foi bem dividido entre as áreas, sendo que algumas poderiam ser encurtadas de modo a apresentar ao utilizador uma maior quantidade de informação, sem que este execute nenhuma operação. Além disso,

foi também notado a falta de responsividade da aplicação, nomeadamente na secção das definições, que é acedida através de uma modal.

2.2.3 Problemas

Efetivamente, foi notada a ocorrência de vários erros e problemas no curto espaço de tempo que a consola foi utilizada. Não foi possível averiguar se estes erros ocorreram devido ao ambiente em que a consola foi testada, e se num ambiente de produção, com todos os serviços corretamente configurados, estes seriam resolvidos. O que é possível concluir é que a utilização do *softphone* externo e a necessidade de ter sempre a chamada de conexão ativa esteve na origem de vários dos problemas notados, contribuindo ainda negativamente a necessidade de reestabelecer a ligação sempre que estes problemas aconteciam.

2.3 A concorrência

De modo a estabelecer uma comparação do software da concorrência com o software desenvolvido pela Altice Labs, e também para ter conhecimento acerca do que o mercado nesta área, ao mais alto nível, tinha para oferecer, foi estudado o produto de várias empresas. Estas empresas foram estipuladas previamente, sendo que também foram analisadas outras que, por motivos de redundância, não serão abordadas neste documento. No entanto, certas empresas estudadas possuem software bastante mais completo do que uma consola de operador, encaixando na definição de **UC**.

O que é UC ?

De acordo com UC Today [16; 17], a plataforma líder de tecnologias desta área, **UC** é o termo utilizado para definir a maneira como diferentes sistemas de comunicação e de colaboração se conectam. Este define a integração de diferentes ferramentas de comunicação, como chamadas de voz, vídeo, mensagens instantâneas, presença, partilha de conteúdo, entre outros, numa única plataforma simplificada. Desta forma, o utilizador pode alternar entre modos de comunicação, na mesma sessão, no mesmo dispositivo, melhorando a experiência e a produtividade [18].

Efetivamente, é argumentável indicar que todos estes modos de comunicação vão além da definição de uma consola de operador, que apenas tem como objetivo fazer a gestão de um elevado número de chamadas, apesar que certos modos, como a troca de mensagens instantâneas, podem auxiliar nestas operações. Apesar de tudo isso, são funcionalidades que ficam ao dispor do cliente quando este adquire os produtos, e como tal, são objeto de comparação na Tabela 1, incluída na secção 2.3.4.

Os produtos de consola de atendimento estudados pertencem às seguintes empresas :

- Vodafone;
- RingCentral;
- 8x8;
- Landis Technologies.

Todas estas empresas adoptam a sua própria solução de consola de operador, tendo sido analisadas em detalhe, e que serão explicadas separadamente, sendo dada especial atenção as suas funcionalidades de modo a construir uma tabela comparativa (ver Tabela 1, incluída na secção 2.3.4).

2.3.1 Vodafone

A consola de operador da Vodafone, *Attendant Console*, trata-se de uma aplicação *desktop* para Windows e OS X, possuindo uma ferramenta de administração online no portal OneNet onde é possível configurar certas definições às quais apenas o administrador tem acesso, como por exemplo criar novas filas de espera [19; 20]. Com o intuito de evitar uma extensão desnecessária desta secção , serão abordadas e esclarecidas, a seguir, apenas as funcionalidades que ainda não foram apresentadas, visando a introdução de conceitos até então não discutidos, assim como a identificação das características presentes neste produto que divergem das encontradas no produto da Altice Labs. No final, será estabelecida uma comparação (ver Tabela 1, incluída na secção 2.3.4) onde estarão presentes todas as funcionalidades deste produto. As funcionalidades mencionadas são as seguintes:

- *Park* de chamadas: permite colocar a chamada em espera e retomá-la num outro telefone. A chamada não fica conectada à linha, mas sim numa área de espera virtual, permitindo que qualquer pessoa continue a chamada, ligando para a extensão onde a chamada fica à espera [21];
- Histórico customizável: O utilizador tem a possibilidade de estabelecer durante quanto tempo pretende armazenar o histórico, incluindo permanentemente [22];
- Gestão de filas de espera: O utilizador tem a liberdade de subscrever as filas de espera, existindo algumas filas que, por defeito, estão já associadas ao utilizador. Ao aceder à fila de espera, o utilizador consegue ver todas as chamadas nessa fila de espera, tendo a liberdade de atender qualquer uma. Na ferramenta de administração é possível configurar as definições destas filas, assim como criar novas [23; 24];

- Conferências: É possível efetuar chamadas de grupo com dois ou mais participantes. Para tal, é necessário ter duas chamadas em curso e juntar as duas chamadas, podendo posteriormente, adicionar mais pessoas até um máximo de quatro. É possível também colocar chamadas de conferência em pausa e separar os participantes (expulsar) [25];
- Diretório de contactos: A diretoria de contactos é ligeiramente limitadora, uma vez que os contactos pessoais do utilizador ficam armazenados localmente, implicando que, caso o utilizador troque de dispositivo, não tenha acesso aos contactos guardados previamente no outro dispositivo. No entanto, o utilizador pode facilmente importar listas de contactos através de ficheiros .csv [26; 27; 20];
- Log de erros: Todos os erros que acontecem na consola são armazenados num histórico de erros, que pode ser enviado diretamente para a equipa de suporte da One Net para que seja feito o *troubleshoot* [28];
- Integração *Customer Relationship Management* (**CRM**): É possível configurar e integrar um servidor **CRM** (Salesforce, Sugar CRM, Microsoft Dynamics CRM ou Outlook), sendo possível juntar toda a informação destes servidores, procurar contactos, gerir chamadas e consultar o seu histórico.

O que é um CRM ?

É uma estratégia que empresas utilizam para gerir os seus clientes. Isto permite que as organizações consigam simplificar certos processos, o que ajuda na construção de relações com esses clientes, no aumento vendas e na melhoria do serviço e do rendimento [29]. Neste cenário, **CRM** diz respeito a uma ferramenta para gestão de contactos, vendas, produtividade e outros, que através da integração com o software de consola de operador, permite que os dados de ambas as plataformas sejam sincronizados.

Interface gráfica

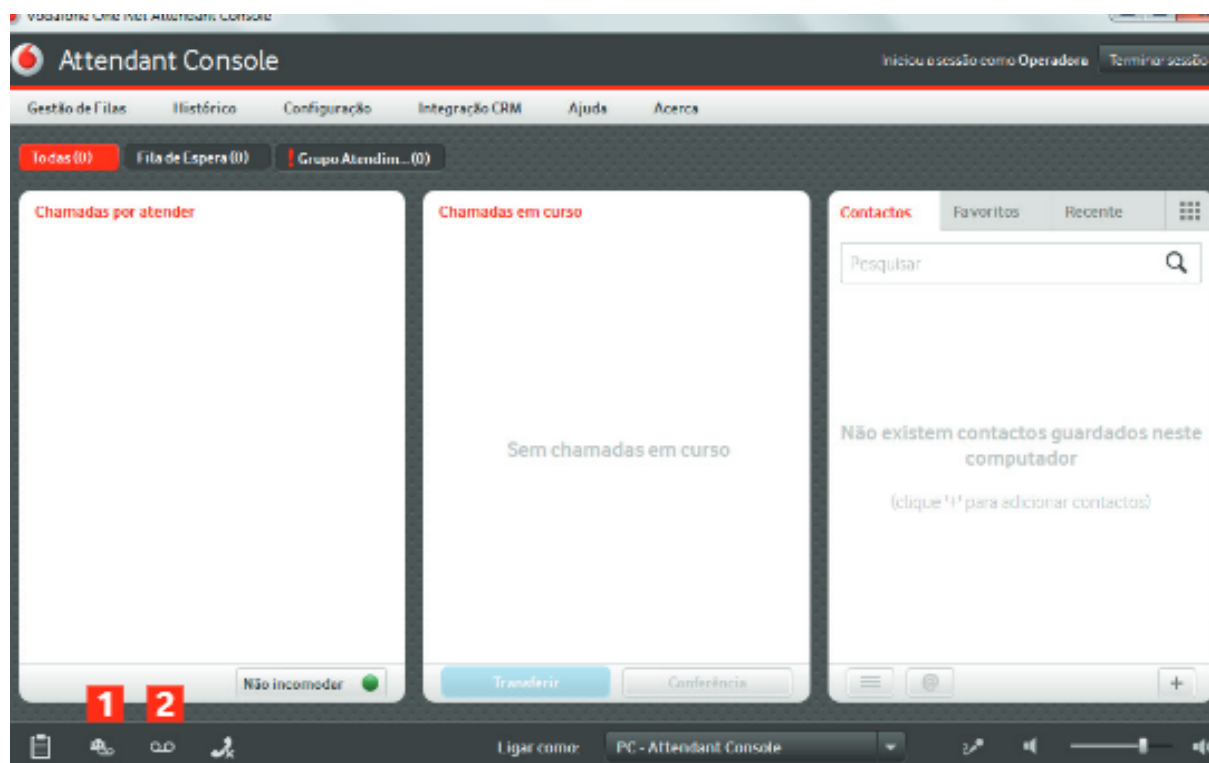


Figura 2: Interface da Consola de Operador da Vodafone

Como podemos ver através da Figura 2 é possível notar que se trata de uma aplicação de apenas uma página, podendo o utilizador abrir vários menus para configurar um elevado número de definições, como tamanho de letra, dispositivos de entrada e saída, entre outros. Além disso, é uma interface relativamente intuitiva, com uma separação principal da interface em três zonas (chamadas por atender, chamadas em curso e contactos), sendo também possível, de certo modo, editar a maneira de como a informação é apresentada ao utilizador.

Apesar disso, o que foi notado é que, devido ao espaço e tamanho da informação apresentada, não é possível apresentar muita informação ao utilizador de uma só vez, sendo necessário este recorrer ao *scroll*. Por exemplo, o menu de chamadas em curso apenas é capaz de apresentar duas chamadas de uma só vez sem a necessidade de uma barra de *scroll*, o que pode ser confuso para o utilizador caso este tenha uma lista grande de chamadas em curso. No entanto, como não foi possível aceder a uma demonstração gratuita deste software, não é possível tirar conclusões concretas sobre a experiência de utilização.

Por fim, apesar de ser um desenho intuitivo, é possível ver que este se encontra ligeiramente desatualizado e não vai de encontro a software desenvolvido mais recentemente, sendo que foram tomadas

algumas decisões duvidosas neste aspeto.

2.3.2 8x8 e Ring Central

8x8 e Ring Central são duas empresas na área das comunicações que fornecem produtos relevantes para o estado de arte. Após as suas plataformas serem estudadas, foi possível concluir que, apesar de possuírem uma plataforma e produtos distintos, aproximadamente, dispõem das mesmas características e funcionalidades, com algumas ligeiras diferenças. Por esse motivo, de forma a sintetizar esta secção, uma grande parte da informação estará relacionada a ambas as plataformas, sendo esta explicada em conjunto.

É importante mencionar ambas as soluções estudadas nesta secção são, quando comparadas aquelas previamente estudadas, de maior dimensão e complexidade, com um número muito mais elevado de funcionalidades, sendo que, as funcionalidades de consola de operador fazem parte de toda a solução de **UC**. Por esse motivo, foram apenas estudadas aquelas mais relevantes através da documentação disponível online, sendo que não foi possível testar uma demonstração gratuita de nenhuma das plataformas.

8x8

8x8 é uma empresa americana que fornece produtos de *Voice Over IP*. Possuem uma plataforma de comunicação que integra comunicação por voz, vídeo, chat, *contact center*, comunicações embebidas e APIs, denominada por *8x8 eXperience Communications Platform*, podendo as várias aplicações desta plataforma ser utilizadas em *desktop*, *mobile* ou *web* [30].

Esta plataforma possui várias opções de compra, com diferentes preços associado a cada uma. Apesar de existirem seis opções de compra, estas podem ser divididas em dois grupos : as primeiras três dizem respeito a *Business Communications*, semelhante a uma consola de atendimento com algumas funcionalidades adicionais, sendo que as últimas três, devido à adição de funcionalidades, convertem o produto de *Business Communications* para um *Contact Center* e que, portanto, não serão objeto de comparação com as restantes plataformas analisadas [8]. No entanto, como existem várias opções de compra para o equivalente à consola de operador, foi analisada aquela que possuía mais vantagens.

Ring Central

Ring Central é uma empresa também americana que fornece serviços de comunicações pela *cloud*. Possuem vários produtos nesta área, cada um dedicado a uma subárea, como telefone, mensagens, Fax, **SMS**, vídeo, entre outros. A informação relativa a planos e preços [31] indica que não é possível a sua

compra separada, estando estes produtos incluídos em diferentes pacotes disponíveis para compra ao utilizador :

- Ring Central MVP - pacote com capacidades de vídeo, mensagem e telefone, sendo este o produto analisado, uma vez que é aquele que dispõe de competências de consola de operador;
- Ring Central Video - pacote que tem como foco principal a gestão de reuniões com voz e vídeo de alta qualidade;
- Contact Center - pacote com as competências já estipuladas necessárias para o funcionamento de um centro de contacto.

É possível adquirir o produto Ring Central MVP através de diferentes opções de compra, sendo que, quanto mais funcionalidades este possuir, mais custos estarão associados a este [31].

Funcionalidades

Como foi previamente explicado, as funcionalidades/caraterísticas destes produtos são semelhantes, e como tal, os pontos aqui mencionados e explicados são comuns a ambos os produtos.

É importante explicar que nesta lista não consta a totalidade das funcionalidades de cada plataforma, uma vez que, como estipulado previamente, estas compõe uma solução de **UC**, sendo que várias não são relevantes para este trabalho. Além disso serão também apenas mencionadas, por motivos de redundância, aquelas que, até agora, não foram abordadas em produtos anteriores, exceto se as funcionalidades contenham diferenças importantes. A lista total estará contida na Tabela 1 , incluída na secção 2.3.4.

As funcionalidades mencionadas são as seguintes:

- *Flip* de chamadas – Permite mover uma chamada entre dois dispositivos que estejam associados ao utilizador sem a necessidade de reiniciar a chamada [32; 33];
- Gestão de Filas de espera - Como a consola anteriormente explicada, também esta possui uma gestão de filas de espera, no entanto, é bastante mais robusta do que a anterior. As chamadas que são colocadas numa fila de espera são servidas aos membros (operadores) da fila de forma cíclica, exceto a membros que estejam em chamada, offline ou em *Do not Disturb*. Assim que o utilizador tiver terminado uma chamada, recebe novamente uma chamada da fila, depois de um tempo configurado pelo administrador [34]. Estas filas são capazes também de possuir membros primários e secundários, sendo que os secundários servem como uma opção de *overflow*, ou seja,

as chamadas apenas são servidas a estes membros caso todos os membros primários se encontrem numa chamada [35]. Além disso, é possível também configurar se os operadores têm ou não a possibilidade de se registarem numa fila [36] e de configurar opções de reencaminhamento para cada fila [37; 38];

- *Barge, Monitor e Whisper* - Permite que supervisores, selecionados na ferramenta de administração, façam a monitorização de uma chamada. Esta monitorização pode ser silenciosa (*Monitor*), permitindo ao utilizador entrar numa chamada sem qualquer tipo de alerta, não dando conhecimento a nenhum dos intervenientes, mas também não dando a possibilidade ao supervisor de intervir. É dada também a possibilidade ao supervisor de fazer o *Whisper*, entrando na chamada, podendo, neste cenário, falar em privado para o operador, sem o conhecimento do cliente, através de áudio unidirecional. Por fim, o *Barge*, permitindo ao supervisor entrar na chamada notificando os intervenientes e tornando essa mesma chamada numa conferência, onde todos são capazes de ouvir e interagir [39; 40];
- Conferências - É possível realizar conferências com um máximo de 500 pessoas, com vídeo e voz *High Definition (HD)* e com controlos de moderador avançados [8; 31].
- *Hot Desking* - permite transformar um dispositivo num *deskphone* partilhado, podendo assim ser utilizado por múltiplas pessoas, desde que seja durante períodos de tempo diferentes, sem concorrência [41; 42];
- Transcrição de *voicemail* - Permite transcrever automaticamente as mensagens de correio de voz para texto, sendo este enviado para o utilizador através do email [43]. É possível também o utilizador aceder à sua lista de *voicemail* quer através das extensões no *softphone* ou *deskphone*, mas também através da aplicação, onde, na secção das chamadas, é possível filtrar por *voicemails* e efetuar um elevado número de operações, como *callback* e *download* [44; 45];
- Mensagens entre equipas - Dá a possibilidade aos utilizadores de não só trocar mensagens com outros utilizadores, mas também de criar salas totalmente customizáveis, públicas ou privadas, com partilha de ficheiros [46; 47];
- Armazenamento de ficheiros de gravações de chamadas ilimitado durante um máximo de 30 dias [8; 31];
- *Auto Attendant Multi Level* - Rececionista automatizado que permite criar experiências customizáveis para os clientes através de múltiplos menus. Devido à sua característica de *Multi Level*, é possível

estabelecer múltiplas camadas de personalização baseadas na localização, prioridade e números de clientes. Através da interação com os menus de voz automatizados, permitem que os clientes se dirijam para as filas ou grupos adequados sem a necessidade de intervenção manual de um operador [48; 49; 50];

- Reencaminhamento avançado - Permite definir as próprias regras de reencaminhamento. Estas regras pode ser aplicadas para um certo utilizador, grupos de atendimento, grupo de clientes e para um específico espaço de tempo, podendo reencaminhar para o voicemail ou para outro utilizador quando este está ocupado. É possível também, através desta ferramenta, definir uma lista de utilizadores bloqueados, automaticamente rejeitando as suas chamadas [51; 52];
- Múltiplas integrações com diferentes aplicações **CRM** - Integrações com aplicações de diferentes áreas, como produtividade, serviços, suporte, entre outros, permitindo executar diferentes operações e atividades relativas a aplicação a integrar, mas a partir do cliente da consola de operador. Revela-se como sendo bastante diferente das integrações mencionadas anteriormente no software pertencente à Vodafone, não só em número de plataformas integradas [53; 54], mas principalmente pelo que as vantagens que estas integrações colocam ao dispor do utilizador, criando uma área dedicada à integração dentro do **CRM**, permitindo que este efetue várias operações da consola de forma direta, sem a necessidade de alternar entre as duas aplicações [55; 56].
- *Single Sign on* - Permite facilitar e agilizar o processo de *onboarding* de novas empresas, permitindo que estas se registem nas plataformas com as suas credenciais corporativas [8; 31];
- Múltiplas ferramentas de relatórios e análises - estes produtos possuem múltiplas ferramentas e funcionalidades que permitem gerar vários relatórios e consultar diferentes dados relativos a chamadas. Através disto é possível fazer uma gestão de qualidade, avaliar os operadores e consultar vários dados estatísticos [57; 58].

Interface Gráfica

Relativamente à interface gráfica, devido a não ter sido possível testar nenhuma aplicação de nenhum produto das duas plataformas, apenas foi possível consultá-las através de documentação online, variando de recortes colocados em diferentes páginas, a páginas de suporte e de treino onde é possível aprender a utilizá-las. Apesar disso, não sendo possível testar e utilizar a aplicação, não é possível tirar conclusões concretas, o que torna, no entanto, a sua consulta importante, uma vez que serve para, posteriormente, definir as interfaces gráficas da aplicação a desenvolver.

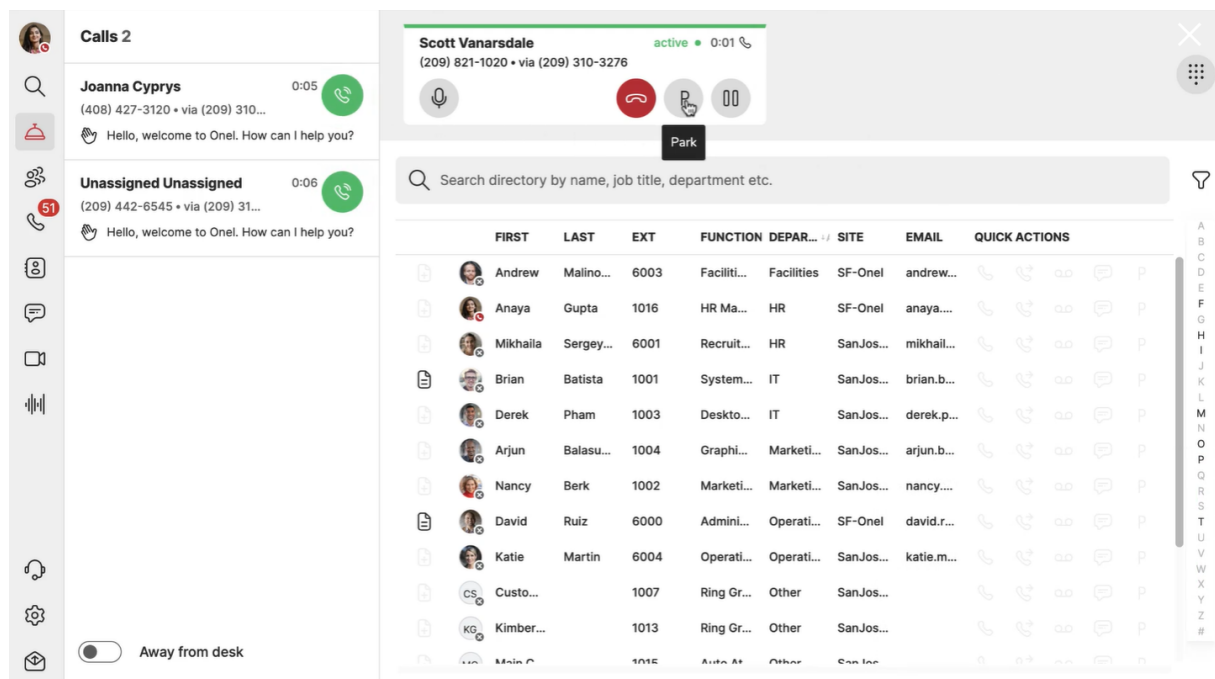


Figura 3: Interface de uma aplicação da plataforma 8x8 [1]

Na Figura 3 consta uma imagem retirada da área de treino da plataforma 8x8. Esta imagem é apenas um exemplo de como o utilizador pode interagir com as chamadas, uma vez que existem mais aplicações nesta plataforma, incluindo aplicações móveis e web. É possível ver que existe um elevado número de menus que é possível expandir, face a todas as componentes de vídeo, mensagens, e outras funcionalidades. No geral, é uma interface bastante intuitiva, com um desenho moderno e atualizado, e que serviu para retirar algumas ideias para o planeamento da interface a desenvolver.

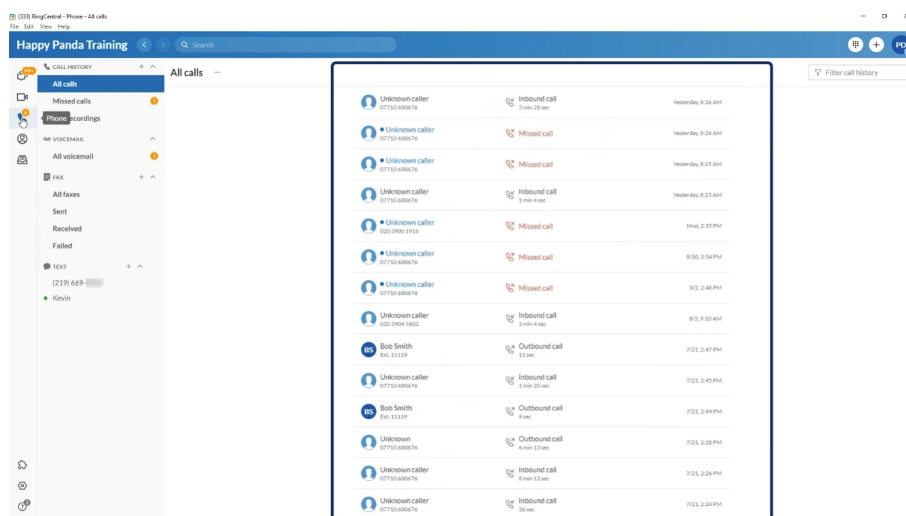


Figura 4: Interface de uma aplicação da plataforma RingCentral

Na Figura 4 consta uma imagem retirada da área de suporte da plataforma Ring Central. À semelhança da plataforma 8x8, existem várias aplicações, sendo que a que consta na figura diz respeito à aplicação desktop do produto estudado, Ring Central MVP. São interfaces também semelhantes, apesar da disposição de informação ser diferente, com um desenho, no geral, moderno e atualizado.

2.3.3 Landis Technologies

Landis Technologies é um parceiro da Microsoft que fornece suporte e consultoria em **IT** e *Unified Communications*, com uma equipa de consultores certificados pela Microsoft que fornecem instalação, migração e suporte para todos os clientes [59] .

O produto de consola de atendimento que é disponibilizado é uma aplicação *desktop* e destaca-se sendo uma *Attendant Console* para Microsoft Teams. Trata-se de um cliente completo para o Microsoft Teams, com o objetivo de facilitar uma gestão de chamadas para rececionistas, e que, por ser um cliente completo, elimina a necessidade da utilização do cliente do Teams, tornando-a uma integração total e eliminando por completo a necessidade de alternar constantemente entre duas aplicações [60; 61].

Efetivamente, um grande número de produtos analisados possibilitam ao utilizador a integração de com várias outras plataformas, quer sejam estas de **CRM**, armazenamento, entre outros. Dada a grande adesão a software como o Teams devido a conversão para um ambiente de trabalho virtual em 2020 [62], estas integrações são de grande valor para o utilizador final, pois facilitam bastante o seu uso simultâneo, sendo possível, de certa forma, utilizar ambas as aplicações na mesma plataforma.

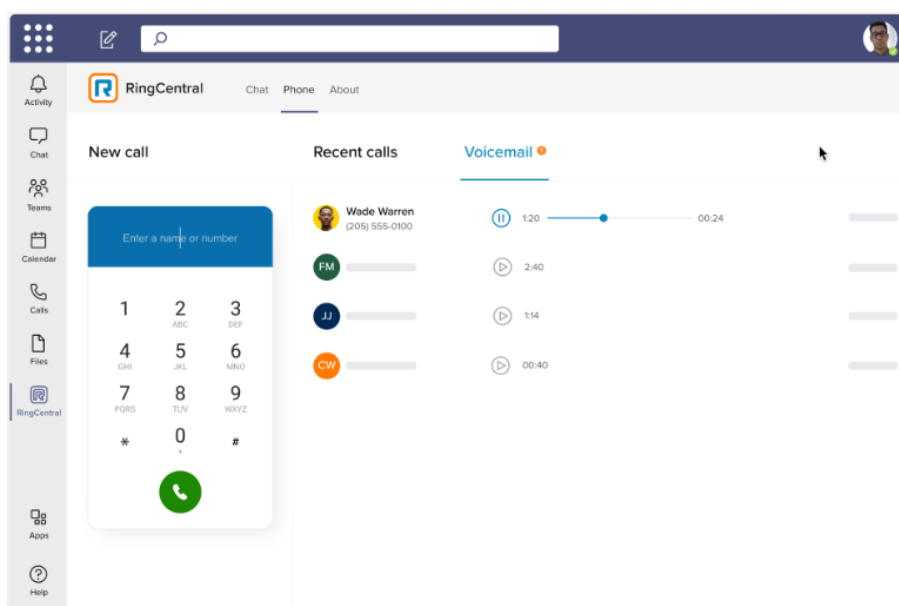


Figura 5: Integração Ring Central com Teams

Na Figura 5 é possível ver como uma integração destas é suposto funcionar. Esta integração diz respeito à Ring Central com a aplicação Microsoft Teams [61], e é uma das várias possíveis, sendo semelhante à integração com a consola da empresa 8x8 [63]. É possível, dentro do cliente do Microsoft Teams, aceder a integração da Ring Central incluindo algumas funcionalidades, como iniciar reuniões, aceder a históricos, e a serviços de *voicemail* e fax, entre outros. Deste modo, caso o utilizador necessite de consultar ou executar alguma operação relacionada com as funcionalidades integradas, é desnecessário aceder à aplicação da Ring Central, bastando aceder ao menu da integração dentro do cliente do Microsoft Teams [64].

Outra grande vantagem da utilização destas integrações é a possibilidade de efetuar chamadas através do Microsoft Teams, usando a *Enterprise-grade telephony* fornecida pelo *Private Branch Exchange (PBX)* da consola de operador, sendo as chamadas ilimitadas [65; 66; 67].

No entanto, apesar das inúmeras vantagens que o utilizador pode usufruir caso utilize este software, estas integrações são limitadoras, uma vez que restringem as operações que podem ser realizadas. Ou seja, existe uma vasta lista de tarefas que não podem ser realizadas dentro do menu da consola no cliente Teams, sendo necessário recorrer ao cliente da consola para que estas possam ser feitas.

É aqui que se destaca o software da Landis Technologies. O objetivo deste produto passa por ser um cliente otimizado para gerir múltiplas chamadas, aprimorando a experiência de chamadas no Teams sem a necessidade de integração, como é o caso acima mencionado. O propósito final é que, sendo um cliente completo, seja desnecessário a utilização da aplicação Microsoft Teams, evitando assim uma experiência negativa ao utilizar software separado e ao constantemente variar entre estes [68].

Funcionalidades

Como foi dito, a grande vantagem e o que destaca esta consola entre as outras é a sua integração completa com o Microsoft Teams. Tal como indicado anteriormente, e para evitar repetição de informação, esta secção apenas irá indicar algumas funcionalidades que destacam esta solução das previamente mencionadas e que não foram explicadas. Como referido, na Tabela 1, incluída na secção 2.3.4 estão listadas todas as funcionalidades para que todos os produtos estudados possam ser comparados. Assim, as funcionalidades que diferenciam a consola da Landis são:

- *Consult Transfer* - além dos dois tipos de transferência de chamadas previamente explicados (com e sem consulta), esta consola introduz um novo tipo de transferência. Esta permite enviar um cartão adaptativo ao utilizador para o qual a chamada irá ser transferida, que funciona como uma notificação, alertando que o utilizador lhe está a transferir uma chamada. Através deste cartão

adaptativo, o utilizador pode aceitar ou rejeitar a transferência [69; 70];

- Calendário - É possível aceder ao calendário de utilizadores que estejam associados a uma conta Microsoft Exchange, sendo possível ver reuniões das quais o utilizador fará parte e outros eventos que este deseje adicionar ao seu calendário [71].

Interface Gráfica

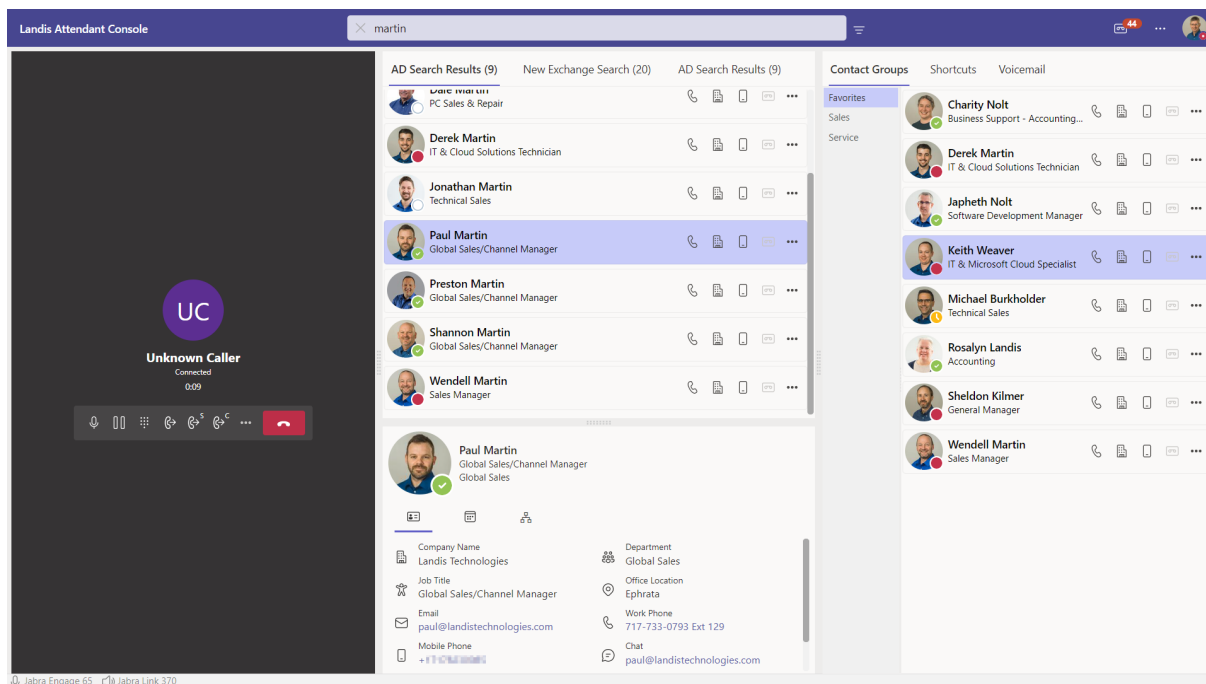


Figura 6: Interface da Landis Attendant Console for Microsoft Teams [2]

Como seria de esperar pelo que pode ser observado na Figura 6, a interface gráfica deste produto é semelhante à interface da aplicação Microsoft Teams, seguindo o mesmo desenho, na tentativa de proporcionar a mesma sensação para utilizadores que transitam ou utilizam o Teams. Foi dada especial atenção à transferência de chamadas, que é uma operação utilizada frequentemente. É também altamente customizável, sendo possível alterar a informação exibida e a disposição dos menus. No entanto, é de notar que, apesar disso, foi dada pouca atenção às chamadas em espera e em pausa, que é algo também bastante importante. Contudo, não se pode concluir nada em concreto, uma vez que não foi possível testar uma demonstração desta consola.

Para concluir, é uma solução que se destaca em relação às outras devido à sua integração total do Teams, mas existem no mercado soluções mais completas e robustas, como o produto proporcionado

pela Ring Central ou 8x8, que possui também uma integração, apesar de não total, robusta da plataforma Microsoft Teams, com a vantagem de possuir um número adicional de funcionalidades de elevada importância e conveniência.

2.3.4 Comparação

Como foi dito anteriormente, o estado de arte apresentado, nomeadamente a investigação sobre outras plataformas que sirvam de referência para o software já desenvolvido e a desenvolver da Altice Labs, teve como objetivos principais a familiarização com diferentes conceitos e definições deste tema. Além disso, serviu também para que, no final, pudesse ser estabelecida uma comparação entre todas estas plataformas, de forma a analisar o papel do software da Altice no mercado, e identificar possíveis falhas e aspetos a melhorar.

Na Tabela 1 consta a lista total de funcionalidades identificadas, indicando se a plataforma em questão possui essa funcionalidade. É importante mencionar que todo este levantamento foi feito através da documentação disponível online de cada uma das plataformas analisadas. Este processo heurístico não é ideal pois alguma desta documentação pode-se encontrar desatualizada, ou alguma informação omitida.

Tabela 1: Comparação entre as plataformas estudadas

Funcionalidades	Altice Labs	Vodafone	Landis	RingCentral	8x8
Efetuar Chamadas	✓	✓	✓	✓	✓
Reencaminhar Chamadas	✓	×	×	✓	×
Transferir Chamadas	✓	✓	✓	✓	✓
Capturar Chamadas	✓	×	×	✓	×
<i>Flip</i> de Chamadas	×	×	×	✓	✓
Park de Chamadas	×	✓	×	✓	✓
Histórico Total / customizável	×	✓	✓	✓	✓
Toque simultâneo / <i>Ring Groups</i>	✓	✓	×	✓	✓
Status de presença	✓	✓	✓	✓	✓
Diretório de Contactos	✓	✓	✓	✓	✓
Gestão de filas de espera	×	✓	×	✓	✓
Conferencias	×	✓	✓	✓	✓
Reuniões com voz e vídeo de alta qualidade	×	×	×	✓	✓

<i>Voicemail</i> visual com transcrição	×	×	✓	✓	✓
<i>Barge, Monitor e Whisper</i>	×	×	×	✓	✓
<i>Hot Desking</i>	×	×	×	✓	✓
<i>Auto Attendant</i>	×	×	×	✓	✓
Gravação de Chamadas	×	×	×	✓	✓
Armazenamento ilimitado	×	×	×	✓	✓
<i>Single Sign-on</i>	×	×	×	✓	✓
Múltiplas ferramentas de análise e relatórios	×	×	×	✓	✓
Troca de Mensagens	×	×	✓	✓	✓
Fax ilimitado	×	×	×	✓	✓
Integrações	×	✓	✓	✓	✓
Softphone Integrado	×	✓	✓	✓	✓

Através desta tabela, é possível comparar de imediato todas as plataformas estudadas. De todas as plataformas, a que se torna mais importante comparar é a consola da Vodafone, por pertencer ao competidor mais direto, sendo que outros competidores internacionais, por exemplo, 8x8, apenas possui chamadas ilimitadas para Estados Unidos e Canadá [8], sendo necessário custos adicionais, como aumentar o plano, para que a lista de países seja aumentada.

Quando comparada com a Vodafone, podemos concluir que não existe uma grande discrepância entre os produtos. Existe, como seria de esperar, algumas diferenças, sendo que a consola de operador da Altice Labs possui algumas funcionalidades não presentes na consola da Vodafone. No entanto, o caso contrário tem bastante mais ocorrências, uma vez que funcionalidades como integrações, conferências, gestão de filas de espera, histórico total e *park* de chamadas, em conjunto com a integração do *softphone*, que, como foi visto anteriormente, introduziu alguns problemas e complicações de utilização na consola da Altice Labs. Assim, é seguro concluir que a Vodafone, em termos de funcionalidades que oferece, possui um produto ligeiramente mais completo.

Porém, o mesmo não se pode dizer quando a comparação é feita com os competidores internacionais, RingCentral e 8x8. Existe um elevado número de funcionalidades que estas plataformas oferecem, incluindo funcionalidades de **UC** que, apesar de se encontrarem fora do âmbito de uma consola de operador, são adições importantes e que facilitam muito o trabalho ao operador, o que no final, melhora a experiência do cliente. Isto faz com que, no geral, estes dois produtos sejam produtos bastante mais

completos, pelo que são eles que dominam o mercado na área de sistemas de telefonia para negócios e soluções de **UC** [72; 73].

No geral, conclui-se que o produto da Altice Labs tem um lugar no mercado, uma vez que apresenta todas as funcionalidades cruciais ou base para o funcionamento normal de uma consola de operador. No entanto, existem vários produtos mais completos e com menos complicações, devido ao maior número de funcionalidades e devido ao *softphone* integrado, que fazem com que este produto possa não ser a primeira escolha dos clientes.

2.4 Sumário

Neste capítulo foram discutidos vários conceitos importantes para o tema da dissertação. Foi abordado o conceito do problema em questão, assim como diferentes conceitos relacionados relevantes, de modo a ser obtido um melhor conhecimento acerca do papel que este trabalho tenciona alcançar.

Foram exploradas as soluções de consola de atendimento da Altice Labs, seguido de um estudo de outras plataformas semelhantes de outras organizações, de modo a que todo este estudo ser condensado numa breve comparação, exposta na Tabela 1, bastante importante para se tirar conclusões acerca do produto desenvolvido, e que se pretende melhorar.

Com o trabalho desenvolvido neste capítulo, é possível planear e traçar objetivos sucintos e realistas de acordo com a solução a implementar, e que serão abordados no capítulo 3.

Capítulo 3

Planeamento

3.1 Âmbito do Projeto

Tendo em conta que o objetivo deste trabalho está relacionado com melhorar um produto já em produção, existe uma quantidade diferente de abordagens possíveis para o desenvolvimento que, antes de dar início a fase de planeamento, é necessário decidir.

As 3 abordagens seguintes foram ponderadas :

- desenvolver uma aplicação nova, focando esta parte do desenvolvimento no *frontend*, utilizando certas *Application Programming Interface (APIs)* já em produção para implementação de certas funcionalidades;
- integração das **APIs** já em produção com um outro cliente já desenvolvido por outrem;
- melhorar a aplicação já desenvolvida.

A terceira abordagem rapidamente foi descartada, uma vez que o projeto atual, ao longo dos anos e devido às alterações que a equipa de desenvolvimento sofreu, as várias alterações que foram feitas implicaram uma má manutenção do código, levando a que cada vez mais seja mais dispendioso, especialmente no que diz respeito ao tempo de desenvolvimento [74], dar continuidade a esta aplicação. Como já foi explorado anteriormente, este foi um dos motivos que levou à necessidade de desenvolver uma nova aplicação que, de certa forma, a substitua. No entanto, escolher outra opção que não esta teria certas implicações, que serão discutidas mais à frente.

Efetivamente, numa fase inicial, a segunda abordagem foi escolhida, uma vez que desta forma o trabalho partiria de uma interface já construída, dando mais tempo e liberdade para o desenvolvimento de uma aplicação mais robusta e completa, que como veremos mais à frente, foi um problema enfrentado, tornando o foco na implementação da integração.

Para isso foi realizada uma investigação acerca de software semelhante aquele que se pretende desenvolver que fosse possível integrar com todo o serviço já desenvolvido e em produção disponibilizado pela Altice Labs.

3.1.1 Possíveis Integrações

Esta pesquisa revelou-se como sendo mais complicada devido à escassez de informação técnica disponibilizada dos produtos. Foi necessário uma pesquisa bastante aprofundada de modo a encontrar alguma informação relativa a este tema. Acrescentando ao problema, nem todas as plataformas se pronunciam acerca deste tema, indicando que para obter informação, é necessário entrar em contacto com os responsáveis pela plataforma [75; 76], o que não era o pretendido para o tempo alocado para esta tarefa.

No entanto, foi na mesma possível levantar alguma informação, ainda que bastante limitada, acerca da possibilidade da integração, sendo que num caso específico foi possível testar uma versão incompleta dessa mesma integração.

Esta parte baseou-se em investigar os produtos de consola de operador já estudados e outros recomendados ou encontrados acerca da possibilidade de integração, geralmente, com um IP **PBX**.

O que é um IP **PBX**

Um IP **PBX** é um sistema de telefonia usado em empresas e organizações para gerir as chamadas telefónicas internas e externas por meio da rede IP, funcionando como uma rede telefónica privada dentro de uma empresa ou organização para lidar com comunicações internas e gerir as chamadas de entrada e saída para a rede telefónica pública [77].

Relativamente às consolas de operador estudadas e apresentadas neste documento, não foi possível tirar nenhuma conclusão acerca de uma possível integração uma vez que não foi encontrada nenhuma informação relativa a este tema na documentação desses produtos. No entanto, após alargar o estudo a outros produtos, os resultados foram diferentes, uns sugerindo que uma integração com um IP **PBX** poderia ser possível, mas seria necessário entrar em contacto, e outros que indicaram que várias integrações eram suportadas, explicando como poderia ser feita, o que permitiu testar e analisar os resultados obtidos.

3.1.2 Integração com Bitrix24

Bitrix24 trata-se de uma plataforma de colaboração que tem a capacidade de gerir os vários aspetos das operações e tarefas diárias de qualquer negócio ou organização. Isto inclui todas as tarefas como **CRM**,

gestão de projetos e colaboração para uma total gestão do negócio [78]. Apesar de se tratar de um produto com um elevado número de funcionalidades que não são pretendidas para este trabalho, a ferramenta de *contact center* [79], apesar de não ser propriamente o pretendido, como visto anteriormente, possuía várias funcionalidades interessantes. Por ser uma plataforma *open source*, é possível criar uma solução à medida de cada um, desenvolvendo e adicionando novas funcionalidades ao produto e alterar as ferramentas já implementadas para acomodar casos de uso específicos, incluindo integrações com outros serviços ou produtos *Software as a Service (SaaS)*, o que era o objetivo para esta fase [80].

Após uma breve procura na documentação deste produto foi verificado que seria possível conectar um **PBX** de duas formas distintas, através da REST **API** ou através de um módulo adicional, cobrado a parte, denominado por *SIP connector* [81].

Esta plataforma apresenta várias opções de compra, e uma vez que o plano mais simples é gratuito, foi relativamente fácil efetuar o registo na plataforma e rapidamente foi aprovada uma instância para o utilizador criado. Apesar disso, o plano gratuito revelou-se bastante limitador, como seria de esperar, o que dificultou a utilização de certas funcionalidades. Além disso, foi necessário ativar um *trial* gratuito do plano *premium* para que fosse possível utilizar e testar a integração com o **PBX** através da REST **API** [82]. Apesar disso, o plano *premium* ativado não possui o módulo adicional *SIP connector*, uma vez que este era cobrado à parte, o que impossibilitou, de forma gratuita, testar a integração via **SIP**, ficando esta secção limitada à informação disponível na documentação. Esta documentação indica que, quando a conexão deste tipo é realizada, para as chamadas externas, a plataforma conecta-se ao servidor especificado e estabelece a conexão depois da autorização, e para as chamadas recebidas, é necessário configurar o IP **PBX** utilizando os parâmetros especificados na área das chamadas recebidas. Depois disto, é possível iniciar e receber chamadas para o número configurado.

Relativamente à integração via REST **API**, esta não funcionou de acordo com o esperado. Só quando este tema foi explorado em específico, é que foram notadas as diferenças e as implicações da integração, quando feita desta forma, ao invés de **SIP**. Enquanto que, com a utilização do *SIP connector*, as chamadas são sempre realizadas na aplicação, sendo dado acesso à área da telefonia, o mesmo não acontece com a integração via REST **API**. Quando esta integração é feita, todo o roteamento e tráfego de voz é gerido pelo **PBX**, contudo, não é possível fazer a gestão da chamada via aplicação Bitrix24, quer seja esta no *browser*, *desktop* ou *mobile*, passando a servir apenas para armazenamento de dados relativos a essas chamadas, sendo necessário que qualquer operação nessas chamadas seja feita no cliente do **PBX** a integrar [83].

Apesar disto, foram executados os testes. Incorporar um **PBX** com este produto de consola de

operador através da REST **API** é feito maioritariamente através da consola, na área de configuração, onde é possível ter acesso à secção de *add-ons*, onde todas as integrações compatíveis, verificadas pela Bitrix24, se encontram (ver Figura 7).

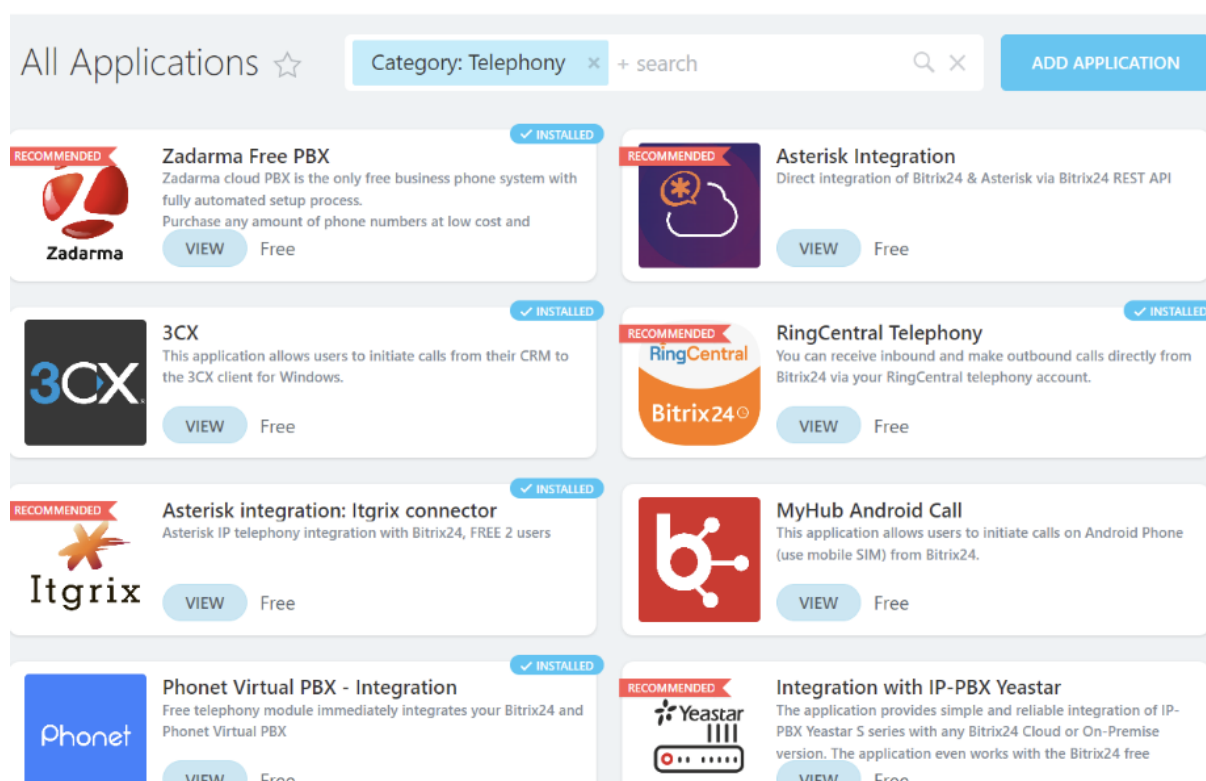


Figura 7: Imagem contendo algumas integrações (obtida em novembro de 2022)

A integração pode ser realizada acedendo à aplicação que se pretende incorporar, instalando e associando as contas através das chaves fornecidas. Na data em que este tema foi investigado, notou-se a existência de vinte e quatro integrações possíveis. No entanto, devido à necessidade de efetuar registo no **PBX** a integrar, foi apenas possível testar uma das vinte e quatro, *Zadarma Free PBX*, uma vez que apenas essa permitiu a obtenção de uma licença de demonstração que permitisse efetuar o pretendido. Apesar disso, é indicado que, independentemente da aplicação a integrar, o comportamento é o mesmo [83].

Depois da conexão estar realizada, o comportamento da integração foi de encontro ao que foi previamente indicado na documentação. Mesmo assim, todos os testes conduzidos foram bastante limitados, devido à falta de um número associado à consola, uma vez que o seu aluguer tinha custos associados.

Por este motivo, foi apenas possível testar chamadas feitas, não sendo possível verificar para chamadas recebidas. Além disso, pelos mesmos motivos, nenhuma chamada era estabelecida devido à falta desse número, sendo que qualquer chamada iniciada era rapidamente terminada. Quando a chamada

era iniciada via área de telefonia Bitrix24, a chamada era redirecionada para o **PBX** incorporado, tocando com o nome *callback*, como visto na Figura 8, sendo necessário atender essa chamada antes desta ser enviada para o destino. Esta operação é da total responsabilidade do **PBX**, sendo que, neste cenário, a Bitrix24 é apenas responsável por enviar o pedido de início.

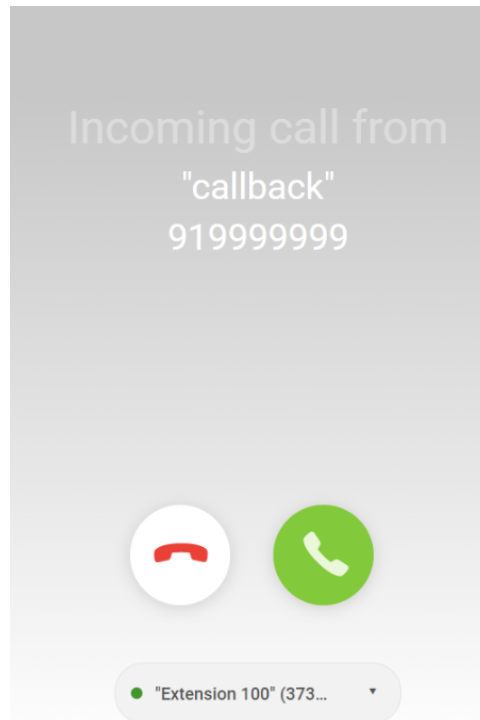


Figura 8: Imagem contendo a chamada enviada pela REST **API** pela Bitrix para o PBX integrado

De outro modo, as chamadas podem também ser iniciadas do lado do **PBX**, através do cliente desenvolvido por cada empresa ou organização responsável por este, no caso da integração testada, através da página web. Em qualquer dos casos, sempre que uma chamada é recebida ou iniciada, é apresentado um formulário dentro da aplicação Bitrix24 com todos os dados relevantes (ver Figura 9), não sendo possível, no entanto, efetuar qualquer operação.

The image shows a Bitrix24 interface window titled "Calling 919 999 999" with a "Minimize" button in the top right. The main content area displays a contact profile with a placeholder icon and the text "No name". Below this, there is a "Deals" section for the contact "919999999 - Outgoing call", showing a "New" status and a progress bar. The "Responsible person" is listed as "andre-f.mendes@alticelabs.com". At the bottom of the window, there are three buttons: "COMMENT", "DEAL", and "INVOICE", and a red "CLOSE" button at the very bottom.

Figura 9: Formulário apresentado ao utilizador na Bitrix24

Tendo em conta que a Bitrix24, como referido previamente, se foca não só em gestão de chamadas, mas também proporciona uma plataforma de colaboração para negócios, a integração via REST **API** é uma funcionalidade que tem a sua utilidade, permitindo agilizar o processo de gestão de chamadas, sem ser necessário aceder à área de telefonia. No entanto, no contexto deste projeto, não tem grande utilidade, pois passa a ser uma plataforma que, relativamente à telefonia, apenas tem o papel de armazenar e consolidar informação, não eliminando a necessidade de um cliente para gerir as chamadas, o que era o pretendido.

Adicionalmente, foram também explorados brevemente os *webhooks*. Estes permitiam desenvolver integrações customizáveis através de métodos REST **API** associados a certas tarefas ou atividades realizadas na plataforma [84]. Com estes *webhooks*, seria possível, através de um breve formulário, associar os métodos REST **API** a serem enviados, por exemplo, quando uma chamada fosse recebida ou efetuada, servindo para enviar dados da Bitrix24 para um serviço externo, ou de um serviço externo para a Bitrix24 [85]. Estes *webhooks* são bastante úteis, porque permitem integrar a área de telefonia com qualquer serviço externo de forma rápida e simples, sem a necessidade de publicar a integração a lista de *add-ons*, como ilustrado na Figura 7. No entanto, apesar dos *webhooks* terem a sua utilidade, como também apenas servem para passagem de informação e como, nesta fase, já se tinha concluído que esta plataforma não ia de encontro com aquilo que se pretendia, esta área não foi explorada em mais detalhe.

3.1.3 Integração com Voice Operator Panel

Voice Operator Panel é um *softphone* profissional e uma consola de operador para rececionistas e operadores [86]. Como indicado na página web deste produto, é ideal para revendedores de IP **PBX** que desejam oferecer um software de consola de operador aos clientes [86], o que vai de encontro aquilo que se pretende.

Uma breve pesquisa sobre esta aplicação rapidamente indica que tem que ser obrigatoriamente incorporada com um IP **PBX**, ao contrário das soluções estudadas até agora que ofereciam o seu próprio **PBX** [87]. Por esse motivo, a aplicação possui, na página web, uma área dedicada explicando todas as plataformas compatíveis, incluindo um guia de como as conectar, e uma secção indicando como pode ser configurado um **PBX** externo genérico. Esta secção indica que é necessário a criação de uma conta **SIP** que a plataforma irá utilizar para se registar no **PBX**. Esta conta é utilizada para receber todas as chamadas, monitorar outras extensões e transferir chamadas para essas extensões. Por este motivo, esta conta deverá ser somente dedicada para a plataforma, e não partilhada com outro telefone ou operador, ficando com o papel de gerir todas as chamadas direccionadas para a organização ou empresa em questão.

Depois desta conta ter sido criada, foi relativamente simples o registo na plataforma de forma a obter acesso a uma versão de demonstração gratuita, que permitiu instalar a aplicação e testar as suas funcionalidades, estando esta já conectada ao IP **PBX** pretendido.

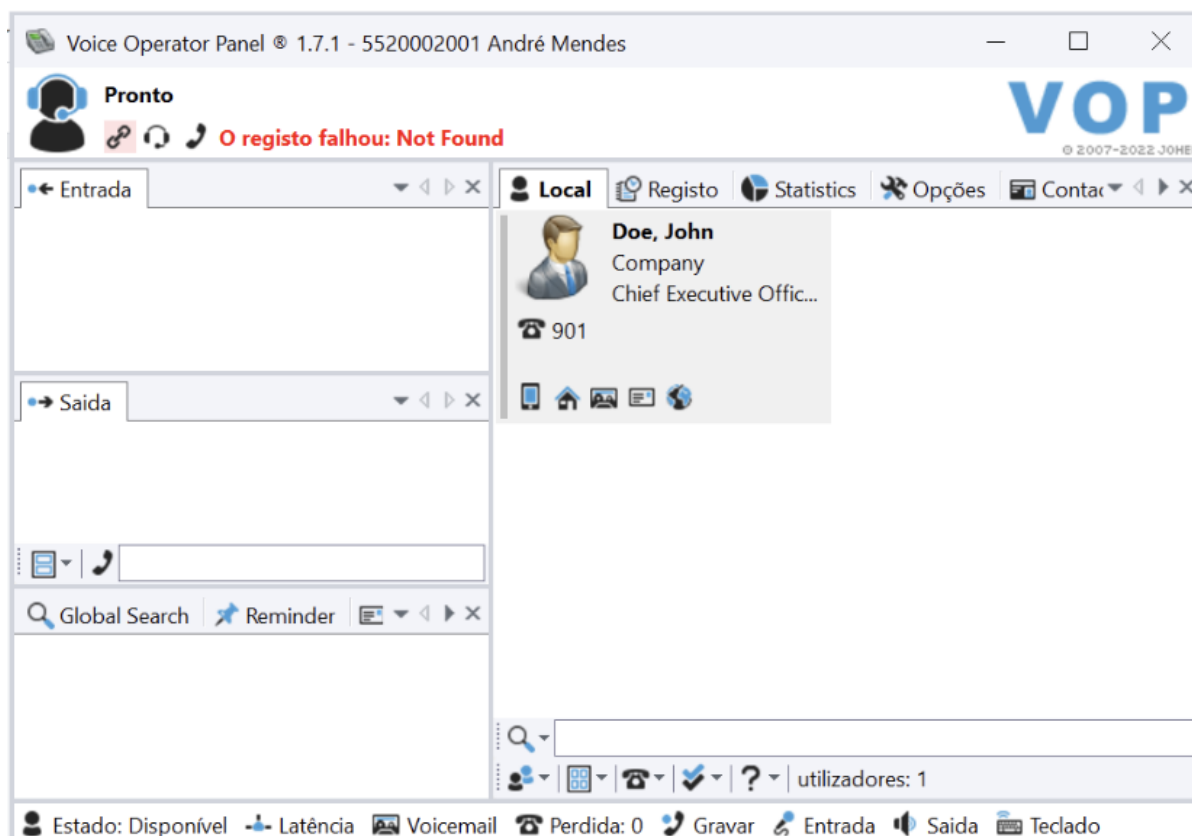


Figura 10: Interface da consola *Voice Operator Panel*, depois de instalada antes do registo no IP PBX

Após analisar a interface, ilustrada na Figura 10, depois de instalada e através das várias imagens disponibilizadas na página web, é possível ver que se trata de uma interface bastante desatualizada, com um *layout* estático e pouco responsivo e com uma paleta de cores que implica uma aparência monótona, o que resulta num desenho pouco apelativo. Além disso, a forma como os menus são apresentados ao utilizador, com um sistema hierárquico que implica um elevado número de cliques para efetuar certas tarefas, e certas funcionalidades que se revelaram complicadas de utilizar, como o teclado para digitar números e o *call park*, resultando numa usabilidade reduzida. No geral, a aplicação contrasta bastante com consolas de operador mais recentes, estudadas no Capítulo 2, com desenhos elegantes e intuitivos, não indo de encontro às expectativas nesta vertente. O mesmo se aplica em termos de funcionalidades, podendo ser comparada com a consola da Altice Labs, no sentido em que possui as funcionalidades básicas e cruciais a uma consola de operador, destacando-se em algumas, como por exemplo, *drag-and-drop* para transferir chamadas e a troca de mensagens, ambas não suportadas na solução da Altice Labs. No entanto, é de salientar que permanece como um produto relativamente inferior em relação a outros, como as soluções fornecidas pela 8x8 e RingCentral.

Relativamente à integração, esta produziu resultados satisfatórios, uma vez que foi possível iniciar, atender e desligar chamadas, funcionando estas como o esperado, assim como pausar e continuar (ver Figura 11).

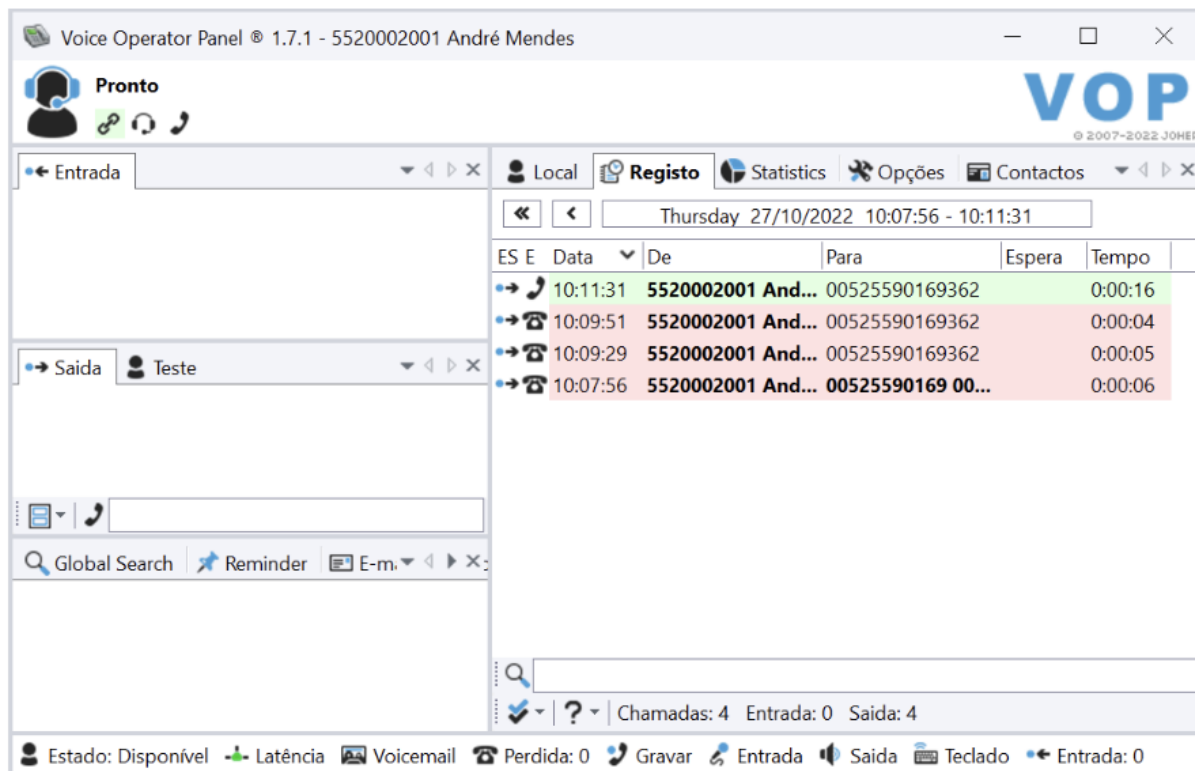


Figura 11: Chamadas efetuadas com sucesso na consola Voice Operator Panel

No entanto, outras funcionalidades tiveram problemas, como a transferência de chamadas e a monitorização do estado de contactos (ver Figura 12). Estes problemas foram analisados em detalhe, mas não foi possível concluir se aconteciam devido a problemas de compatibilidade ou devido a falhas no ambiente onde o **PBX** se encontrava hospedado.

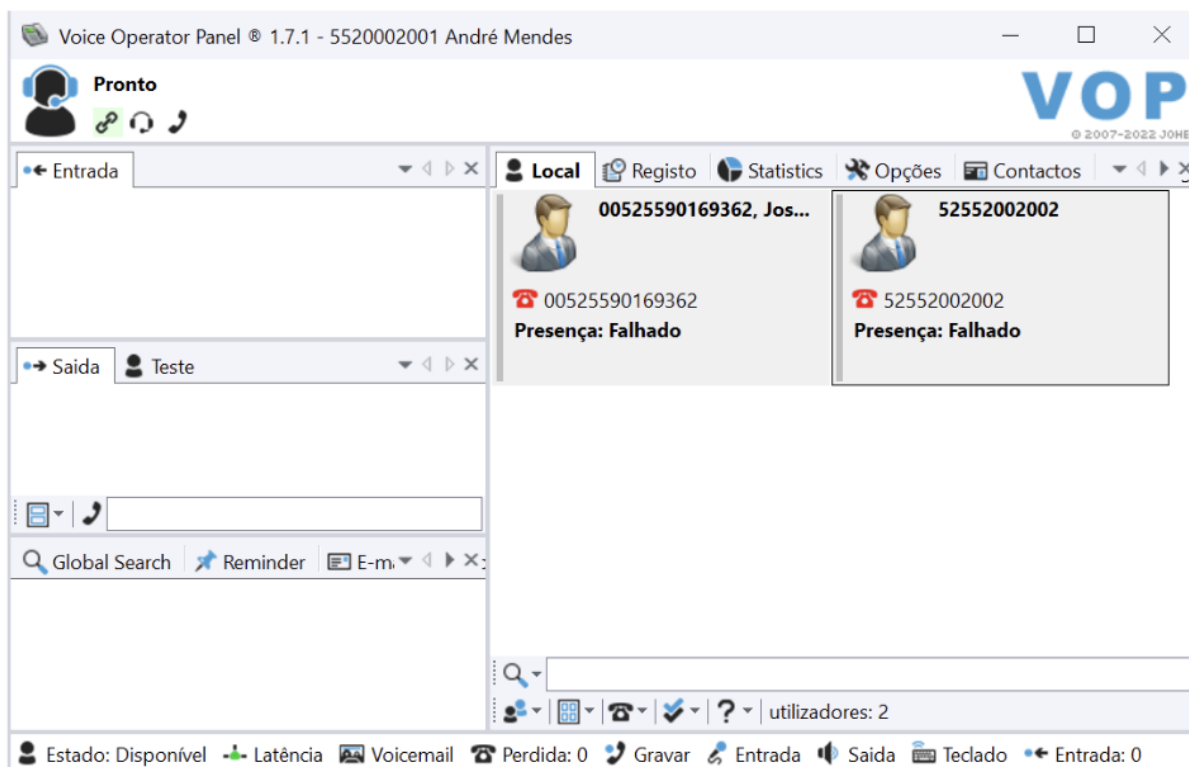


Figura 12: Erros encontrados em certas funcionalidades na consola *Voice Operator Panel*

3.1.4 Conclusões

No final da investigação realizada relativa à possibilidade de incorporar o serviço da Altice Labs com um cliente já desenvolvido de uma outra empresa, conclui-se que os resultados obtidos não foram satisfatórios o suficiente para que esta abordagem fosse considerada. Relativamente à consola Bitrix24, a integração revelou-se possível via REST **API**, mas no entanto, o comportamento da plataforma neste cenário não correspondia ao pretendido, uma vez que todas as funcionalidades de consola eram delegadas para o **PBX**. Assim sendo, não elimina a necessidade de um cliente que faça a gestão das chamadas, funcionando apenas como uma plataforma de consolidação de informação, o que bastou para descartar esta solução. Por outro lado, a incorporação da consola *Voice Operator Panel* revelou-se possível através de **SIP**, sendo possível testar uma versão parcialmente funcional da integração. No entanto, o produto não correspondeu às expectativas, possuindo uma interface antiquada e obsoleta e com uma lista de funcionalidades limitadora. Por estes motivos, conclui-se que esta consola não seria uma melhoria significativa relativa à consola atual, o que levou a que esta possibilidade fosse também abandonada.

Para concluir, o estudo feito não produziu resultados satisfatórios para que esta abordagem fosse seleccionada. Grande parte dos produtos estudados não especificam a possibilidade de integrações, e

quando o fazem, indicam que é necessário entrar em contacto para discutir os detalhes, algo que não era viável face ao tempo alocado para este estudo. Relativamente às plataformas onde foi possível verificar a integração, esta não foi de encontro às expectativas. Por estes motivos, foi necessário avançar com a única abordagem restante.

3.1.5 Abordagem escolhida

Por exclusão de partes, como visto na secção 3.1, a única abordagem restante, depois do estudo e posterior exclusão das restantes, diz respeito ao desenvolvimento de uma aplicação nova. Tendo em conta o âmbito do projeto e o tempo disponibilizado para o desenvolvimento, não era plausível que o desenvolvimento englobasse todas as vertentes de uma aplicação web. Apesar de ser possível, o resultado final estaria bastante longe das expectativas devido à falta de tempo para cada vertente e todas as implicações que isso acrescentaria ao trabalho.

Por esse motivo, foi decidido que seria melhor focar na implementação de um protótipo, neste caso, de um cliente web, utilizando certas **APIs** já em produção para implementação de um conjunto limitado de serviços / funcionalidades. Desta forma, todo o foco e o tempo estaria investido neste cliente web, o que permitiria a implementação de uma aplicação mais completa e robusta, ao invés de ser necessário dividir todo este esforço pelo cliente web e uma **API**.

3.2 Planeamento e Decisões

Tal como foi dito previamente, a escolha da abordagem selecionada teve sérias implicações devido às limitações impostas pelas **APIs**. Como foi decidido que seriam utilizadas algumas **APIs** já em produção, a lista de funcionalidades a serem implementadas no cliente web estão limitadas às operações suportadas pelas **APIs**, que neste caso, estabeleciam um limite significativo.

Posteriormente, numa fase mais avançada do desenvolvimento, foram tomadas decisões que permitissem contornar este obstáculo, no entanto, estas limitações não deixam de ter um impacto negativo no âmbito do trabalho a desenvolver.

Após uma leitura e estudo relativo à documentação interna da Altice Labs foi possível entender quais as operações suportadas e, consequentemente, as limitações impostas pelas duas **APIs** a utilizar:

- Computer Telephony Integration **API** - a *Computer Telephony Integration (CTI)* é uma **API** que permite a aplicações externas integrar o serviço *Advanced Business Communications(ABC)* - serviço de comunicações utilizado pela Altice Labs - para realizar operações relacionadas com telefonia e

para ser notificado acerca de eventos de telefonia de utilizadores. O problema com esta **API** é que as operações que esta suporta são bastante limitadas, nomeadamente:

- autenticação - esta **API** utiliza uma abordagem baseada em *tokens* e sessões para lidar com a autenticação. Ao enviar um pedido de autenticação, este é analisado internamente, retornando, caso efetuada com sucesso, o *JSON Web Token (JWT)* a ser enviado em pedidos subsequentes, para que seja garantida a devida autorização;
- iniciar chamada - este método é utilizado para iniciar uma chamada por um utilizador para um destino. Este pedido pode também ser efetuado caso o utilizador se encontre numa chamada, sendo que neste cenário, a chamada atual é colocada em pausa;
- limpar chamada - este método é utilizado para terminar uma chamada em progresso de um número específico associado ao utilizador autenticado;
- gerir gravação durante chamada - estas operações, compostas por 4 diferentes pedidos, são utilizadas para iniciar, pausar, continuar ou parar uma gravação de uma chamada;
- pedidos de subscrição a eventos - este método é utilizado por sistemas externos para indicarem que pretendem receber notificações relativas ao estado de um número associado a um utilizador, com o limite de uma subscrição por utilizador;
- eventos - estes eventos tratam-se de notificações assíncronas enviados pelo **ABC**, através de uma *websocket*, para informar aplicações externas acerca de atividades dos utilizadores. Estes eventos são utilizados em conjunto com o pedido de subscrição, sendo que após a subscrição, através da *websocket*, o sistema externo é notificado sempre que o estado do número subscrito é alterado.

Apesar de suportar diferentes operações, é possível ver que na área de telefonia, existe uma grande limitação naquilo que é suportado, sendo apenas possível iniciar e terminar chamadas. Apesar de ser possível, de forma indireta, pausar uma chamada iniciando outra quando a chamada a pausar se encontra em progresso, depois desta ser pausada, é impossível continuar a chamada, além de que esta forma de pausar não é ideal. Além disso, como esta operação não é diretamente suportada, não é recebida notificação acerca deste evento, o que torna impossível tornar a informação persistente.

- SelfCare **API** - esta **API** é utilizada para integrar as funcionalidades de contactos, incluindo as operações de listar, criar, editar e eliminar, possuindo quatro métodos diferentes para suportar

estas quatro operações. Além disso, como se trata de uma **API** completamente separada da anterior, é necessário também efetuar a autenticação, que funciona de forma semelhante.

3.2.1 Requisitos

O estudo incluído na secção anterior teve elevada importância uma vez que seria necessário entender o que era suportado pelas **APIs** para definir com exatidão o que se pretendia desenvolver. Para isso, foi utilizado o método priorização de *Moscow*, que permite priorizar requisitos em quatro categorias diferentes [88]:

- *must have* - requisitos que devem, sem exceções, fazer parte do projeto;
- *should have* - requisitos de alta prioridade, mas a sua ausência não compromete o projeto;
- *could have* - requisitos que são desejáveis mas não necessários;
- *won't have* - requisitos que não fazem parte desta fase do desenvolvimento, mas podem ser incluídos numa fase futura.

Tabela 2: Lista de requisitos ordenados pela priorização Moscow

Must Have	Should Have	Could Have	Won't Have
Iniciar Chamadas	Gravação de Chamadas	-	Reencaminhar Chamadas
Desligar / Rejeitar Chamadas	-	-	Transferir Chamadas
Histórico de Chamadas	-	-	Status de Presença
Diretório de Contactos	-	-	Captura de Chamadas
Autenticação	-	-	Pausar Chamadas
-	-	-	Atender Chamadas via Consola

Relativamente às funcionalidades *must have*, foram apenas colocadas aquelas que eram suportadas pelas **APIs** ou que não necessitavam de uma **API** para implementação. A gravação foi colocada na secção de *should have* uma vez que, apesar de ser possível a sua implementação, não é uma funcionalidade à qual foi atribuída uma prioridade estrita. Por último, todos os requisitos colocados na secção de *won't have* são funcionalidades que fazem parte da consola atual, e como tal, deviam ser incluídos no desenvolvimento, mas pelos motivos acima indicados, a sua implementação revelou-se como impossível.

3.2.2 Tecnologias a usar

Para o desenvolvimento de qualquer aplicação, selecionar as tecnologias adequadas é uma decisão de elevada importância que pode influenciar significativamente o sucesso e a eficiência do projeto. Mesmo neste cenário onde o objetivo passa pelo desenvolvimento de um protótipo, e não um produto final, é importante que esta seleção seja feita com elevado critério para que o processo de desenvolvimento e a qualidade do protótipo seja do mais alto nível possível.

Esta fase do planeamento teve como objetivo aprofundar e analisar a adoção de uma *framework* Javascript para o desenvolvimento deste trabalho em específico. Efetivamente, após analisar várias fontes que abordavam deste tema [89; 90], os fatores que mais influenciam esta escolha estão relacionados com a complexidade do trabalho, a experiência e a colaboração com a equipa, o suporte da comunidade, o desempenho, a curva de aprendizagem, a longevidade da *framework* e necessidade de customização.

A complexidade do projeto revela-se como sendo um dos fatores mais importantes no que leva a esta decisão, uma vez que a complexidade das aplicações web é algo bastante volátil, desde pequenas páginas com informação estática a aplicações *Single-Page* complexas.

O que é uma Single-page Application (SPA)?

Uma *Single-page Application* trata-se de uma aplicação web que apenas carrega um único documento web, sendo que o corpo deste documento é depois atualizado através de **APIs** Javascript ou outras **APIs** de forma a alterar o conteúdo [91]. Isto permite aos utilizadores interagirem com a aplicação sem ser necessário carregar páginas diferentes, implicando um ganho no desempenho da aplicação e uma experiência mais agradável para o utilizador.

Neste contexto, a aplicação a desenvolver encaixa, apesar de não totalmente, nesta definição, uma vez que o utilizador se encontrará maioritariamente na página principal da aplicação onde é realizada a gestão das chamadas, tratando-se de uma página altamente dinâmica devido ao elevado fluxo de chamadas, que implicam constantes alterações na interface. Existe apenas uma segunda página, utilizada apenas para autenticação, contendo um formulário para que o utilizador consiga inserir e submeter as suas credenciais.

Isto obviamente tem certas implicações, uma vez que **SPAs** requerem uma elevada interatividade, atualizações em tempo real e elevado processamento do lado do cliente, o que implica uma maior complexidade [92]. Tendo isto em conta, uma *framework* Javascript, com a capacidade de lidar com a gestão do estado da aplicação e o desenvolvimento de componentes para a interface do utilizador é capaz de melhorar significativamente a eficiência do processo de desenvolvimento para estes projetos mais complexos [93].

É também importante mencionar que uma *framework*, como Angular ou Vue, possuem um melhor desempenho no que diz respeito à atualização do conteúdo da página [94]. Como se prevê que, com um elevado fluxo de chamadas, implicando atualizações do estado e renderizações de modo a transmitir essa informação ao utilizador, alterar o conteúdo da página seja uma tarefa realizada frequentemente, a utilização de uma *framework* permitirá tirar vantagem do melhor desempenho para estas atualizações.

Por estas razões foi decidido que seria adotada uma *framework* Javascript para o desenvolvimento do cliente web. No entanto, com o rápido crescimento e evolução observado nas aplicações web, o número de *frameworks* tem vindo a aumentar exponencialmente, tornando a escolha da ferramenta mais adequada uma escolha igualmente importante.

Efetivamente, como neste cenário, a aplicação a desenvolver trata-se de uma aplicação externa, desenvolvida do início, foi dada total liberdade na escolha de qual *framework* a adotar para o seu desenvolvimento. Ao efetuar uma pesquisa aprofundada sobre este tema, foi considerado um estudo realizado anualmente, *State of JavaScript* [95], que trata de coletar dados sobre mais de vinte mil desenvolvedores de *software* com o intuito de identificar tendências atuais e futuras. Este estudo permite analisar, entre muita coisa, quais as *frameworks* mais utilizadas, o que permite ter uma ideia geral da preferência e nível de satisfação das mesmas (ver Figura 13).

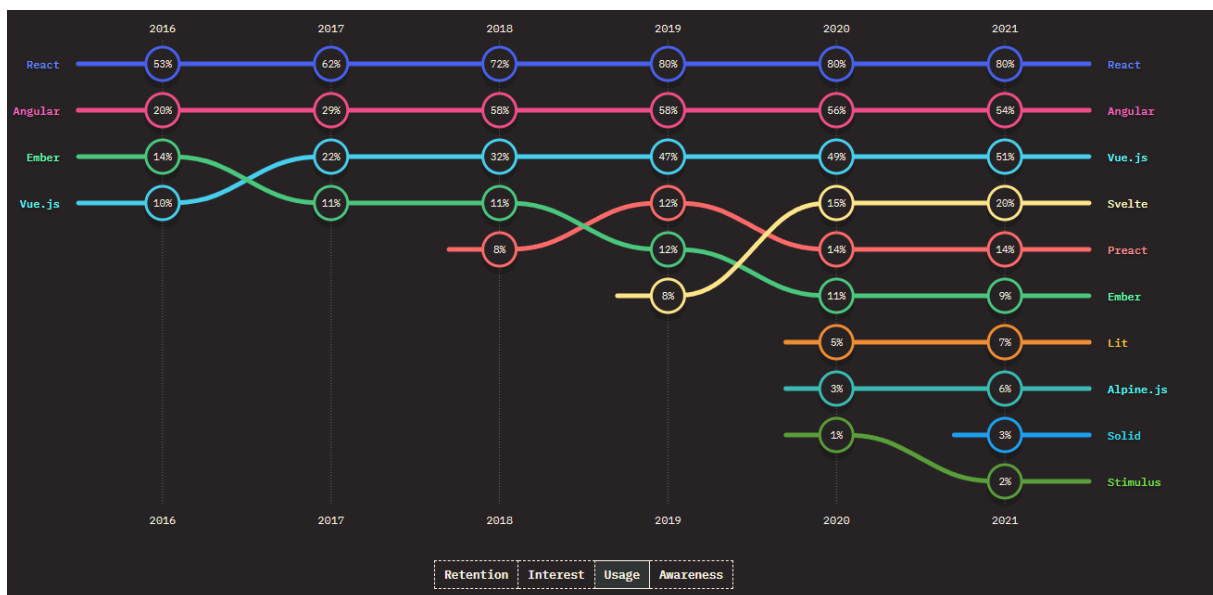


Figura 13: Taxa de utilização das *frameworks* Javascript

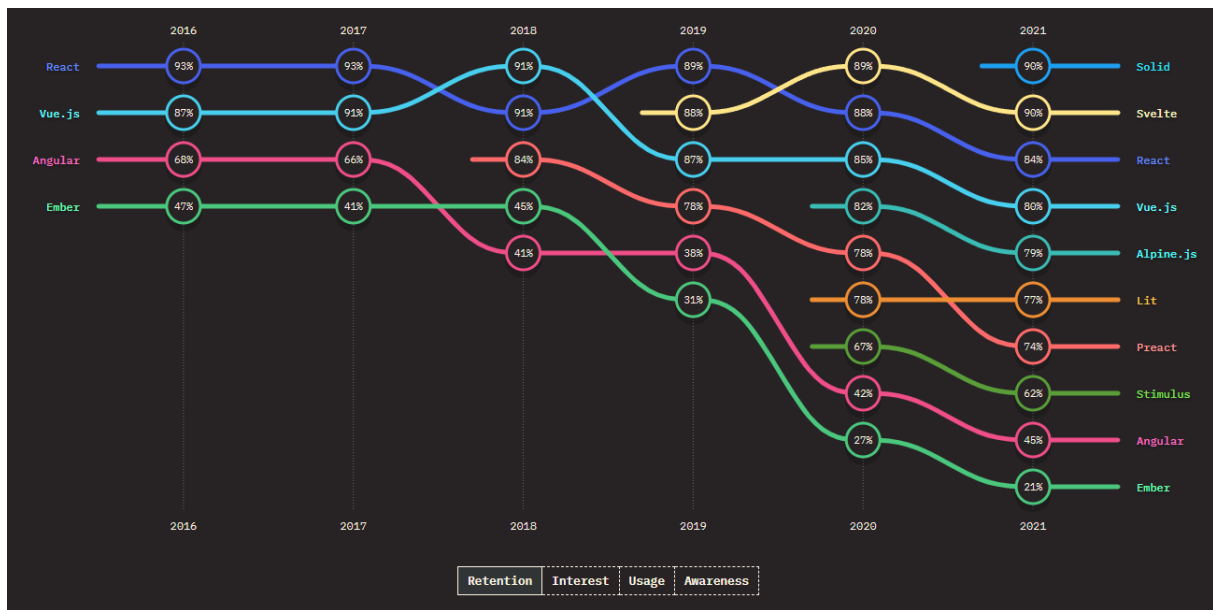


Figura 14: Taxa de retenção das *frameworks* Javascript

Estes dados foram retirados da pesquisa referente ao ano de 2021, uma vez que a mesma foi atrasada um ano [96]. Através das Figuras 13 e 14 é possível tirar algumas conclusões. Nomeadamente, React, Angular e Vue.js são, com uma margem confortável, as três *frameworks* mais utilizadas, sendo assim desde 2016, ano em que esta pesquisa foi iniciada. Relativamente à taxa de retenção, esta possui resultados diferentes, com duas ferramentas relativamente recentes, Solid e Svelte, a dominar nesta área, seguidas das mencionadas previamente.

Com esta pesquisa, apesar do número de desenvolvedores questionados ser relativamente baixo [97], correspondendo a menos de 1 por cento, é possível retirar uma ideia bastante genérica de quais ferramentas devem ser consideradas.

Como visto anteriormente, Solid e Svelte são as duas ferramentas que obtiveram os melhores resultados relativamente à retenção no ano de 2022. No entanto, é importante considerar vários aspetos, e um dos mais importantes é a longevidade da ferramenta. Como estas duas opções apenas ganharam popularidade nos últimos anos, possuindo, mesmo assim, baixos valores na área de utilização, é importante considerar que possuem uma comunidade mais pequena e como tal, sofrem bastante na área do suporte e recursos, uma vez que a comunidade é a fonte principal de documentação, tutoriais e pacotes externos.

Por esta razão, foi principalmente considerado o React, Angular e Vue.js. As suas qualidades e vantagens foram analisadas para que fosse possível tomar uma decisão o mais fundamentada possível. No final, os motivos que levaram à seleção incidiram, maioritariamente, na experiência da equipa de desen-

volvimento. Como todos os membros da equipa de desenvolvimento possuíam um amplo conhecimento de React, esta foi a ferramenta escolhida. Apesar de não ser uma *framework*, mas sim biblioteca, sendo esta uma escolha tão adequada para o trabalho a desenvolver como as outras, a experiência já adquirida pela equipa facilitou o desenvolvimento, uma vez que desta forma, não foi necessário uma fase de aprendizagem tão longa, sendo possível reservar mais tempo para o desenvolvimento.

Além disso, foi também decidido que com o React seria utilizado a linguagem Typescript. Esta linguagem funciona como uma extensão da sintaxe de Javascript que adiciona *static typing*, ou seja, permite a adição de tipos [98]. Esta adição tem como objetivo melhorar significativamente a manutenção e escalabilidade de código, ficando este mais fácil de ler, compreender e depurar. Apesar de escalabilidade e manutenção não ser uma preocupação de elevada prioridade para o desenvolvimento de um protótipo, Typescript é visto como uma linguagem bastante mais robusta e capaz de melhorar a fase de desenvolvimento, funcionando bem em conjunto com o paradigma de componentes utilizado pelo React que, apesar de introduzir uma curva de aprendizagem ligeiramente mais acentuada, a longo prazo torna-se vantajoso.

3.3 Arquitetura e Funcionamento

No contexto de desenvolvimento de software em constante evolução, as interfaces do utilizador possuem um papel fundamental ao garantir que o utilizador tem uma experiência positiva ao utilizar a aplicação. A interface serve como principal ponto de interação com utilizador, com o papel de o manter interessado, ditando a satisfação e o sucesso geral do produto. O objetivo desta secção passa por demonstrar os aspetos de desenho e implementação da aplicação com auxílio de diagramas para demonstrar o seu funcionamento e com o recurso a *mockups* para demonstrar o planeamento da interface.

Pelo que foi visto até agora, tendo em conta a abordagem selecionada e o estudo de *frameworks*, a aplicação a desenvolver possui a arquitetura de alto nível contida na Figura 15.

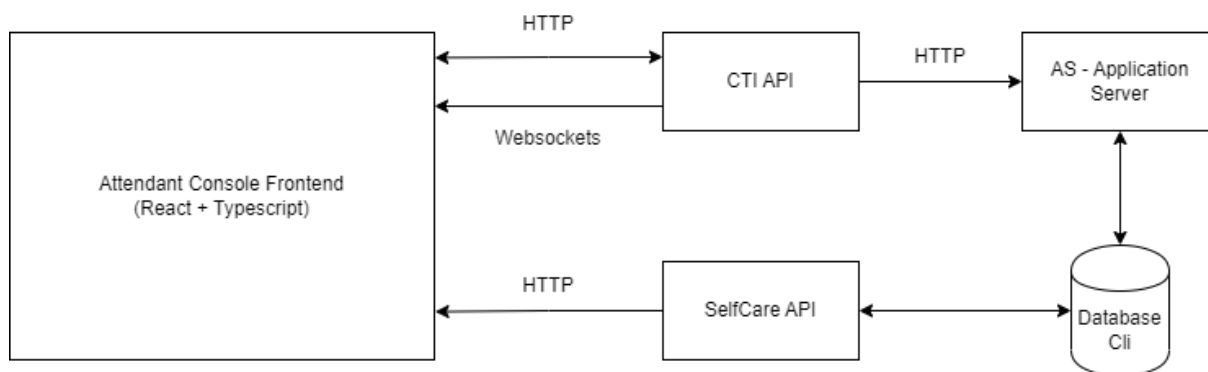


Figura 15: Arquitetura de alto nível da Consola de Operador

Como visto anteriormente, o cliente web irá comunicar com a **CTI API** e a **SelfCare API** de modo a implementar as funcionalidades de telefonia e contactos, respetivamente. Esta comunicação é feita através de HTTP, sendo que para as notificações dos eventos, uma vez que estas são enviadas de forma assíncrona pela **API** sempre que é feita uma alteração ao estado, a comunicação é feita através de uma *websocket*.

De seguida, é importante definir com cuidado o comportamento da aplicação. Para tal, foram elaborados fluxogramas que ilustram todas as tarefas realizadas pela aplicação e o comportamento típico do utilizador ao realizar as várias atividades.

Para tal, estes diagramas foram repartidos em três diferentes secções:

- autenticação - diagramas que apresentam as atividades relacionadas com a autenticação;
- chamadas - diagramas que apresentam as atividades relacionadas com as chamadas;
- contactos - diagramas que apresentam as atividades relacionadas com o diretório de contactos.

3.3.1 Autenticação

Referente a esta secção existe um único diagrama que regula o acesso do utilizador à consola (ver Figura 16). Quando isto acontece, a componente consola necessita de verificar se o utilizador já se encontra autenticado, e caso não, redirecioná-lo para a página de *login*. Depois disso, é necessário que o utilizador introduza as credenciais e proceda à submissão do formulário, o que implica enviar dois pedidos HTTP separados, um para cada **API**. Posteriormente, as respostas de sucesso a estes pedidos incluem todos os *tokens* e dados do utilizador necessários à utilização da consola, redirecionando o utilizador para a página principal. Após estas tarefas, inicialmente a consola passa por uma fase de carregamento, onde

é obtida a lista de contactos através de um pedido enviado para a SelfCare **API** e onde é efetuada a subscrição, enviando um pedido para a **CTI API**, para que as notificações dos eventos sejam enviados para a *websocket*. Assim que estas duas tarefas estejam realizadas, a fase de carregamento termina e o utilizador é capaz de utilizar a consola.

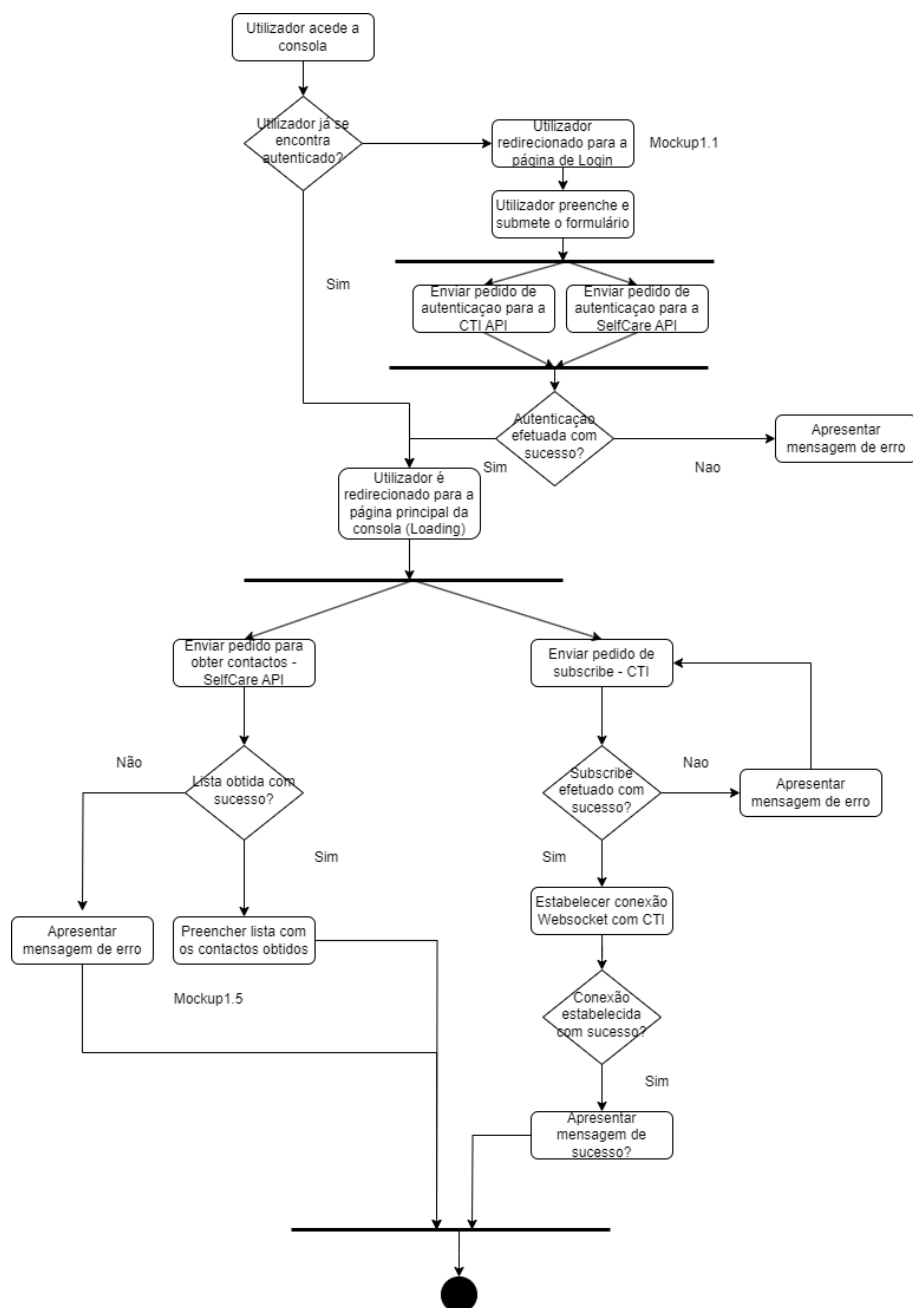


Figura 16: Diagrama de fluxo associado à Autenticação

3.3.2 Chamadas

Relativamente a esta secção, existe uma variedade de operações que é necessário abordar, incluindo todas as operações suportadas pela consola e alguns sub-fluxos, incluídos nos diagramas, relativos ao envio de notificações da **API** para a consola, através da *websocket*.

De acordo com a documentação interna da Altice Labs, como já foi estipulado, a **CTI API** permite o envio de notificações, utilizadas para, de forma assíncrona, notificar a consola acerca de atualizações do estado do número associado ao utilizador. Esta abordagem permite que a consola receba notificações relativas a chamadas iniciadas, recebidas ou terminadas, permitindo desta forma atualizar a interface para transmitir essa informação ao utilizador. Estas notificações podem ser de quatro tipos diferentes:

- Chamada Iniciada - enviada quando uma chamada é iniciada pelo operador;
- Chamada Alertada - enviada quando uma chamada é recebida, ou seja iniciada do utilizador para o operador;
- Chamada Atendida - enviada quando uma chamada é atendida;
- Chamada Terminada - enviada quando uma chamada é terminada.

Além disso, estas notificações permitem efetuar um tratamento de erros relativos a operações iniciadas via consola. Como visto anteriormente, é possível, através de diferentes pedidos HTTP, realizar certas operações em chamadas. No entanto, o comportamento observado da **API** nestes cenários revelou-se inconsistente, sendo que a resposta de sucesso enviada nestes pedidos nem sempre garantia que as operações eram concluídas, devido a erros de comunicação da **API** com o serviço. Para evitar estes acontecimentos, em qualquer operação iniciada pela consola, as notificações são utilizadas para averiguar se estas foram concluídas com sucesso, de forma a manter a informação presente na consola consistente.

3.3.2.1 - Notificação Chamada Iniciada

Inicialmente, o processo de iniciar chamada assume que o utilizador já se encontra autenticado, e que portanto todo o processo descrito na Figura 16 já se encontra terminado. É também importante notar que este processo pode ser iniciado de várias formas: através do teclado de marcação, através da lista de contactos ou através do histórico. De seguida, quando a chamada é iniciada pelo utilizador, é necessário enviar um pedido para a **CTI API** de forma a reencaminhar o pedido para o serviço, que trata de iniciar a chamada. Posteriormente, é necessário efetuar as verificações mencionadas no parágrafo acima,

analisando se foi recebida a confirmação de chamada iniciada através das notificações. Se for o caso, é necessário verificar se existem alguma chamada em progresso, colocando-a na área de chamadas pausadas, apresentando a nova chamada, ou, caso não seja recebida qualquer confirmação, apresentar uma mensagem de erro ao utilizador.

O diagrama da Figura 17 contém todo este fluxo de operações, incluindo um sub-fluxo associado a receção de notificação de Chamada Iniciada, sendo apenas necessário verificar se existe uma chamada em conversação antes de a adicionar.

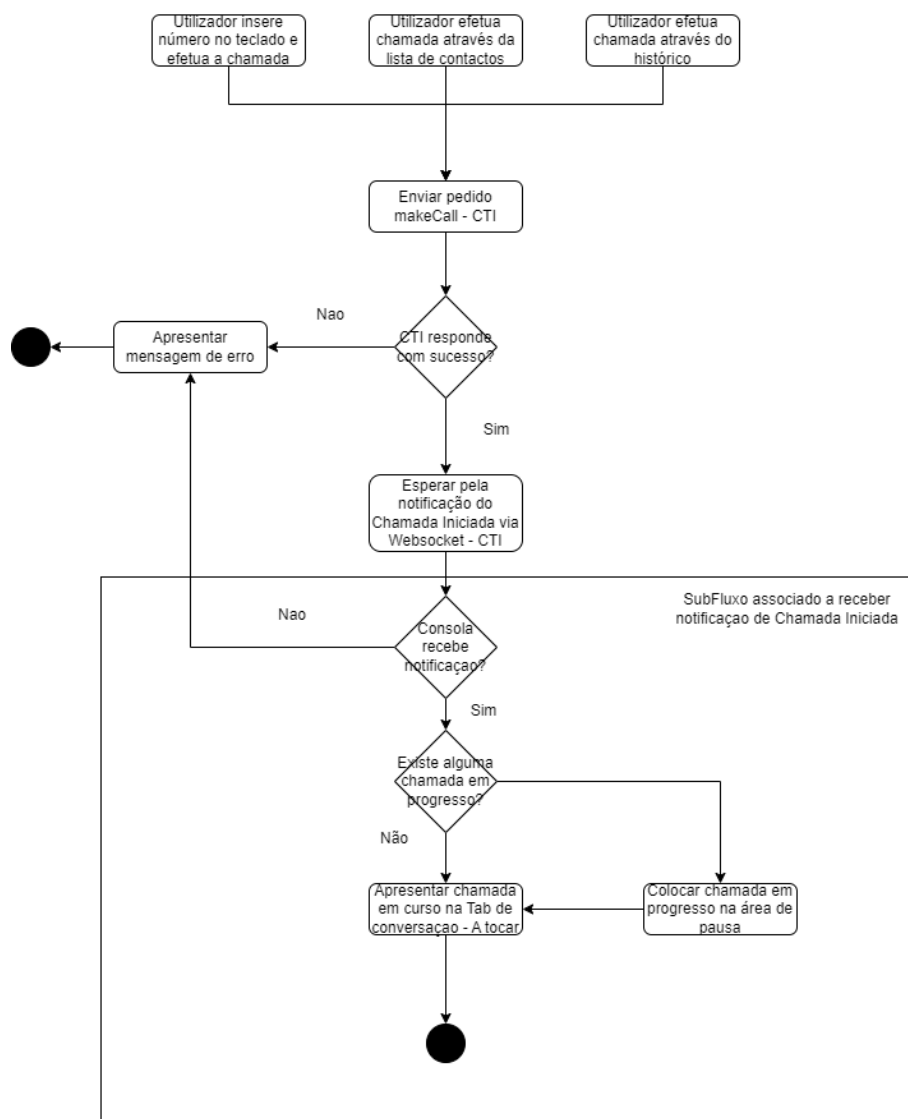


Figura 17: Diagrama de fluxo associado a iniciar chamada

Além do fluxo de operações descrito, é importante mencionar que este apenas diz respeito ao iniciar chamada. Depois disto, é também necessário um outro fluxo, que, apesar de estarem relacionados, o seu processamento é feito totalmente separado devido ao modo de funcionamento das **API** a integrar

para esta vertente.

3.3.2.2 - Notificação Chamada Atendida

Sempre que uma notificação de chamada atendida é recebida, é importante, de forma a evitar informação duplicada, procurar a chamada nas listas apresentadas ao utilizador. Como esta notificação indica que a chamada foi atendida, é possível assumir que esta já se encontra apresentada ao utilizador na área de chamadas a tocar ou na área de chamada em conversação, mas ainda a tocar. Como tal, é importante executar esta verificação para que esta chamada possa ser filtrada da lista, caso necessário, e alterar o seu estado. É necessário também prestar atenção a casos mais específicos, como o cenário em que uma chamada é atendida, mas com uma chamada diferente em progresso. Como indicado previamente, sempre que este cenário específico acontece, a chamada em progresso é colocada em pausa, de forma a colocar a nova chamada em conversação. Este é um processo que não recebe qualquer notificação ou confirmação por parte da **API**, uma vez que a operação de pausar não é suportada por esta. No entanto, apesar de não ser recebida qualquer notificação de confirmação, acaba por não ter impacto, uma vez que, assim que uma chamada é pausada não pode ser retomada, devido à falta dos métodos que implementem estas operações. O comportamento notado indica que qualquer chamada colocada em pausa desta forma é terminada pouco tempo depois pela **API**.

O diagrama da Figura 18 descreve este processo:

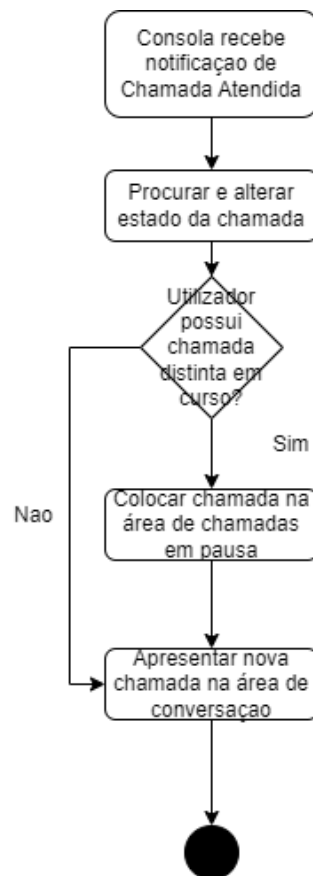


Figura 18: Diagrama de fluxo associado a Chamada Atendida

É importante mencionar que, apesar da consola não suportar, devido à falta de métodos nas **APIs**, a funcionalidade de atender chamada, o utilizador pode, de qualquer das formas, atender uma chamada recebida através do *softphone*, servindo este diagrama para representar essa ocorrência.

3.3.2.3 - Notificação Chamada Recebida

O diagrama da Figura 19 diz respeito ao processo associado sempre que uma notificação de chamada recebida é enviada para a *websocket*. Nestes casos, esta notificação indica sempre que uma chamada foi realizada de um utilizador para o operador com sessão efetuada na consola, e portanto basta adicionar essa chamada à lista de chamadas a tocar.

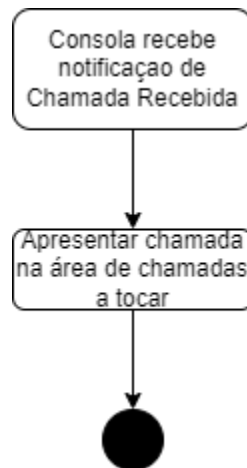


Figura 19: Diagrama de fluxo associado a Chamada Recebida

3.3.2.4 - Notificação Chamada Terminada

À semelhança do processo de iniciar chamada, é necessário ter certos pontos em consideração quando é feito o tratamento de um pedido de terminar chamada. Como esta operação pode também ser iniciada através da consola via **CTI API**, é preciso ter o devido tratamento de erros e verificar se é recebida a notificação de confirmação adequada. Inicialmente, esta operação pode ser iniciada através da área de conversação, área de chamadas a tocar ou área de chamadas pausadas, além do *softphone* utilizado. Posteriormente ao pedido ser enviado, é necessário esperar pela notificação de chamada terminada para que esta possa ser removida da área onde se encontra. Caso, ao final de alguns segundos, não seja recebida qualquer notificação de chamada terminada relativa à chamada em questão, é necessário apresentar uma mensagem de erro ao utilizador. Por último, estando a chamada terminada, esta é adicionada ao histórico do utilizador.

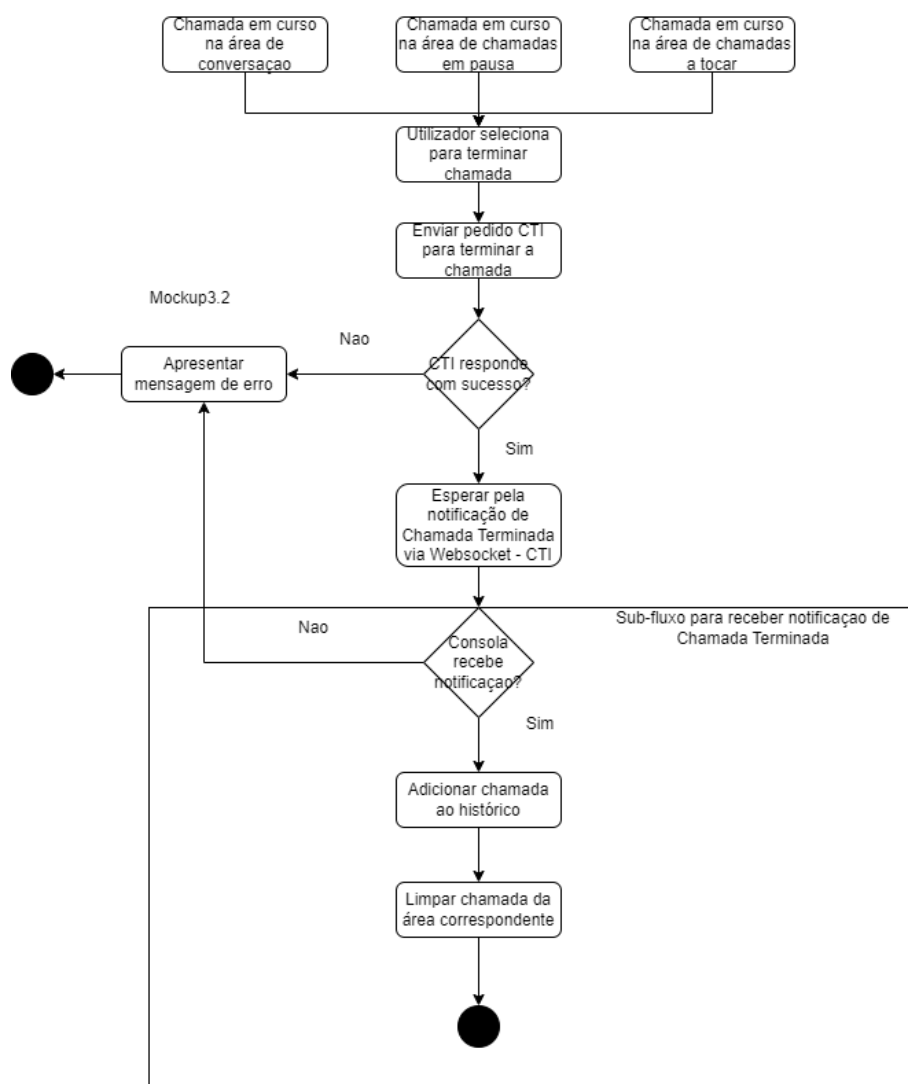


Figura 20: Diagrama de fluxo associado a terminar chamada

Como é possível ver através do diagrama de fluxo da Figura 20, este processo de terminar chamada contém um sub-fluxo associado a receber uma notificação de chamada terminada, uma vez que as chamadas podem ser terminadas pelo *softphone* ou pelo outro participante, o que implica terminar a chamada sem qualquer interação prévia do utilizador na consola.

3.3.3 Contactos

No que diz respeito aos contactos, os processos associados às operações possíveis são relativamente simples. São suportadas as operações que permitem fazer a gestão completa, que inclui listar, editar e, caso sejam, contactos pessoais, criação e eliminação destes (o tipo de contactos, incluindo pessoais, são explicados na secção 3.4.2.3). Além destas operações, é importante mencionar que uma alteração

nos filtros implica também uma nova listagem, assim como uma eliminação ou criação de um contacto, uma vez que estas operações tem implicações na ordem da lista, que podem alterar a página atualmente apresentada ao operador.

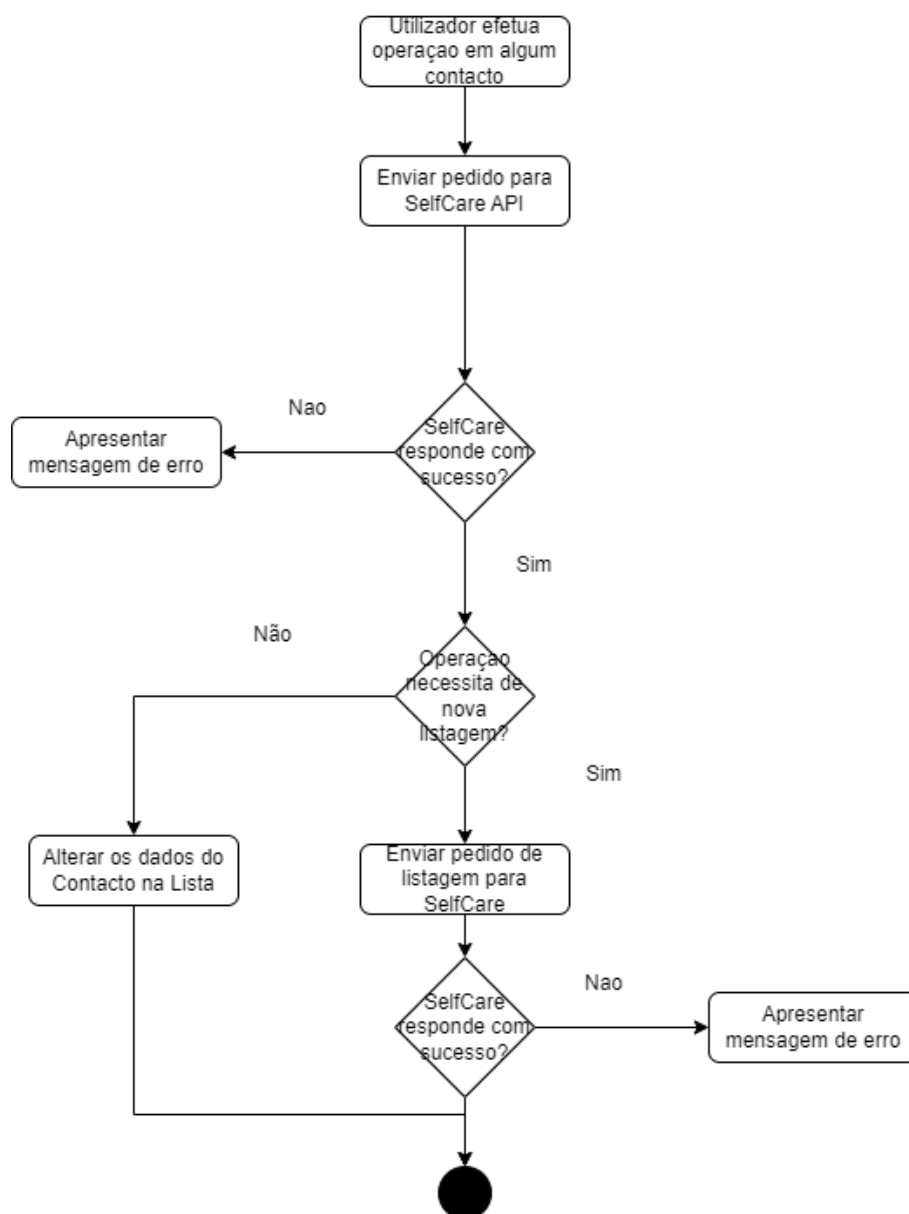


Figura 21: Diagrama de fluxo associado a uma operação nos contactos

O diagrama da Figura 21 diz respeito a uma operação de listagem, edição, criação ou eliminação de algum contacto, podendo ser necessário, como indicado, o envio de um pedido para obter a nova lista através da SelfCare **API**, caso a operação possa ter algum impacto na forma como estes são apresentados.

3.4 Desenho da Interface do Utilizador

No que diz respeito ao desenvolvimento de software, e neste caso, ao desenvolvimento de uma aplicação com foco exclusivo na parte do cliente, a criação de uma interface intuitiva e *user-friendly* tem um papel crucial, sendo outra vertente que tem peso suficiente para determinar o sucesso ou a falha de uma aplicação. Provas disso são os inúmeros acidentes que foram causados, mesmo que apenas parcialmente, por um mau desenho de interfaces do utilizador. Um bom exemplo disso é o voo *Iran Air 655*. Este voo comercial foi incorretamente abatido por um míssil antiaéreo disparado pelo navio *USS Vincennes* dos Estados Unidos, resultado na morte de 290 pessoas. Aos diferentes motivos atribuídos ao acidente, um dos fatores revelou-se como sendo um *desenho* pobre, o que levou a dificuldades ao visualizar os diferentes dados do voo, como trajetória, velocidade e altitude, resultando numa decisão errada que levou ao seu abate [99].

Como é óbvio, este exemplo, e outros, compõe situações onde problemas de *User Interface (UI)* e *User Experience (UX)* escalam a situações dramáticas e catastróficas podendo, em casos mais extremos, colocar em risco a vida de pessoas. Apesar disso, em situações onde tal escalamento não é previsto como possível, o desenho da **UI** deverá ter sempre uma elevada prioridade, uma vez que problemas pode levar a graves consequências para a aplicação e consequentemente para o negócio.

Uma interface de utilizador pobre pode implicar uma taxa maior de erros, maiores custos de treino, e um abandono maior por parte dos utilizadores ou clientes. Relativamente ao negócio, isto traz como consequências um maior custo e um menor lucro, resultados que se pretendem evitar [99; 100; 101; 102].

Além disso, uma interface bem planeada e desenvolvida, com foco nos utilizadores, pode ter um forte impacto na produtividade dos operadores que possam, no futuro, utilizar uma versão deste protótipo mais avançada, assim como outras vantagens claras para o negócio, como uma maior retenção de clientes, menor custos de treino e manutenção [99; 103]

Por estes motivos, foi dada elevada importância a este processo de planeamento associado à interface da aplicação. Para isso, foi utilizada maioritariamente uma abordagem orientada a objetivos [104], sendo que esta metodologia baseia-se em, através de uma análise cuidada dos objetivos e das intenções do utilizador final, entender a essência das interações do utilizador e, desta forma, implementar produtos de alta qualidade adequados a estes.

De modo a alcançar este objetivo, devido à experiência de certos membros da equipa com produtos semelhantes, foi utilizada uma ferramenta de prototipagem onde, de forma iterativa, os diferentes desenhos foram propostos e avaliados de acordo com o que foi explicado acima. A ferramenta de prototipagem

utilizada foi o Figma [104], devido à experiência previamente adquirida nesta plataforma, o que permitiu acelerar o processo de desenvolvimento.

Como foi indicado previamente, o objetivo desta aplicação passa por ser uma **SPA**, o que permite tirar máximo partido do seu comportamento dinâmico e das suas interações e atualizações em tempo real, com a exceção de uma página de autenticação, colocada à parte. Como tal, o espaço para cada componente teve que ser cuidadosamente planeado de forma a ser o mais intuitivo e *user-friendly*. Foi também analisado, dentro dos possíveis, a interface de vários produtos semelhantes previamente analisados (8x8 [105], RingCentral [106], Vodafone One Net [107], Imagicle [108], Voice Operator Panel [109]) e outras que, por motivos de redundância, não foram abordadas neste documento, de forma a servir de inspiração e analisar como certos problemas foram abordados e resolvidos.

Foi também necessário prestar especial atenção às cores utilizadas em todos os ecrãs, uma vez que estas tem que estar em conformidade com a paleta de cores utilizada e aprovada pela Altice Labs. No entanto, isto foi apenas efetuado numa fase mais avançada do planeamento, sendo que alguns ecrãs podem não conter as cores corretas. Além disso, vários destes ecrãs foram sujeitos a alterações durante a fase de desenvolvimento, que serão mais à frente indicadas e justificadas.

3.4.1 Página de Autenticação



The image shows a login form mockup. At the top center is the 'altice labs' logo, consisting of a stylized 'a' icon followed by the text 'altice' and 'labs' below it. Below the logo is the text 'Iniciar Sessão'. Underneath this are two input fields: the first is labeled 'Email' and the second is labeled 'Password'. The 'Password' field has a small eye icon to its right, indicating a toggle for visibility. At the bottom of the form is a black rectangular button with the white text 'Entrar'.

Figura 22: Mockup da Página de Autenticação

Como é possível ver através da Figura 22, a página de autenticação, como possui uma página inteiramente dedicada, é relativamente simples, com um único formulário para que o utilizador consiga introduzir as credenciais e um simples botão, com uma animação de carregar, de modo a fornecer algum *feedback* ao utilizador enquanto a autenticação é realizada. Isto permite indicar que, como é necessário autenticar em duas **APIs** diferentes, que apesar de ser um processo que pode ser demorado, já se encontra a decorrer, de forma a evitar a submissão de vários formulários.

3.4.2 Overlay da Aplicação

O *overlay* geral da aplicação é bastante importante, uma vez que dita como os diferentes menus e as diferentes áreas são apresentadas no ecrã, o que influencia bastante como a informação é apresentada ao operador e os fluxos necessários para que tarefas que incluam diferentes áreas sejam realizadas.

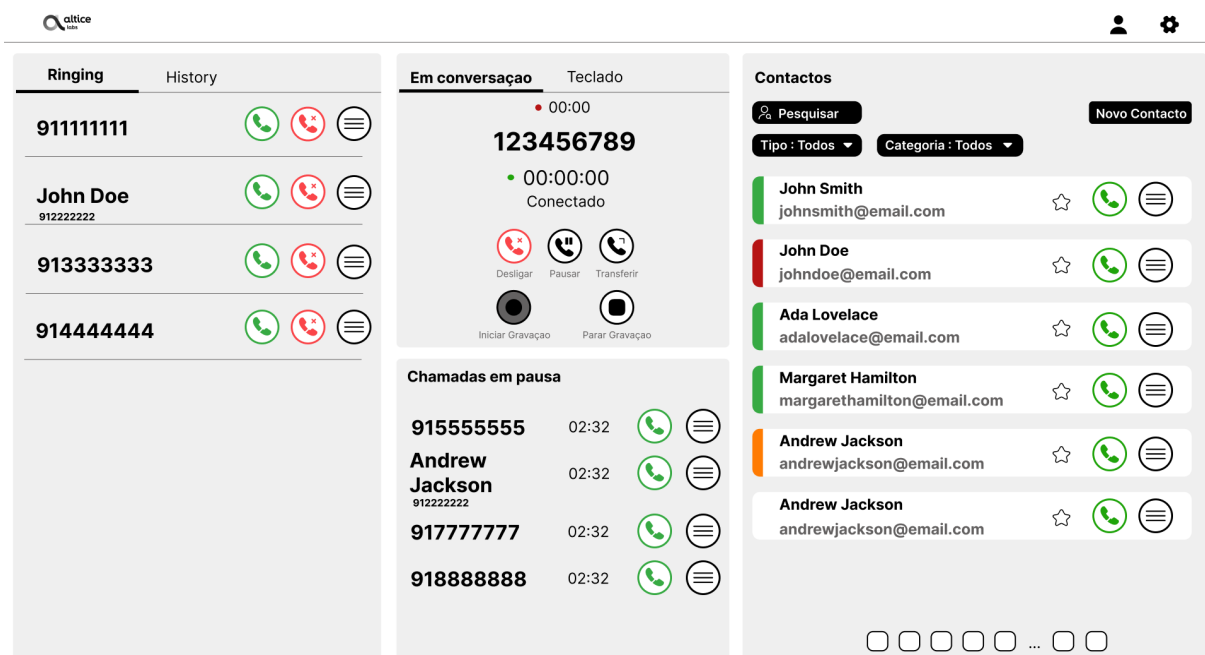


Figura 23: Mockup geral associada à disposição dos menus

No geral, o *layout* apresentado na Figura 22, que se trata da página principal da aplicação, pode ser dividida em três áreas:

- área de chamadas a tocar;
- área de chamadas em progresso;
- área de contactos.

Foi escolhida uma disposição vertical para os menus uma vez que, depois de várias iterações, esta foi a que demonstrou ter uma apresentação mais clara e mais intuitiva da informação e do fluxo das operações suportadas pelas **APIs**. Inicialmente, a área das chamadas a tocar apresenta todas as chamadas recebidas pelo operador. Esta área foi colocada do lado esquerdo, de forma a que existisse mais espaço para apresentar o máximo de chamadas possíveis sem a necessidade de interagir com uma barra de *scroll* ou paginação, e como as chamadas vão chegando sequencialmente à medida que o utilizador as recebe, não tem qualquer impacto na carga da **API**.

Foram utilizados *tabs* para que o utilizador conseguisse alternar a informação presente nesta área e em outras, dando a possibilidade de alterar a informação apresentada para o histórico, que será mais à frente explicado. Desta forma, à semelhança da listagem de chamadas a tocar, é obtido mais espaço para que, caso este pretenda procurar um registo específico, seja mais fácil de o visualizar e, com o recurso a *tabs*, esconder informação uma vez que se prevê que o histórico seja apenas utilizado esporadicamente.

Imediatamente ao lado foi colocada a área de chamadas em progresso. Devido ao espaço ocupado por estas, esta área foi dividida em duas sub-áreas, sendo colocada a área de chamada em conversação e de chamadas pausadas. Foram novamente utilizados *tabs* para que o utilizador conseguisse aceder ao teclado de marcação, onde é possível digitar um número e iniciar chamada.

Por último, a área de contactos foi também colocada em modo vertical. Esta decisão vai em contraste com o que foi observado em vários outros produtos semelhantes, mas uma vez que o serviço não suporta a inclusão de imagens para cada contacto, uma lista vertical revelou-se como sendo a forma mais compacta e clara de apresentar a informação. Uma grande dificuldade encontrada nesta área foi o mecanismo de filtragem, uma vez que a lista vertical dificultou a sua apresentação no ecrã, o que implicou uma fase de planeamento mais demorada e rigorosa relativa a esta vertente.

Existe também a barra de navegação, colocada no topo da página, onde o utilizador poderá aceder ao perfil autenticado e efetuar algumas operações.

3.4.2.1 - Área de Chamadas a Tocar

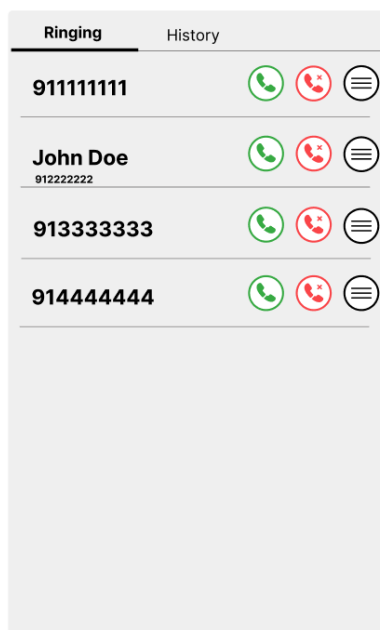


Figura 24: Mockup associada a área de Chamadas a Tocar

Nesta área, apresentada na Figura 24, cada chamada é apresentada ao utilizador com a informação considerada relevante. Neste caso foi decidido que apenas seria apresentado o número e o nome do utilizador, caso este esteja guardado nos contactos. Depois disso, do lado direito são apresentadas as operações possíveis, sendo estas atender ou rejeitar a chamada, e um botão que permite abrir um menu de contexto com outras operações menos utilizadas, como guardar nos contactos. Apesar das cores indicadas nos botões não irem de encontro à paleta de cores da Altice Labs, após algumas iterações foi decidido que seria melhor ir de acordo com o que é esperado pelos utilizadores, que é neste caso, a cor verde para o botão de atender e a cor vermelha para o botão de rejeitar. O uso de cores diferentes, apesar de tornar a interface bastante mais uniforme com um espetro de cores em conformidade com o resto da aplicação, poderia ser confuso para os utilizadores que, diariamente, utilizam um telemóvel e que, portanto, estão habituados a cores específicas, como o vermelho e o verde, para efetuarem operações de atender e desligar chamadas.

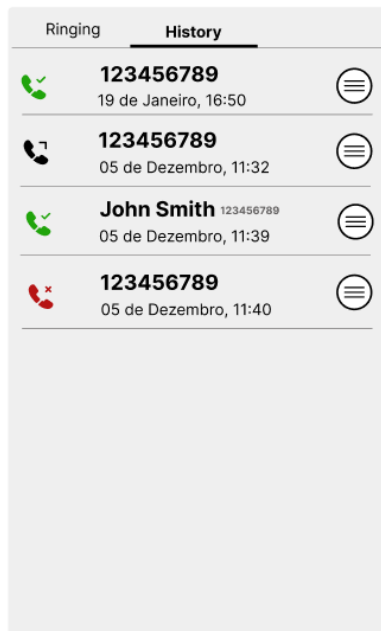


Figura 25: Mockup associada a área de Chamadas a Tocar com o Histórico selecionado

Ao aceder ao histórico através do *tab* correspondente, é apresentado o registo de chamadas em que o utilizador esteve envolvido (ver Figura 25). A informação presente nesta lista é também bastante sucinta, contendo apenas o número ou nome associado ao número, caso este pertença aos contactos, assim como o estado da chamada, podendo estar ter sido:

- efetuada e atendida;
- efetuada e não atendida;
- recebida e atendida;
- recebida e não atendida.

Este estado é indicado através do ícone e da cor utilizada. Além disso, é também indicado a data da chamada, mais especificamente, quando esta foi terminada, e um botão que, à semelhança do *tab* de chamadas a tocar, contém outras operações, nesta fase não definidas.

3.4.2.2 - Área de Chamadas em Progresso



Figura 26: Mockup associada a área de Chamadas em Progresso

Apesar da funcionalidade de pausar e retomar uma chamada não estarem totalmente implementadas nas **APIs**, é importante ter esta área em consideração sendo assim possível planejar com antecedência o funcionamento de todas as áreas da consola.

Esta área, ilustrada na Figura 26, tem como objetivo apresentar todas as chamadas em progresso, ou seja, a chamada em conversação e as chamadas que se encontram pausadas. Estas são apresentadas ao utilizador na mesma área pois, desta forma, é mais intuitivo compreender em quais chamadas o operador se encontra. Ao colocar os dois menus, separados, um em cima do outro, fornece um fluxo de informação simples e claro, sendo que as chamadas passam para o menu de baixo quando são pausadas e para o menu de cima quando são retomadas.



Figura 27: Mockup associada ao teclado de marcação

Foram novamente utilizados *tabs* de forma a que o utilizador consiga aceder ao teclado de marcação (ver Figura 27). É a partir daqui que o operador pode iniciar uma chamada para um número inserido, quando este não se encontra presente na consola sobre outra forma (contacto ou histórico). É também fornecida a possibilidade de digitar o número utilizando os botões que representam o teclado, caso o operador prefira a ter que recorrer ao teclado físico do computador ou do dispositivo que estiver a utilizar.

Caso o *tab* de conversação esteja seleccionada, é apresentada ao utilizador a informação da chamada em conversação, sendo importante apresentar o estado da chamada (a tocar ou conectada), assim como a duração, o número do utilizador e, por último, todas as ações possíveis, utilizando novamente a mesma lógica para as funcionalidades que não serão ou poderão não ser implementadas (ver Figura 28).



Figura 28: Mockup associada à chamada em conversação no estado A Chamar

Relativamente à área de chamadas pausadas, esta funciona de forma semelhante à área de chamadas a tocar, sendo apresentado o número ou nome do utilizador remoto, o tempo total da chamada, e as ações

possíveis.

3.4.2.3 - Área de Contactos

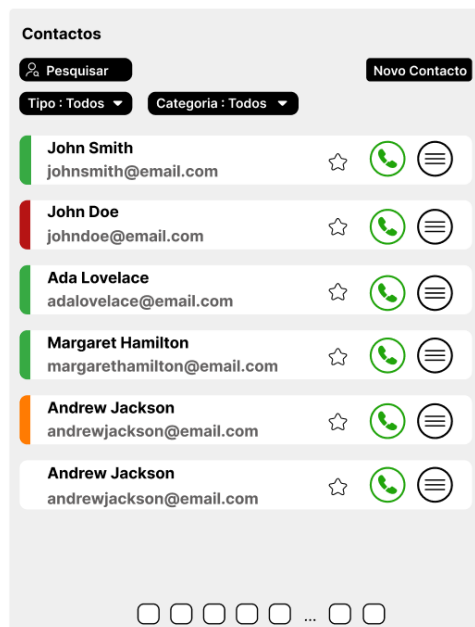


Figura 29: Mockup associada à área de Contactos

A área de Contactos, ilustrada na Figura 29, revelou-se como sendo a área mais complexa devido a todos os detalhes e as diferentes operações suportadas. Primeiramente, é necessário apresentar, no topo desta área, as diferentes opções de listagem da lista de contactos, nomeadamente, uma barra de pesquisa e os dois diferentes tipos de filtros, relativos ao tipo:

- pessoais - contactos pessoais criados e geridos pelo utilizador;
- corporativos internos - contactos internos, geridos pela empresa, que o utilizador apenas pode adicionar/remover dos favoritos e adicionar notas;
- corporativos externos - contactos externos, geridos pela empresa, que o utilizador apenas pode adicionar/remover dos favoritos e adicionar notas.

e relativos à categoria, tendo esta como opções:

- todos;
- monitorados - contactos que o operador tem acesso ao estado, explicados na secção 2.2.

Além disso, foi também adicionado na mesma posição, ao lado da barra de pesquisa, o botão de adicionar contacto, permitindo abrir um formulário de criação de forma a que o utilizador consiga adicionar um novo contacto pessoal à lista. Colocando-o no lado posto barra de pesquisa permite que este fique visível para o utilizador, mas ao mesmo tempo afastado da área dos filtros, exceto se o utilizador utilizar o browser numa resolução bastante pequena, que será abordado mais à frente.

Posteriormente, os contactos são apresentados ao operador com recurso à paginação, contendo informação acerca do nome e email, assim como um ícone de estrela, funcionando como um botão para indicar se pertence para à lista de favoritos e para adicionar ou remover o contacto dessa lista. Além disso, é apresentado um botão de iniciar chamada para o contacto, podendo este iniciar chamada de imediato, ou abrir um pequeno menu de contexto com as várias opções, caso possua mais do que um número (consultar Figura 30), e um botão final para operações adicionais, como editar e eliminar, caso este contacto pertença aos contactos pessoais. No início de cada contacto é possível ver uma pequena barra com diferentes cores: verde, vermelho e laranja. Esta serve para representar o estado de presença do contacto, neste caso, disponível, ocupado e a tocar, respetivamente, sendo que esta barra pode ser inexistente caso o contacto não seja monitorado. Deste modo, é possível apresentar o estado do contacto sem ser necessário alterar um filtro ou alterar para uma área diferente, como é o caso da consola atual da Altice Labs, facilitando assim a consulta.

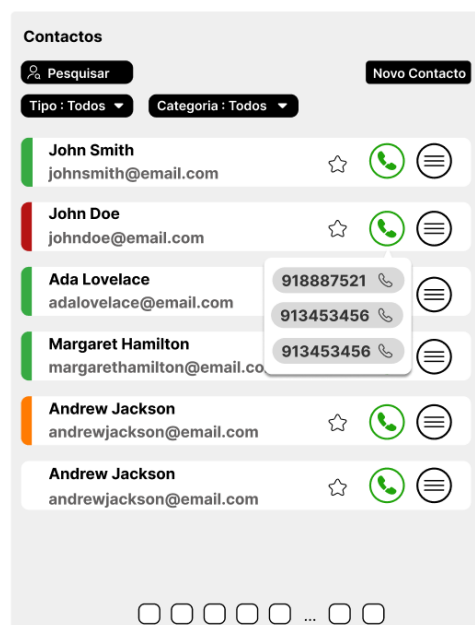


Figura 30: Mockup associada à área de Contactos, com os vários números associados a um contacto

Além disso, cada contacto pode ser expandido quando seleccionado para revelar mais informação,

como ilustrado na Figura 31. Neste modo, o botão de chamada é substituído por um botão dedicado a cada número, colocado no final do componente, revelando-os sem qualquer interação adicional necessária. Além disso, é também revelado o tipo do contacto e as notas associadas, que podem ser editadas independentemente do tipo de contacto. O estado de presença passa também para o lado das notas, sendo utilizado um ícone e uma breve descrição.

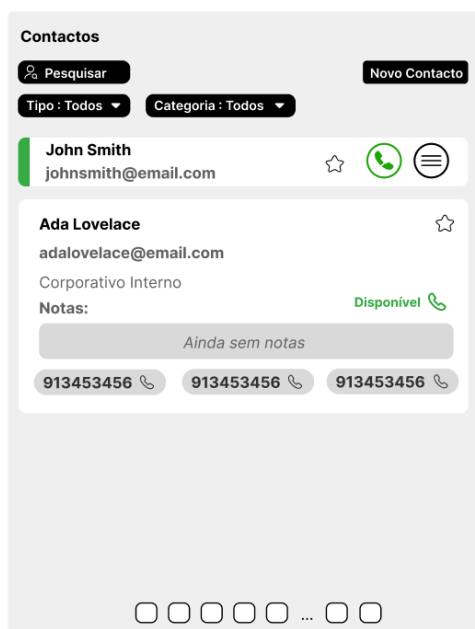


Figura 31: Mockup associada à área de Contactos com contacto expandido

3.4.3 Área de Help Online

Para esta fase do planeamento foi também decidido que deveria ser incluído o *Help* Online (ver Figura 32), de forma a facilitar a sua implementação futura. Este *Help* Online tem como objetivo fornecer ajuda a operadores inexperientes com este produto ou como área de esclarecimento de dúvidas facilmente acessível, sucinta e apelativa aos utilizadores.

Durante um longo período de tempo, a forma geral de fornecer ajuda a um produto digital resumia-se em longas páginas de documentação que funcionavam como manuais e através de áreas de *Frequent Asked Questions* (**FAQs**). No entanto, uma breve pesquisa rapidamente revela que este método não gera bons resultados, uma vez que, não só não é um método apelativo para os utilizadores [110], como também frequentemente é incapaz de resolver os problemas e as dúvidas destes [111; 112].

De modo a evitar isso, foi criado um pequeno *tour* da aplicação, acionado pelo utilizador, com recurso a uma modal, permitindo a apresentação na mesma página, e com paginação respetiva de modo a

fornecer algum contexto relativo ao progresso do utilizador durante o tutorial, na tentativa de aumentar ao máximo a taxa de conclusão [113; 114].

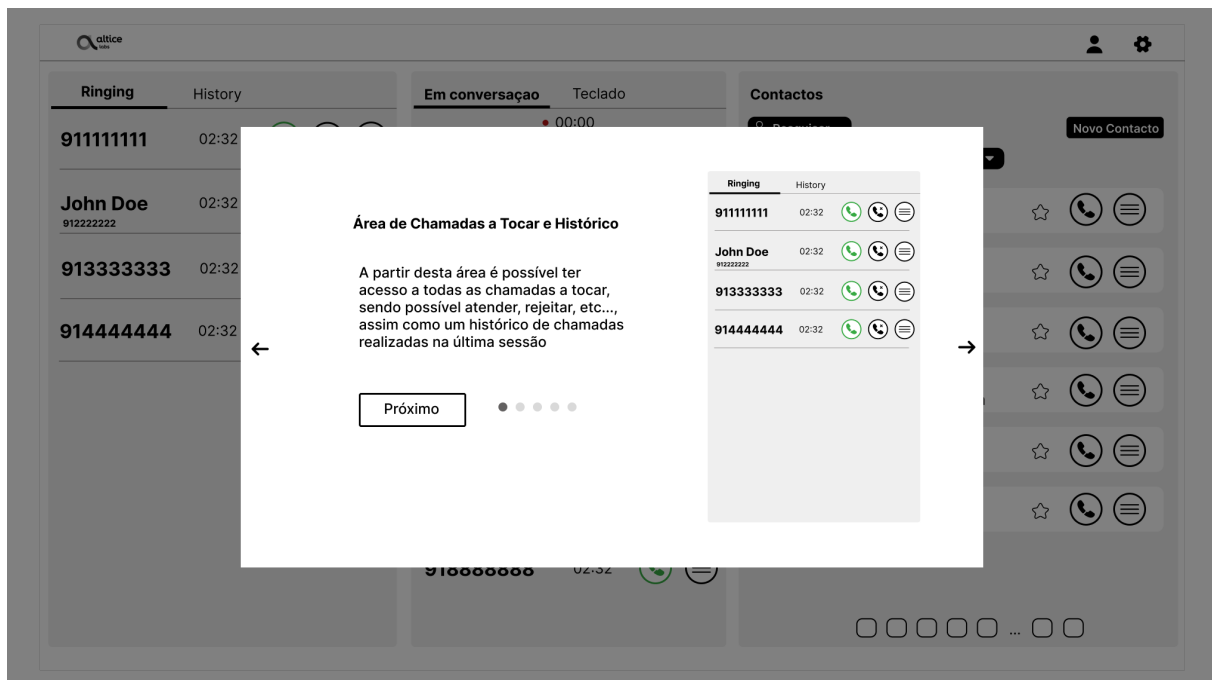


Figura 32: Mockup associada ao *Help* Online

O objetivo principal deste planeamento passa por tirar o máximo partido do paradigma de componentes utilizado pelo React para reutilizar todas as áreas implementadas na interface. Isto permite que, caso qualquer componente seja alterado, as alterações sejam também refletidas nesta área, o que, no longo prazo, melhora bastante a manutenção e a consistência do *Help* Online. A ideia geral passa por criar uma página para cada área da aplicação: chamadas a tocar, chamada em conversaão, chamadas pausadas e contactos, sendo possível programar algumas interações realizadas de forma automática para mostrar ao operador todas as operações, em tempo real, suportadas pela aplicação, e como estas funcionam. Além disso, é possível também colocar um breve texto do lado esquerdo que permite explicar casos mais específicos ou complicados, se for necessário.

Através deste ecrã, é possível ver que a modal permite ao utilizador realizar ou consultar este breve tutorial da aplicação sem interromper as suas atividades, funcionando a paginação como uma barra de progresso. Além disso, o utilizador pode optar por fechar este modal em qualquer altura.

3.5 Conclusão

Em retrospectiva, apesar de, por motivos de redundância, nem todos os *mockups* desenvolvidos estarem presentes neste documento, esta fase de planeamento revelou-se como incompleta. Por motivos de tempo e de cumprimento de calendário, e devido a várias áreas do desenho terem que ser reestruturadas múltiplas vezes, esta fase não foi planeada com rigor suficiente, e certas vertentes da interface, como o formulário de criação e edição de contacto e o comportamento responsivo da aplicação foram ignoradas. Como seria de esperar, esta falta de rigor gerou certas dificuldades ao desenvolver as áreas que careceram deste planeamento.

Além disso, será notado, no próximo capítulo, que vários componentes tiveram que ser alterados face ao planeamento, principalmente devido a uma funcionalidade adicional implementada e outros que, assim que implementados, foi possível ver que podiam ser melhorados, sofrendo alterações de forma a estarem em maior conformidade com o esperado.

3.6 Sumário

Este capítulo documentou as várias etapas que fizeram parte do processo de desenvolvimento. Inicialmente, foi necessário definir com detalhe qual o âmbito do trabalho, uma vez que, devido à natureza deste, não se encontrava explicitamente delineado. Com esse processo de decisão tomado, uma vez que foi optado pelo desenvolvimento de um protótipo de um novo cliente, integrando certas **APIs** já existentes, foi necessária uma extensiva fase de planeamento de forma a definir, com critério, a aplicação, antes de dar início ao desenvolvimento.

Um dos passos mais importantes do planeamento passou por um estudo relativo às tecnologias a utilizar no desenvolvimento. Algumas opções viáveis foram consideradas, sendo que, no final, foi escolhida a tecnologia React, que se considerou ser a mais adequada. Adicionalmente, de modo a completar o âmbito do trabalho, foi necessário definir os requisitos de utilizador da aplicação, sendo necessário um estudo detalhado relativo às **APIs** de modo a fazer um levantamento das funcionalidades suportadas por estas e o seu funcionamento, sendo utilizada uma lista de prioridades para tal.

Este processo de planeamento englobou também o desenho da arquitetura da aplicação, mostrando os sistemas externos com que esta interage e os protocolos de comunicação aplicados. Além disso, foi também estabelecido o funcionamento da aplicação, nomeadamente, as interações necessárias de acordo com as diferentes operações realizadas pelo utilizador final, para que fosse possível fazer o desenho da

interface gráfica, com recurso a *mockups*.

Capítulo 4

Implementação

Neste capítulo será detalhada a implementação de todo o sistema de consola de operador, sendo explicados os aspetos técnicos, como todos os componentes desenvolvidos, a arquitetura da aplicação e todas as decisões tomadas neste processo.

4.1 Considerações preliminares

Primeiramente, é importante explicar uma das alterações feitas ao planeamento, ou neste caso, uma adição de uma funcionalidade, que tem um elevado impacto em todo o resto da aplicação.

Como foi visto nas secções anteriores, a escolha da abordagem selecionada teve uma forte implicação no âmbito do projeto, uma vez que todas as **APIs** e serviços a integrar eram fortemente restritivos nas funcionalidades suportadas, levando a que essas limitações abrangessem a consola, que assim não tinha forma de as implementar. De modo a tentar contornar este problema, foi decidido que o protótipo a desenvolver poderia beneficiar de um modo de simulação, de forma a que a consola pudesse ser utilizada num modo *offline*, com dados simulados, sem qualquer comunicação com as **APIs**, permitindo assim, para este modo, a implementação das funcionalidades não suportadas pelas **APIs**.

Dada esta oportunidade, foram implementadas as seguintes funcionalidades adicionais:

- atender chamadas via consola;
- pausar e resumir chamadas.

Esta decisão foi bastante vantajosa uma vez que estas funcionalidades são importantes para a aplicação, e a sua ausência implicaria o desenvolvimento de um protótipo bastante incompleto. Desta forma, apesar de, no modo online, estas duas funcionalidades não estarem implementadas, é na mesma possível visualizar como é que estas funcionariam. Apesar desta enorme vantagem de eliminar parcialmente as restrições, a sua implementação acresceu à complexidade da solução uma vez que se torna necessário

fazer a gestão do modo de operação da consola, o que inclui alterações na interface e na camada aplicacional. Além disso, isto não surgiu como uma solução completa para o problema, apenas temporária de forma a desbloquear o desenvolvimento do cliente *web*.

4.2 Arquitetura do Front-end

A arquitetura da aplicação tem um papel fundamental no seu desempenho e manutenção. Esta arquitetura dita a estrutura e, de certa forma, o comportamento que permite que a aplicação desempenhe todas as suas tarefas, definindo como os componentes interagem, como os dados são armazenados e como a interface é apresentada [115; 116]. A arquitetura aqui mencionada difere da arquitetura abordada na secção 3.3 (ver Figura 15), na medida que pretende detalhar a arquitetura do cliente web desenvolvido, ao invés da arquitetura da aplicação com os sistemas externos utilizados.

Por estas razões foi prestada bastante atenção ao desenho e implementação da arquitetura de baixo nível [117] da aplicação, a divisão nos vários componentes da interface e padrões de desenvolvimento aplicados, com o objetivo de usufruir de vantagens relevantes para o âmbito deste trabalho [118; 119; 120], nomeadamente:

- modularidade - uma granularidade maior de código complexo permite partir este em componentes reutilizáveis, o que aumenta a reusabilidade e manutenção do código;
- separação de preocupações - separar a lógica de apresentação da lógica de negócio implica uma maior organização e colaboração;
- desenvolvimento acelerado - simplificar o processo de desenvolvimento através de componentes independentes.

Após um estudo relativo a diferentes práticas de desenvolvimento do React, foram levantados alguns padrões de desenvolvimento e práticas que devem ser analisadas antes de dar início ao desenvolvimento da aplicação [121; 122]. Um padrão de desenvolvimento é um padrão que descreve um problema que é frequentemente verificado em diferentes sistemas, sendo que descreve também uma abstração de software que resolve esse problema, de forma a que esta solução possa ser utilizada repetidamente [123; 124].

Este conceito é algo que pode ser aplicado a várias áreas do quotidiano, sendo uma destas ao desenvolvimento de uma aplicação *frontend*, especialmente quando é utilizada uma tecnologia como React.

Um dos grandes problemas da utilização desta tecnologia passa pela falta de uma estrutura ou arquitetura, proveniente do facto de que esta tecnologia, apesar de múltiplas vezes ser mencionada como uma *framework*, se tratar apenas de uma biblioteca.

Um dos grandes motivos que levou à escolha do padrão de desenvolvimento abaixo mencionado está relacionado com a necessidade de uma gestão de estados complexos relativos às chamadas e aos contactos, assim como aos dados do utilizador. Esta gestão do estado completo leva a três desafios:

- o envio de pedidos para as diferentes **APIs** em diferentes áreas da aplicação, e por implicação, em diferentes componentes da interface, acedendo à informação do utilizador, nomeadamente *tokens*, em todos estes componentes;
- as funcionalidades iniciais incluíam a adição do nome do contacto guardado quando era recebida uma nova chamada, quer seja esta iniciada ou recebida, e como esta informação não era fornecida pela **CTI API**, esta associação teria que ser feita no cliente;
- a adição de dois modos de operação diferentes, online e simulação, implicam que vários componentes necessitem de saber qual o modo de operação para serem executadas as tarefas adequadas.

Estes três itens podem ser vistos como problemas uma vez que implicariam algo denominado por *prop drilling* (ver Figura 33). Passar *props* (propriedades/objetos onde o valor de atributos é guardado) pelos componentes é a forma correta de passar dados pela árvore de elementos da interface, uma vez que a tecnologia utilizada suporta um fluxo descendente e unilateral de dados de um componente para os componentes filho. No entanto, pode ser algo que rapidamente se torna inconveniente quando estes dados tem que ser passados para um componente localizada no fundo da árvore [3].

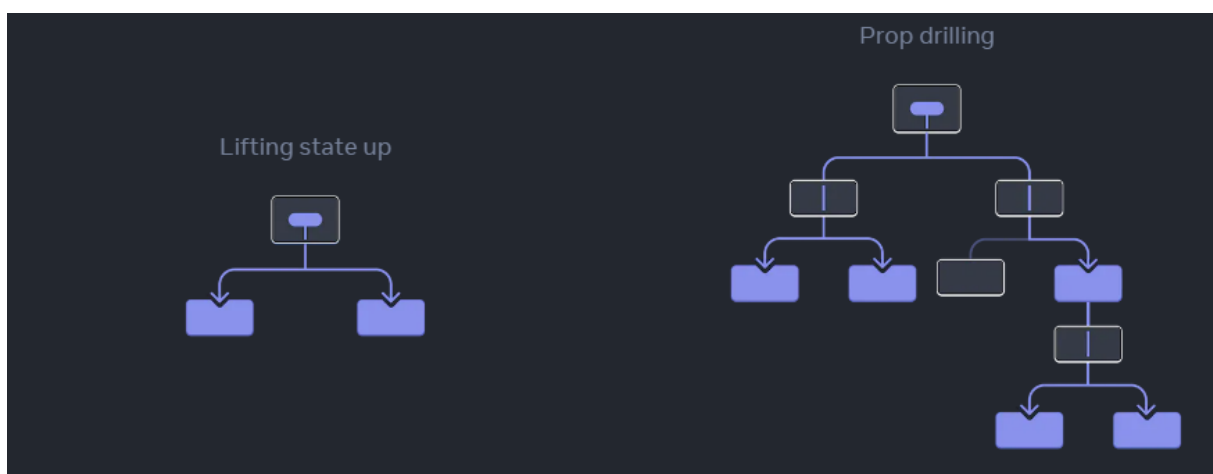


Figura 33: Imagem utilizada para descrever *prop drilling*, retirada da documentação oficial do React [3]

É precisamente neste contexto que surge o padrão de desenvolvimento *React Provider Pattern*, que utiliza uma funcionalidade do React denominada por React Context (ver Figura 34). Esta funcionalidade permite enviar informação entre componentes dentro de uma árvore de componentes, saltando todos os componentes intermediários e transmitir a informação apenas para os componentes desejados, sem ser necessário passar os dados manualmente em cada nível [3; 125; 126; 127].

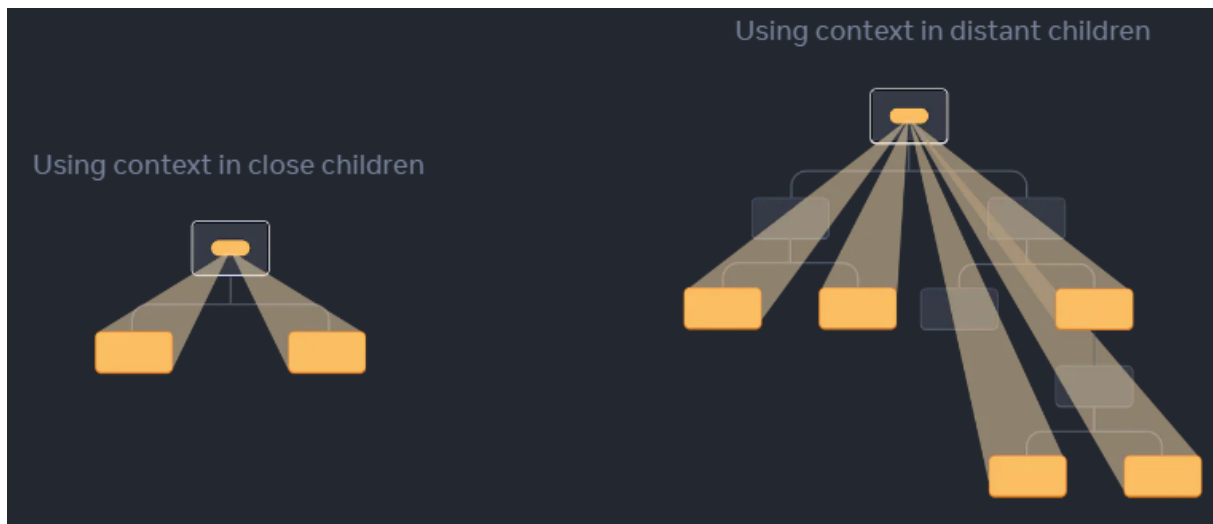


Figura 34: Imagem utilizada para descrever React Context, retirada da documentação oficial do React [3]

A implementação do React Context é bastante direta e clara, podendo ser dividida em três passos:

- criar e exportar o contexto, que trata de passar os dados pretendidos;
- utilizar o contexto nos componentes que necessitam dos dados, feito através do *hook useContext*, fazendo com que o componente subscreva aos dados do contexto especificado;
- fornecer esse mesmo contexto através do componente que especifica os dados, envolvendo todos os componentes pretendidos com o fornecedor de contexto.

Desta forma, é possível utilizar esta implementação para a lógica dos estados das chamadas, dos contactos, e do utilizador autenticado, para que toda a informação possa ser acedida em todos os componentes da árvore, simplificando bastante o processo.

No entanto, isto não basta para as funcionalidades pretendidas, uma vez que é necessário que estes componentes consigam alterar os dados em tempo real, como adicionar chamadas iniciadas pelo operador via consola, criação e edição de contactos e alteração do modo de operação da consola. Para tal, é necessário combinar o contexto, explicado acima, com um *reducer*. React *reducers* permitem consolidar

toda a lógica de atualização de dados isolada de todos os componentes numa única função, denominada *reducer* [128].

Esta abordagem tem a enorme vantagem de simplificar estas atualizações de estado, o que no caso concreto da aplicação a desenvolver, é bastante benéfico, uma vez que os diferentes estados a gerir terão um elevado número de alterações complexas. De acordo com a documentação do React [128], a gestão do estado, quando esta funcionalidade é utilizada, passa de uma simples alteração do estado para especificar ao *reducer* o que o utilizador fez utilizando o *dispatch* para indicar as ações correspondentes ao gestor de eventos.

O *reducer* é utilizado em conjunto com o *context* de forma a que o estado e a função de *dispatch* sejam colocados no contexto e passados aos componentes necessários, sendo que assim estes são capazes de aceder a estes dados e acionar alterações nestes, evitando o mencionado *prop drilling* para ambos os casos (ver Figura 35).

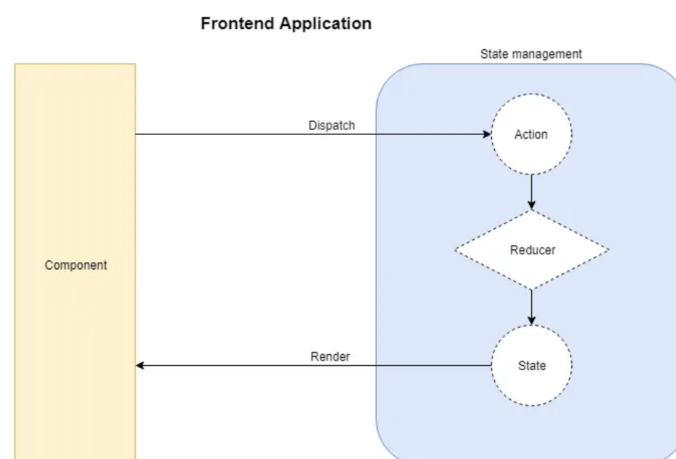


Figura 35: Funcionamento da gestão de estado com *reducer* e *context* [4]

Esta combinação de *reducers* com *context*, mencionado pela documentação React por escalar *reducer* com *context* [3] compõe grande parte da lógica aplicacional da solução, sendo responsável por armazenar os dados da aplicação, requisitados pelos vários componentes da interface, assim como fazer a sua gestão.

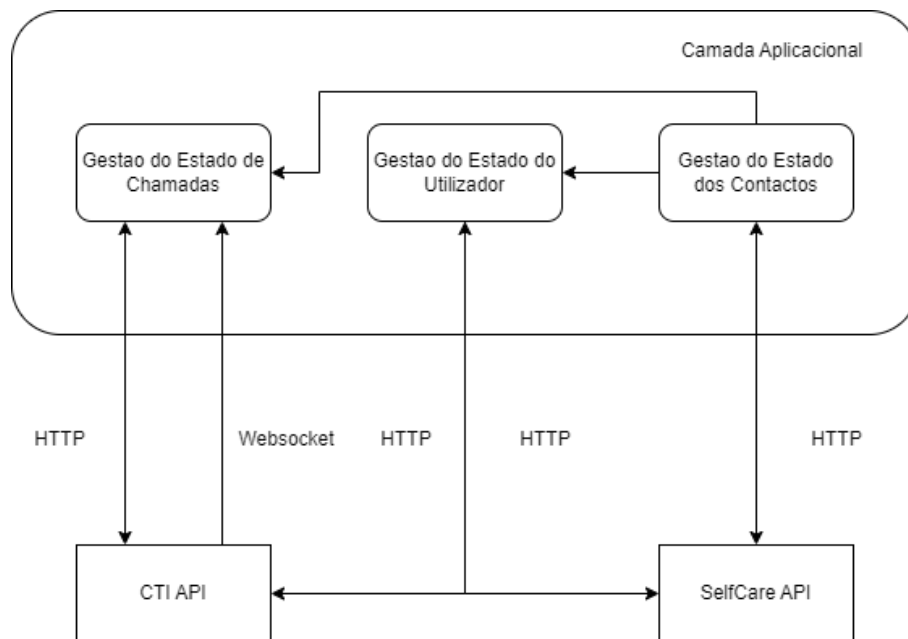


Figura 36: Camada aplicacional do protótipo

Através do diagrama da Figura 36 é possível ver o comportamento da camada aplicacional para lidar com todas as alterações de estado associadas. Primeiramente, o contexto associado ao utilizador trata de realizar a autenticação, como ilustrado na Figura 16, sendo necessário efetuar a autenticação em ambas as **APIs** uma vez que, como são completamente distintas, é obrigatório um *token* diferente para cada uma. Depois disso, o *token* é guardado neste contexto para que possa, posteriormente, ser utilizado pelo contexto associado aos contactos para obter a lista através da SelfCare **API** e pelo contexto das chamadas para subscrever na **CTI API**. Esta informação é também utilizada em qualquer pedido subsequente.

Com as Figuras 34 e 35, constrói-se a arquitetura geral da aplicação, representada na Figura 37.

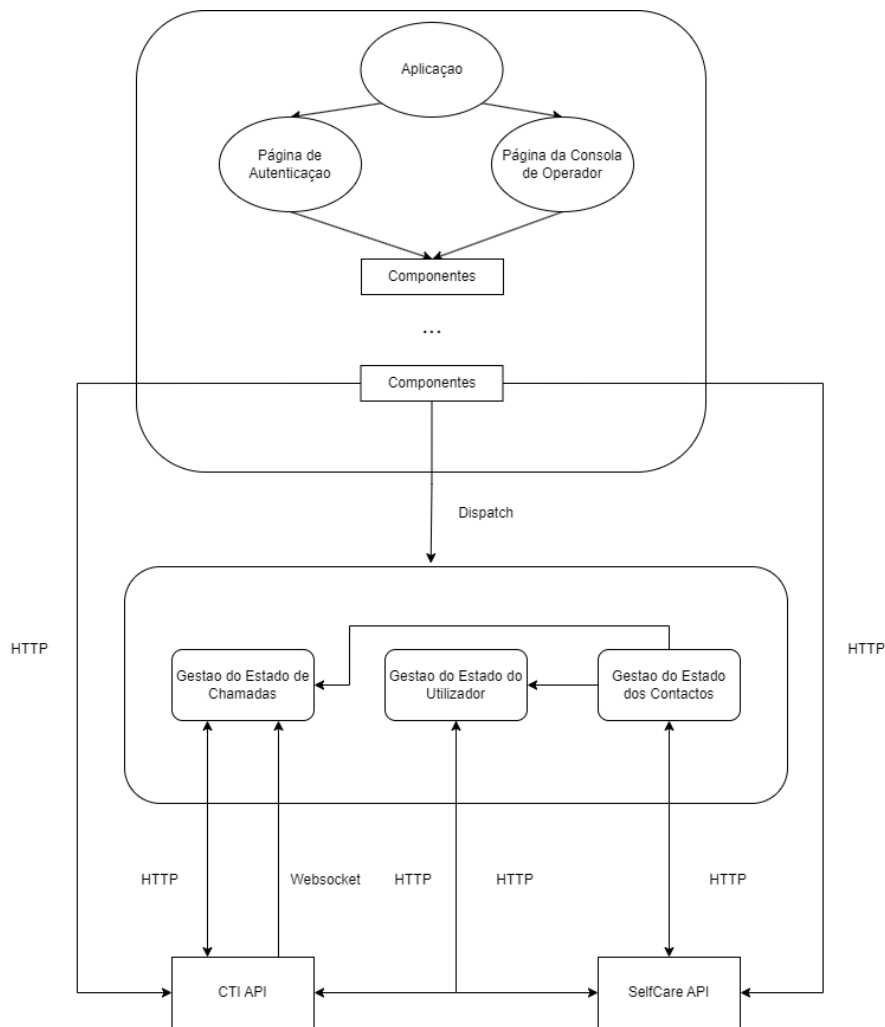


Figura 37: Arquitetura de alto nível da aplicação *frontend*

Como é possível observar na imagem, os componentes consomem a informação proveniente da camada aplicacional, e produzem alterações nestes dados através do *dispatch*. Os componentes encontram-se conectados a todas as **APIs**, uma vez que é necessário, em certos casos, enviarem pedidos HTTP uma vez que estes são assíncronos, e como tal, não devem pertencer ao *reducer* [129]. Neste caso, os pedidos são enviados pelos componentes e, de acordo com a resposta, é feito o respetivo *dispatch*.

Gestão do estado do Utilizador

Este componente da camada aplicacional, tal como o nome indica, é responsável por tratar da lógica da autenticação e de armazenar e fornecer os dados relevantes do utilizador, assim como o modo de utilização da consola, indicando se esta está a operar no modo simulação ou no modo online.

Estes dados do utilizador são compostos pelos *tokens* das **APIs** e do nome e email do utilizador. Toda esta informação é mantida no armazenamento local do *browser*, fazendo assim com que a sessão

do utilizador permaneça ativa mesmo que este encerre o browser. Esta sessão não é, no entanto, ilimitada, sendo que possui um tempo limite atribuído pelo servidor ao *token*, que, após terminar, invalida o *token*, rejeitando os pedidos.

Além disso, como indicado, este componente armazena também a informação relativa ao modo de operação da consola de forma a passar essa informação aos componentes e aos outros contextos, estipulando quais as operações a serem executadas. Por último, é também necessário manter dois valores indicando se o processo de autenticação das duas **APIs** já se encontram finalizados, uma vez que, como ilustrado na Figura 16, depois da obtenção dos *tokens*, o contexto dos contactos e das chamadas iniciam as suas operações em paralelo. Este valor indicativo do processo de autenticação é importante, pois é posteriormente utilizado por outros componentes de forma a transmitir ao utilizador se a consola já se encontra pronta a ser utilizada.

Para terminar este contexto, através do *reducer*, são suportadas as seguintes operações:

- efetuar *logout* - limpa todos os dados do utilizador do contexto e do armazenamento do *browser*, assim como o histórico de chamadas;
- autenticar na **CTI API** - assim que o *login* é feito e os pedidos de autenticação são enviados em paralelo pelos componentes respetivos, esta operação de guarda toda a informação relevante obtida desse processo associado à **CTI API**;
- autenticar na SelfCare *api* - efetua exatamente o mesmo que o método de cima, mas para a SelfCare **API**;
- alterar modo - altera o modo de utilização da consola, iniciando novamente o processo de autenticação do utilizador. Caso seja para o modo de simulação, inicializa os valores pré-configurados.

Gestão do estado de chamadas

Este componente funciona da mesma forma que o componente abordado acima, sendo que tem o papel de gerir e fornecer toda a informação relevante das chamadas e do respetivo *reducer* para suportar diferentes operações. Possui também alguns dados pré-configurados que são utilizados no modo de simulação.

Como é possível ver no diagrama da Figura 37, este componente consome a informação do contexto do utilizador para ter acesso à sua informação de forma a enviar pedidos HTTP e para ter acesso ao modo de operação da consola, valor que é importante para definir, em tempo real, quais operações a executar.

Sempre que o modo de operação é alterado, é necessário acionar as seguintes operações:

- caso seja para o modo online, é necessário limpar todas as chamadas que, se existem, é devido ao modo de simulação e portanto devem ser apagadas, sendo também necessário repor o histórico de chamadas a partir do armazenamento do *browser*. Posteriormente, é necessário enviar um novo pedido de subscrição, como visto na metade inferior da Figura 16;
- caso seja para o modo de simulação, é necessário encerrar a conexão com a *websocket* e repor os dados das chamadas para os dados iniciais pré-configurados associados ao modo de simulação.

Este componente é também responsável por tratar de toda a lógica associada à conexão com a *websocket*, iniciada após o pedido de subscrição, com o devido tratamento de erros caso a conexão seja interrompida e com as devidas funções que lidam com todas as notificações recebidas. Este estado de conexão é fornecido aos componentes subscritos, indicando se a consola se encontra disponível para ser utilizada no modo online, uma vez que, caso a conexão não esteja estabelecida, é impossível receber notificações relativas a chamadas, tornando impossível a sua utilização.

Através destas notificações e das ações realizadas pelo utilizador, é mantida uma lista de chamadas a tocar, lista de chamadas pausadas, a chamada em conversação e o histórico de chamadas, possuindo todos os dados relevantes e que são passados aos componentes subscritos, para que possam ser apresentadas ao operador.

É, no entanto, maioritariamente através das notificações que estes dados são alterados, uma vez que é para a *websocket* que são enviadas todas as chamadas recebidas ou operações realizadas diretamente no *softphone*, assim como a confirmação de que as operações realizadas na consola foram efetuadas com sucesso, e como tal, é crucial ter um processo de tratamento de notificações bem definido (Figura 38):

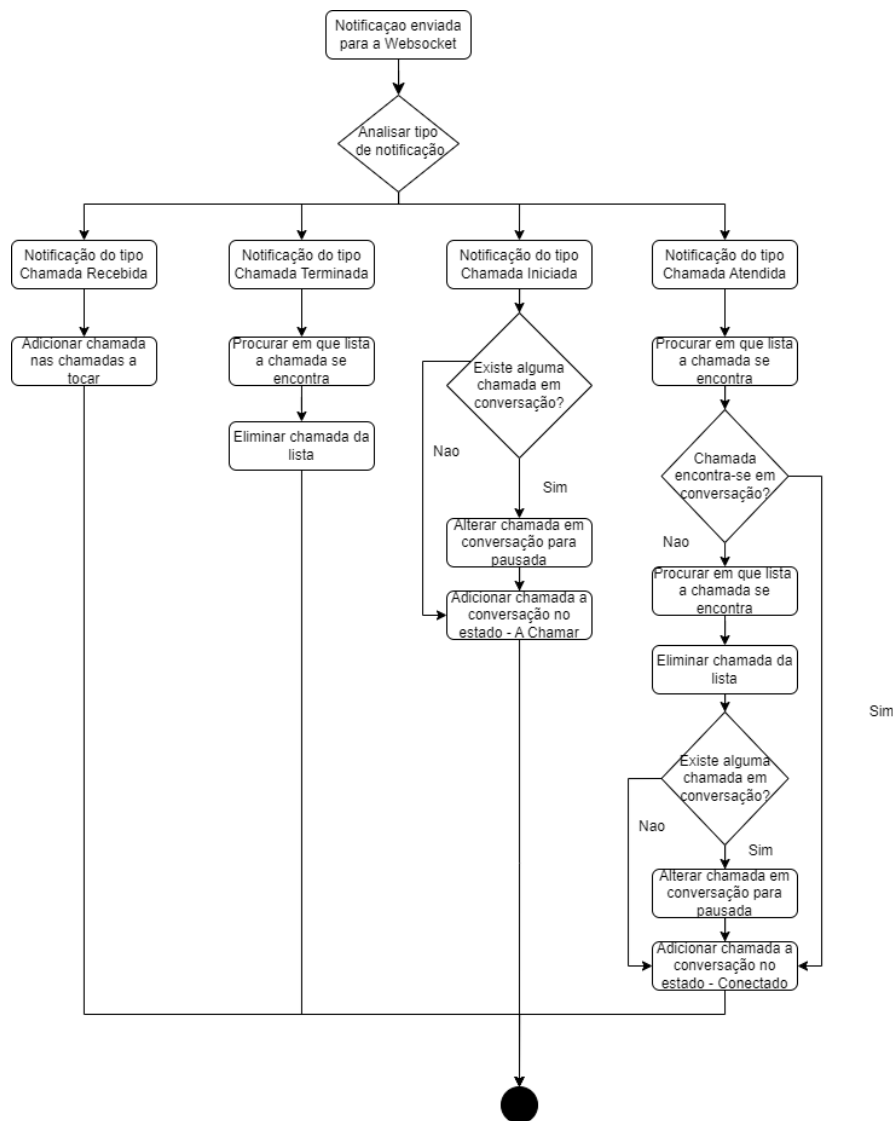


Figura 38: Lógica associada ao tratamento das notificações

Por último, o *reducer* associado a este contexto suporta seis operações diferentes, sendo que algumas destas são necessárias para evitar problemas originários no serviço integrado:

- iniciar chamada - utilizado para adicionar uma nova chamada iniciada pelo operador via consola. Como estabelecido previamente, esta ação é apenas utilizada depois do pedido para a respetiva **API** ser enviado e a resposta de sucesso ser recebida caso seja no modo online, sendo que no modo de simulação pode ser adicionada de imediato. Devido ao comportamento da **API** neste cenário, é necessário implementar uma lógica de prevenção de erros, uma vez que chamadas iniciadas por este método apenas são enviadas para a *websocket* quando atendidas no destino. Como não é assegurado que, após resposta de sucesso enviada pela **API**, a chamada foi iniciada, devido a problemas que possam acontecer na comunicação da **API** com o serviço, é necessário

executar estas verificações para, em caso de anomalia, apresentar mensagem de erro ao utilizador;

- terminar chamada - permite limpar uma chamada em progresso via consola. À semelhança do método anterior, no modo online, esta ação é apenas utilizada depois do pedido e respetiva resposta de sucesso ser obtida da **API**. Como esta resposta de sucesso não assegura que a chamada será terminada, devido a problemas de comunicação da **API** com o serviço, este método tem a responsabilidade de, ao final de um determinado tempo pré-definido, verificar se a chamada foi terminada, para caso não ter acontecido, apresentar uma mensagem de erro ao utilizador. Caso seja no modo simulação, a chamada é de imediato removida;
- atender ou resumir chamada - utilizado para implementar as funcionalidades de atender e resumir chamadas, não permitidas pela **API**, no modo simulação;
- pausar chamada - utilizado para implementar a funcionalidade de pausar chamada não suportada pela **API** no modo de simulação.

Gestão do estado dos Contactos

O último contexto abordado diz respeito aos contactos. Este tem um funcionamento ligeiramente diferente do anterior, uma vez que neste caso, não é necessário qualquer componente enviar pedidos HTTP, estando todos estes ao cargo do contexto. Na Figura 36, é possível ver que os componentes se encontram conectados à SelfCare **API**, que é a utilizada para esta vertente, mas que, no entanto, é somente utilizada por estes na fase de autenticação.

Os dados fornecidos por este contexto são compostos pela lista atual de contactos, podendo trocar de acordo com a página selecionada, assim como a página selecionada e número total de páginas. É necessário também informação relativa ao estado da lista de contactos, indicando se esta se encontra a carregar, para apresentar uma barra de progresso ao utilizador, ou mensagens de erro caso algo de errado aconteça.

É também guardado num objeto todos os dados relativos aos filtros aplicados. Estes filtros são alterados pelo *dispatch* acionado pelo utilizador, de modo a que, sempre que um destes valores é alterado, o contexto envie um novo pedido para a **API** de forma a apresentar uma nova lista com os filtros aplicados. Como este pedido se encontra dentro do contexto e não do *reducer*, não é necessário os pedidos serem efetuados nos componentes, o que permite uma maior modularização. Além disso, é também importante prestar atenção ao modo de operação da consola, uma vez que, sempre que este é alterado para simulação, é necessário alterar a lista de contactos para os valores pré-configurados. Neste caso, sempre que

os filtros ou página sejam alterados, essa lista precisa de ser alterada ou filtrada.

Relativamente ao *reducer*, este suporta um elevado número de operações que permitam manipular todos os filtros selecionados e os próprios contactos. Foi criada uma ação para cada filtro, assim como ações para editar, criar e eliminar contactos, que dependendo do modo de operação da consola, podem acionar um novo pedido e retornar a nova lista ou apenas atualizar a lista, que é realizada apenas após o pedido de edição receber resposta de sucesso.

4.3 Interface

A área de resultados obtidos tem como objetivo demonstrar a interface desenvolvida e indicar todas as alterações feitas de acordo com os *mockups* desenvolvidos durante a fase de planeamento. Para isso, serão utilizadas árvores de componentes, assim como serão retiradas e explicadas as imagens mais relevantes da interface final, uma vez que não é possível colocar imagens relativas a todo o seu comportamento.

O Modelo de Objeto de Documentos (em Inglês - **DOM**) representa a estrutura hierárquica do documento HTML associado à página web numa estrutura semelhante a uma árvore, onde cada elemento do documento é representado como um nó na árvore, podendo este ter nós pais e nós filhos, definindo assim a estrutura da página [130; 131].

São precisamente estas relações hierárquicas, e os componentes que compõe a interface, assim como o encapsulamento em blocos modulares e reutilizáveis que se pretende mostrar com os diagramas de árvores de componentes [132]. Com estas árvores, que tencionam mostrar apenas os componentes React implementados, é possível expor como as funcionalidades a implementar foram divididas em problemas de menor tamanho, aumentando o nível de organização do código e promovendo a escalabilidade.

Por questões de apresentação, estes diagramas estão apresentados por partes, de modo torná-los mais legíveis.

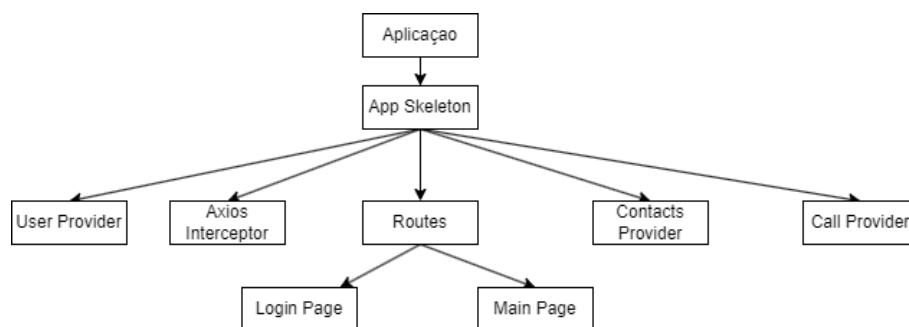


Figura 39: Parte superior da árvore de componentes

Através do diagrama da Figura 39 é possível ver como está definida a parte mais perto da raiz da aplicação desenvolvida. As rotas definem quais páginas são atribuídas aos diferentes *Uniform Resource Locator (URLs)*, incluindo as verificações necessárias relacionadas com a autenticação, direcionando o utilizador para diferentes páginas caso esteja ou não autenticado. Adicionalmente, o componente *Axios Interceptor* é responsável por intercepar todos os pedidos HTTP feitos através do cliente *axios* [133] de modo a executar um tratamento de erros geral, ou seja, verificar se as mensagens de erro provenientes das **APIs** indicam se o *token* se encontra expirado, e caso aconteça, terminar a sessão do utilizador. Finalmente, os componentes *User Provider*, *Contacts Provider* e *Call Provider* correspondem aos contextos do utilizador, contactos e chamadas, respetivamente.

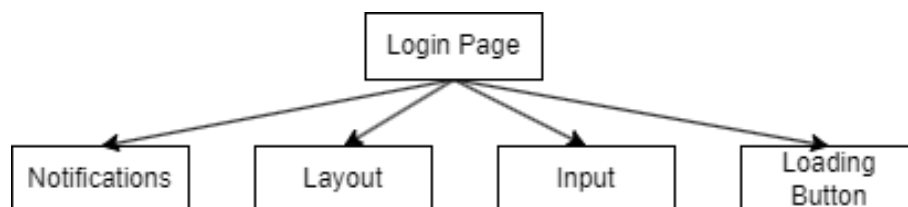


Figura 40: Árvore de componentes relativa à Página de Autenticação (*Login Page*)

Como ilustrado na Figura 40, a página de autenticação possui uma árvore de componentes relativamente simples, uma vez que disponibiliza ao utilizador um formulário com apenas dois campos para *e-mail* e password, apresentados com o componente *Input* e um botão de submissão, composto pelo componente *Loading Button*, que apresenta uma animação quando este é submetido. Além disso, o componente *Notifications* possibilita a exibição de uma pequena notificação no ecrã, de erro ou sucesso, utilizada para várias finalidades. Por fim, o componente *Layout* trata apenas de fazer a divisão do espaço do ecrã de acordo com o desejado.

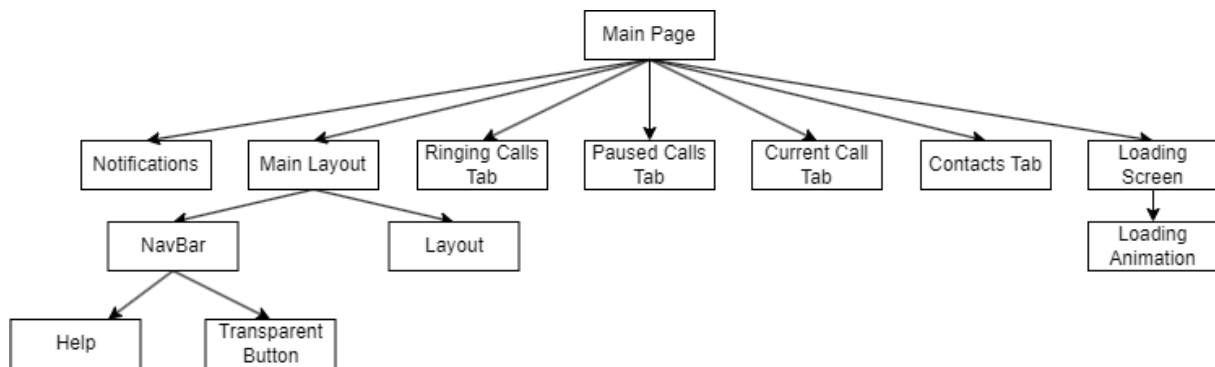


Figura 41: Árvore de componentes relativa à Página da consola (*Main Page*)

Através do diagrama da Figura 41 é possível ver a estrutura da página da consola. Por motivos de visualização, devido à sua elevada complexidade e um elevado número de profundidade, os componentes *Ringing Calls Tab*, *Paused Calls Tab*, *Contacts Tab* e *Help* serão divididos para os seus próprios diagramas. Esta página reutiliza os componentes de *Notifications* e *Layout* previamente explicados, utilizando o *Main Layout* para apresentar uma barra de navegação ao utilizador, que possui também o *Help Online*. Este será expandido mais à frente, relativo à página de ajuda. Além disso, o *Loading Screen* é responsável por apresentar um ecrã com uma animação de carregamento, sendo utilizado quando a conexão à *websocket* ainda não se encontra finalizada, impedindo o operador de utilizar o resto da aplicação e informando-o que ainda não pode ser utilizada.

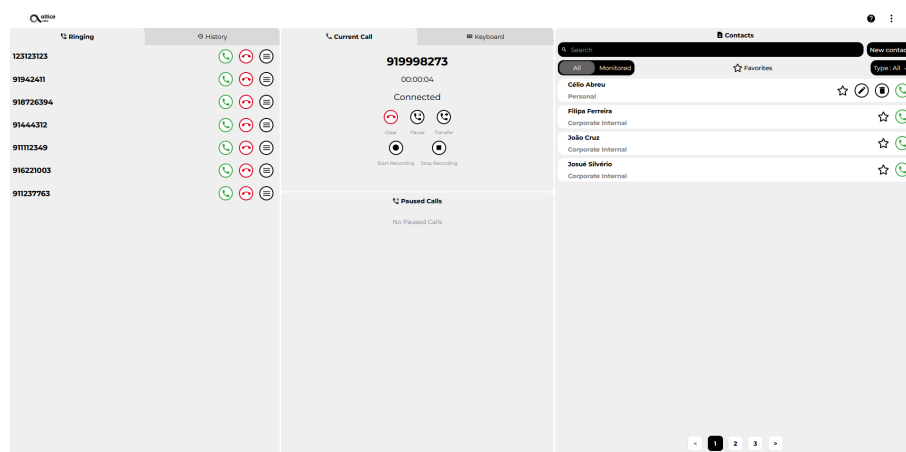


Figura 42: Imagem da interface da *Main Page*

A imagem incluída na Figura 42 mostra que a interface final sofreu várias alterações face ao que foi planeado. As alterações efetuadas específicas de cada área serão discutidas em detalhe mais à frente, quando a árvore de componentes para cada uma delas for abordada. Uma das alterações ao *layout* da

aplicação diz respeito à barra de navegação, uma vez que foi um componente que não foi planeado em detalhe devido à pouca funcionalidade que se previa que esta possuísse. No entanto, com a adição do modo de simulação, foi necessário a implementação de um mecanismo que permitisse alterar entre os modos de operação, sendo adicionado neste componente. Para isso, foi acrescentada uma pequena área do utilizador, indicada pelo ícone mais à direita na barra de navegação, onde, quando pressionado, é aberto um pequeno menu (Figura 43).

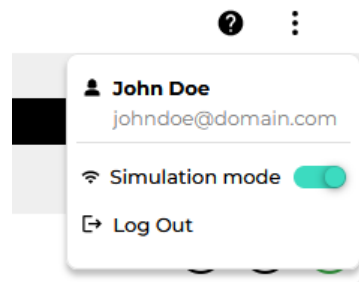


Figura 43: Imagem da interface da zona do perfil do utilizador

Esta zona indica o nome e email do utilizador autenticado, possuindo também a opção de alterar entre modo simulação e modo *online*, assim como uma opção para terminar a sessão. Além disso, o ícone mais à esquerda é utilizado para dar início ao tutorial presente na área de *Help Online*.

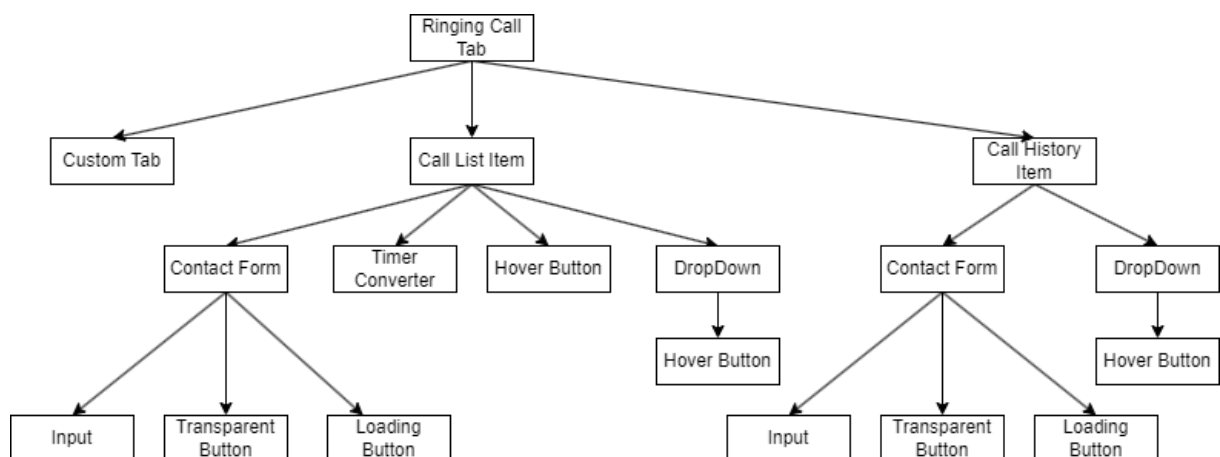


Figura 44: Árvore de componentes relativa ao componente *Ringing Call Tab*

No diagrama da Figura 44, pertencente ao componente *Ringing Call Tab* e utilizado na página da consola, é possível ver a elevada reutilização de componentes e que estes são utilizados em diferentes níveis da hierarquia, quando comparado com os diagramas anteriores. O componente *Ringing Call Tab* é responsável por apresentar ao operador a lista de chamadas a tocar e a lista dos registos de chamadas

(Figura 45 e 46). Esta área encontra-se maioritariamente inalterada, sendo utilizados *tabs* para alterar o conteúdo a apresentar entre a lista de chamadas a tocar e a lista do histórico, provenientes do contexto de chamadas, mapeadas de forma a utilizar um componente *Call List Item* e *Call History Item* para apresentar cada uma, respetivamente. Para terminar esta área, o *DropDown* serve para o menu de contexto, onde entre várias opções, consta a opção de adicionar ao contacto. Como tal, é utilizado o componente *Contact Form* de forma a abrir o formulário, inicializado com os valores provenientes da chamada.

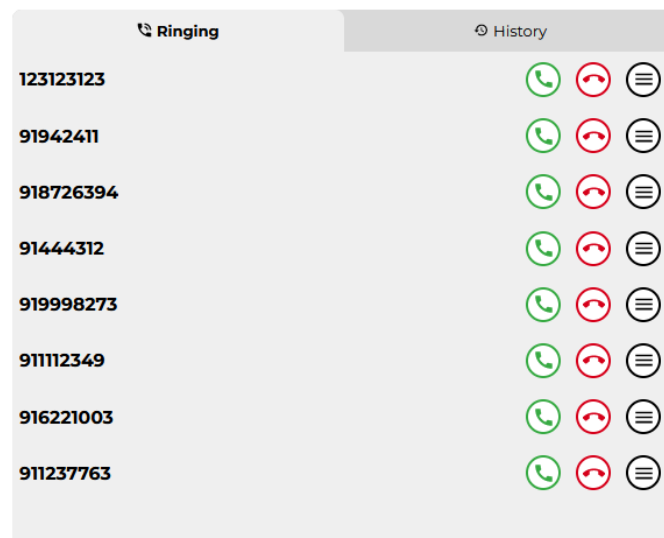


Figura 45: Imagem da interface relativa à área de chamadas a tocar

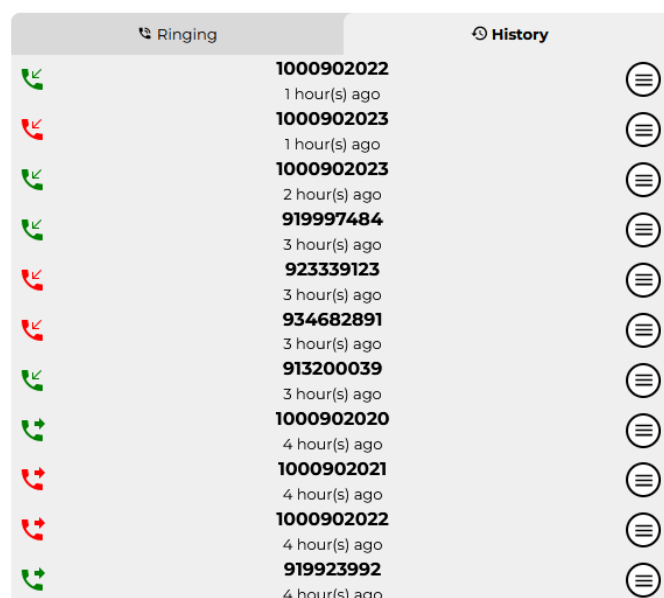


Figura 46: Imagem da interface relativa à área do histórico de chamadas

Uma das alterações não apenas desta área, mas da aplicação no geral, está relacionada com as *tabs* utilizadas. Como é possível ver através das Figuras 45 e 46, o aspeto deste componente foi totalmente reavaliado de modo a fornecer uma indicação mais simples e clara de qual elemento da *tab* se encontra selecionado, assim como um desenho mais limpo [134]. Adicionalmente, vários ícones foram alterados, uma vez que os ícones escolhidos para a fase de planeamento eram os fornecidos pela ferramenta e, portanto, eram meramente uma sugestão de apresentação, sendo escolhidos outros mais adequados. Por último, foi efetuada uma alteração que removeu alguma funcionalidade da aplicação. Tal como visto na fase de planeamento, as chamadas recebidas ou efetuadas, independentemente da área onde estas se encontrem, continham o nome do contacto associado ao número, caso existisse (Figura 23 e Figura 24). Esta funcionalidade foi uma grande motivação que levou ao processo de decisão da arquitetura aplicacional do protótipo desenvolvido, uma vez que era algo não implementado na **API** utilizada para as chamadas, e teria que ser implementada no cliente. Ou seja, por este motivo, sempre que uma notificação de chamada era recebida pela primeira vez, seria necessário procurar o contacto desse número e associar à chamada, para que essa informação persistisse pelo ciclo de vida completo no armazenamento da aplicação. No entanto, foi algo que se revelou impossível, uma vez que a **API** dos contactos, por motivos de desempenho, apenas permite obter através de um pedido HTTP um número limitado de contactos, de forma a implementar um método de paginação. Por esta razão, é necessário assumir que, na maior parte dos casos, a lista de contactos completa não se encontrará armazenada na consola. Portanto, não é possível associar os nomes guardados dos contactos sem uma perda de desempenho significativo da SelfCare **API** e do cliente web desenvolvido, sendo uma funcionalidade que foi abandonada. Portanto, quer seja na área de chamadas a tocar, pausadas, em conversação, ou histórico, o nome do contacto nunca estará associado ao número, caso este exista.

Relativamente à área de chamadas pausadas, é reutilizado o componente *Call List Item* para apresentar as chamadas a tocar, sendo o único componente reutilizado em *Paused Calls Tab*, e portanto, carece de árvore de componentes. A interface desta área é ilustrada na Figura 47.

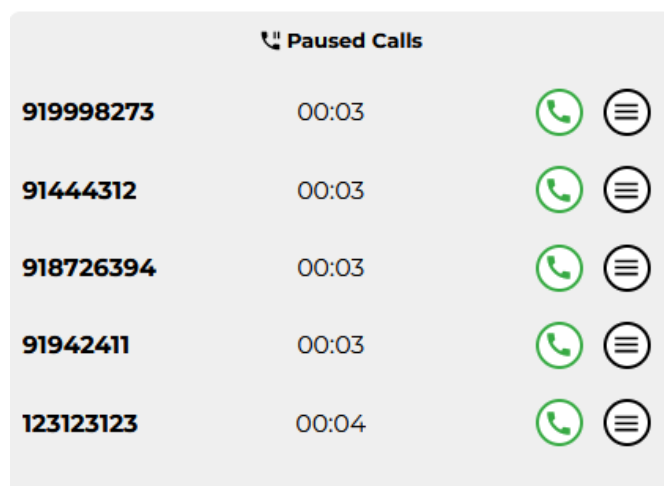


Figura 47: Imagem da interface relativa à área de chamadas pausadas

Através desta figura é possível ver que a interface permaneceu relativamente igual ao que estava planeado, com a exceção de alguns ícones e com o nome do contacto associado ao número, alterações que já foram abordadas.

No que diz respeito à área onde é apresentada a chamada em conversação, esta possui a seguinte estrutura:

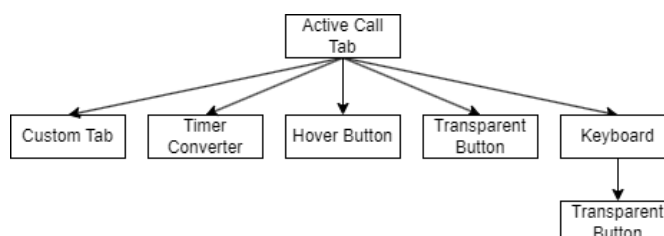


Figura 48: Árvore de componentes relativa ao componente *Conversation Call Tab*

Através do diagrama na Figura 48 é possível ver que esta área possui uma árvore relativamente mais simples. É utilizado o componente *keyboard* para apresentar o teclado ao utilizador, com o auxílio do componente *Transparent Button* para mapear os vários botões. O *layout* do teclado é escolhido através de uma configuração, e pode ser facilmente alterado. À semelhança das restantes áreas, é utilizado o *Custom Tab* para que o utilizador consiga alterar do teclado, para a informação da chamada em conversação, que é obtida através do contexto das chamadas.

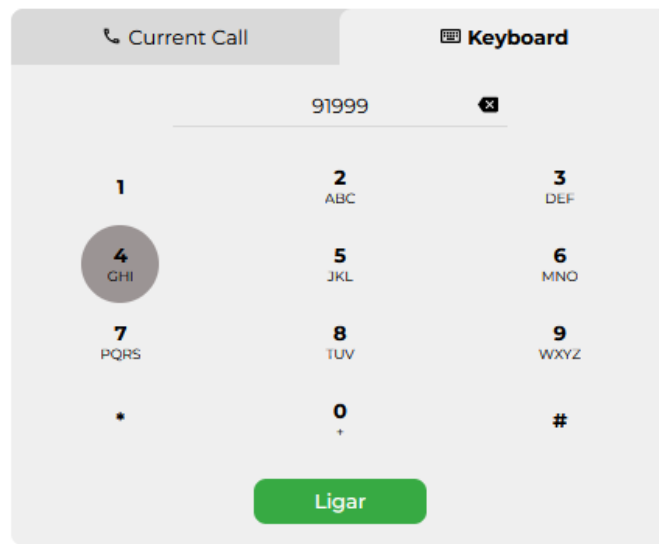


Figura 49: Resultado final da implementação do teclado

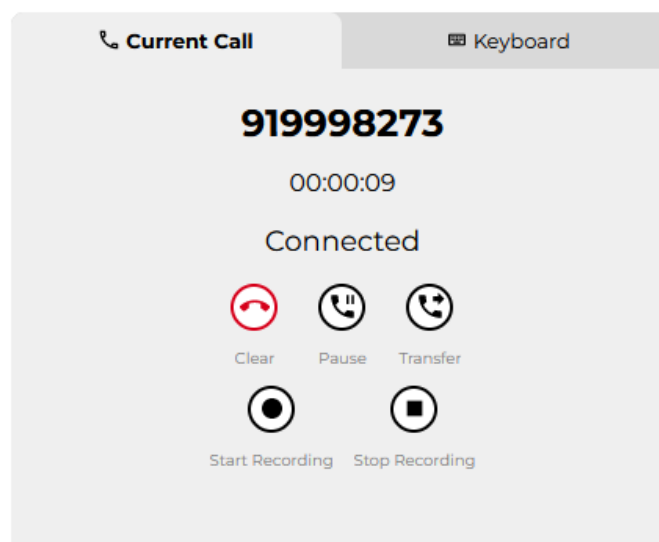


Figura 50: Resultado final da implementação da área de chamada em conversação

Através das Figuras 49 e 50 é possível ver o aspeto final da interface no que diz respeito ao teclado e à chamada em conversação. Esta parte sofreu relativamente poucas alterações de acordo com o que estava previamente planeado. É possível ver na Figura 49 o comportamento dos botões do teclado quando o cursor passa por cima de cada um, assim como o botão que permite iniciar a chamada para o número inserido. Posteriormente, a área da chamada em conversação apresenta os detalhes da chamada relevantes para o operador, assim como as operações suportadas de limpar e pausar chamada, estando esta última implementada no modo simulação (caso contrário é apresentada uma mensagem de erro).

Inclui ainda outros botões de funcionalidades que, por não serem de elevada prioridade ou não fazerem parte do âmbito da consola, não se encontram implementadas, mas, uma vez que a fase de planeamento teve em conta a sua existência, foram adicionadas.

Quanto à área dos contactos, esta possui a seguinte estrutura:

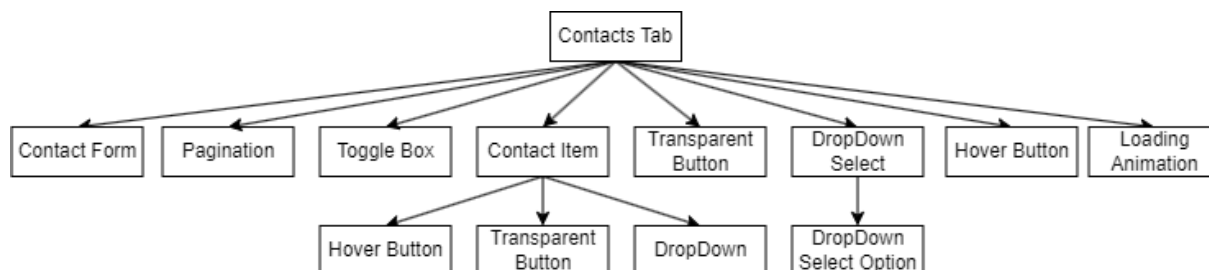


Figura 51: Árvore de componentes relativa ao componente *Contacts Tab*

O diagrama da Figura 51 indica que foram novamente utilizados vários componentes previamente aplicados, de forma a manter uma interface consistente e familiar para o utilizador. É também utilizado o componente *Contact Form*, previamente visto em outras áreas, para que os contactos possam ser criados e editados.

O componente *DropDown Select*, uma variação do componente *DropDown*, possui uma funcionalidade que, em conjunto com o elemento *Toggle Box*, permite implementar a zona dos filtros, sendo esta zona alvo de várias diferenças, quando comparada com o previamente planeado. Por último, o componente *Contact Item* é utilizado para mapear e apresentar todos os contactos contidos na lista.

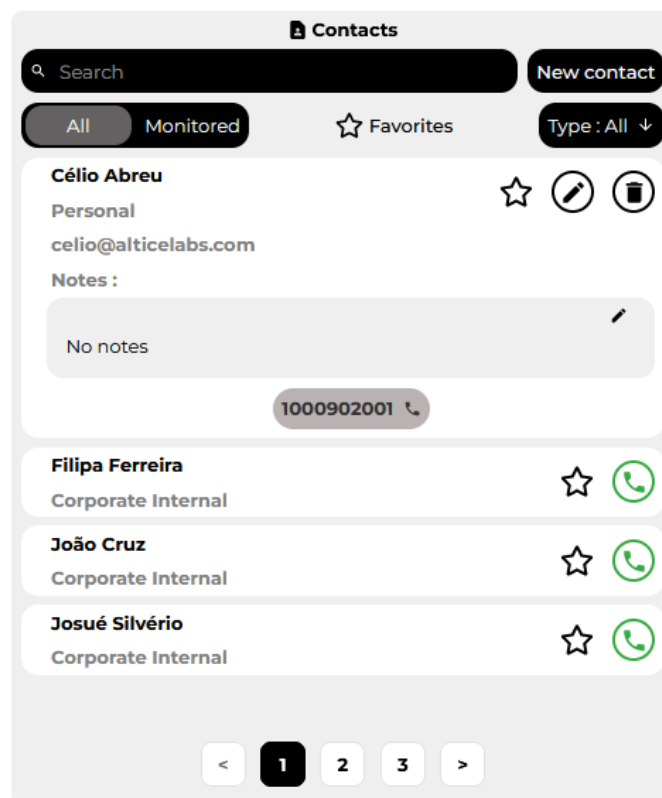


Figura 52: Imagem da interface relativa à área dos contactos

A Figura 52 apresenta as alterações efetuadas face aos *mockups* discutidos na fase de planeamento. De forma a simplificar a utilização dos filtros, as categorias, que previamente utilizavam um pequeno menu de *dropdown*, à semelhança da que se encontra do lado direito com o nome "Tipo", foram alteradas para um *toggler*, que permite alterar entre todos os contactos, ou apenas os monitorados. Os favoritos foram separados para um pequeno botão sobre a forma de ícone, que permite filtrar a lista selecionada pelos favoritos. Desta forma, são mantidas todas as funcionalidades dos filtros. No entanto, o método de filtração torna-se mais claro e com um desenho e utilização mais simples, uma vez que juntar filtros como monitorados e favoritos, que funcionam de forma diferente, poderia ser confuso para o utilizador final. É importante mencionar que neste caso, o estado "Todos" (apresentado na imagem por *All*) se refere aos contactos não monitorados, sendo portanto, mutuamente exclusivos. Entende-se que isto pode apresentar alguns problemas de utilização, mas simplesmente não foi encontrada uma melhor palavra para descrever a lista de contactos que não contém os monitorados. Como planeado, os contactos podem ser expandidos, podendo também ser editados ou eliminados, caso sejam do tipo pessoais.

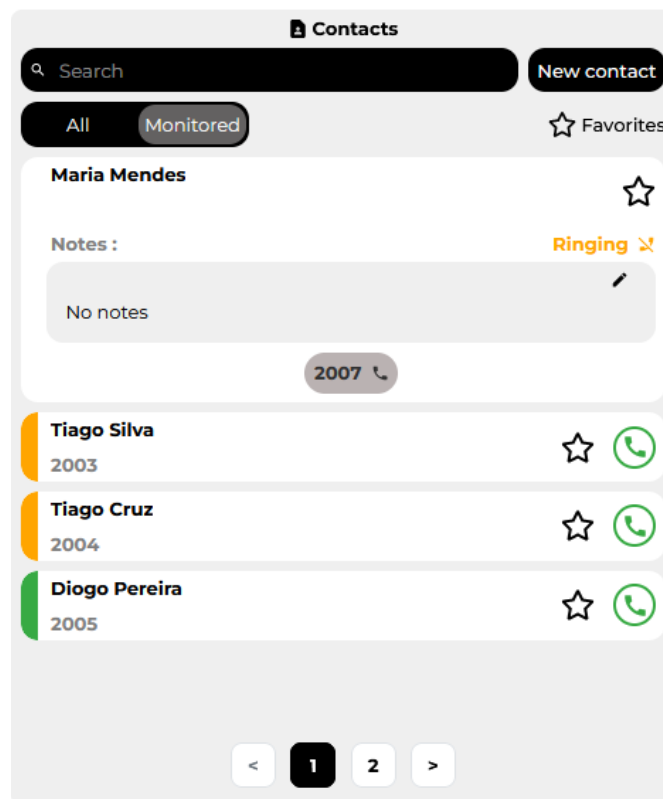


Figura 53: Imagem da interface relativa à área dos contactos monitorados

Relativamente aos contactos monitorados (Figura 53), estes também foram implementados de acordo com o planeado, sendo possível, no modo simulação, gerar um estado aleatório e alterar, de acordo com um intervalo pré-definido, o estado de cada contacto. O objetivo foi simular como a interface se comportaria, caso esta funcionalidade estivesse implementada nas **APIs**. No modo online, como o estado de presença não é suportado, este é inexistente, sendo sempre atribuído um estado de erro. É também removido o filtro *Tipo*, uma vez que neste cenário, os filtros não são aplicáveis.

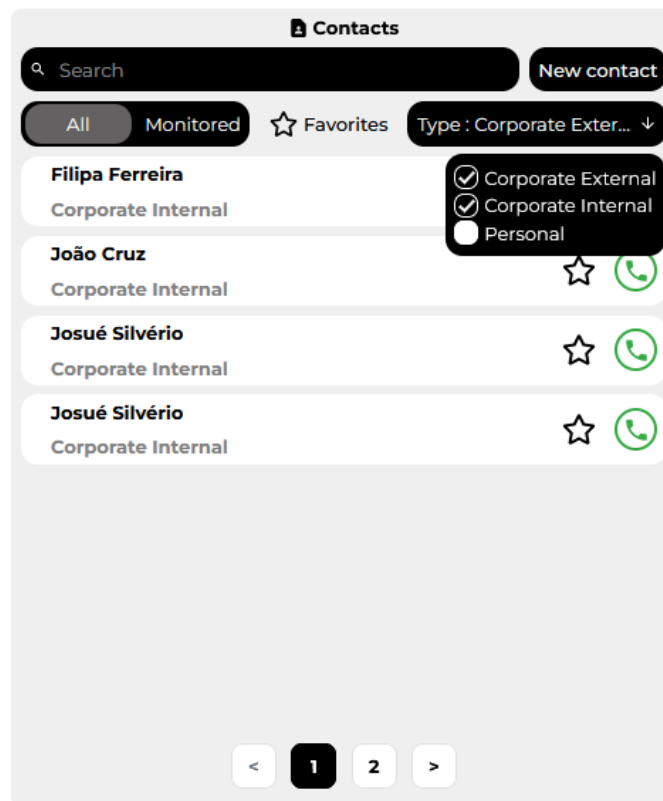


Figura 54: Imagem da interface relativa à área dos contactos com os filtros expandidos

A área do filtro *Tipo* pode ser expandida (Figura 54), o que revela todas as opções desta categoria, sendo que por defeito, todos se encontram seleccionados, de forma a revelar a lista completa. O comportamento desta área é bastante direto, ou seja, caso uma opção se encontre seleccionada (*Corporate External* e *Corporate Internal*), indica que a lista irá conter os contactos que encaixam nessas categorias, e os que não se encontram seleccionados (*Personal*) não estarão na lista.

Para finalizar a área dos contactos, é necessário mencionar o formulário de edição e criação (Figura 55).

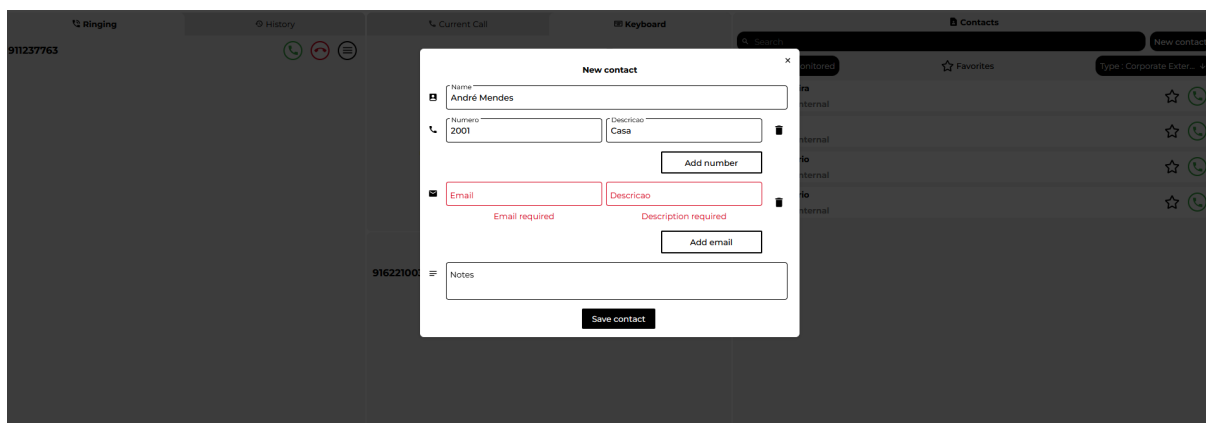


Figura 55: Imagem da interface relativa ao formulário de edição e criação de contactos

Este formulário é apresentado através de uma modal, para que o utilizador consiga visualizar as restantes áreas da interface e não interrompa, por completo, o seu trabalho. Este formulário permite a adição de múltiplos números e emails, assim como fazer o devido tratamento de erros, tal como implementado pela **API**. Este é o mesmo formulário caso o utilizador pretenda criar um novo contacto, editar um já existente ou criar um novo através de um registo ou chamada recebida, variando apenas, para o operador, nos dados iniciais apresentados.

Adicionalmente, é necessário abordar a constituição da área do *Help Online* (ver Figura 56)

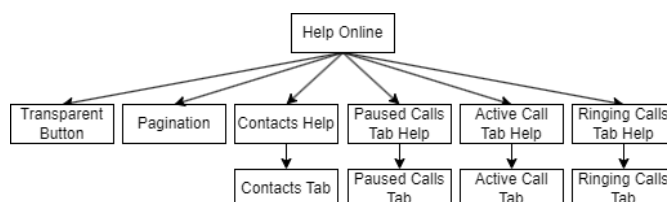


Figura 56: Árvore de componentes relativa ao *Help Online*

Através deste diagrama é possível concluir que esta área possui também um esquema relativamente simples, reutilizando apenas alguns componentes básicos para apresentar as operações possíveis de avançar, recuar ou sair do tutorial. Através da parte mais à direita do diagrama, é possível ver que os componentes utilizados para apresentar as restantes áreas da plataforma são reutilizados nesta área, apesar de não inteiramente. Ou seja, os componentes *Contacts Tab*, *Paused Calls Tab*, *Active Call Tab* e *Ringing Calls Tab* são ligeiramente alterados para que se comportem da forma desejada, mantendo-se assim uma elevada reutilização dos mesmos.

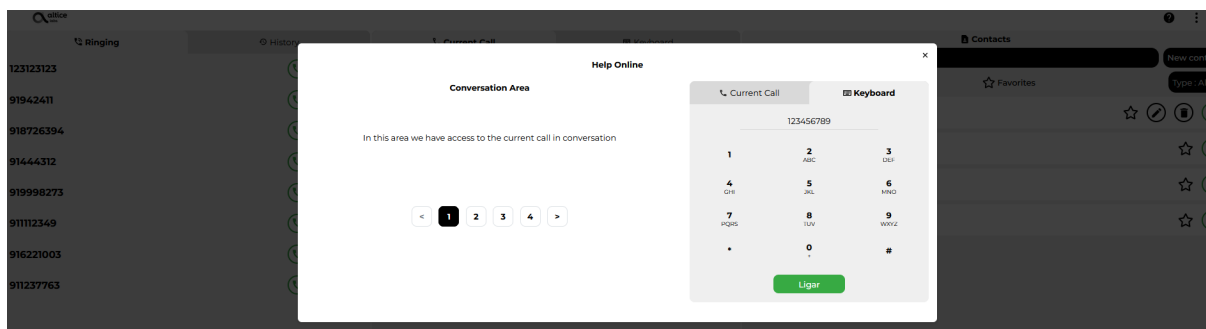


Figura 57: Imagem da interface relativa ao *Help Online*

Através da Figura 57 é possível ver que a interface se manteve bastante fiel ao planeado, sendo utilizada uma página para cada área da consola que necessitasse de explicação. Apesar de não ser possível demonstrar neste documento, cada página possui uma interação pré-programada, de forma a demonstrar ao utilizador as funcionalidades da aplicação. Neste caso, a página selecionada diz respeito à área de chamada em conversação, sendo que a interação pré-programada demonstra o número a ser digitado na caixa de texto (tal como está na figura), iniciando a chamada, o que passa de imediato a *tab* para a chamada em conversação, mostrando esta a ser iniciada e atendida pelo destino.

Para finalizar esta secção, uma das questões que não foi abordada durante o planeamento refere-se à responsividade da aplicação em termos de adaptabilidade, ou seja, à capacidade da interface se adaptar a diferentes dispositivos e resoluções de ecrã [135]. Pretende-se assim que os elementos da página ajustem o seu tamanho e a sua posição de acordo com o tamanho do ecrã. Este aspeto é bastante importante, uma vez que dá a possibilidade aos operadores de utilizarem a plataforma em vários dispositivos, aumentando a flexibilidade de utilização, por exemplo, em casos de emergência, com a possibilidade de utilização da aplicação num dispositivo móvel.

Apesar de ser de importante, é necessário ter em conta que a aplicação desenvolvida tem como objetivo auxiliar o operador a gerir um elevado número de chamadas, e como tal, a eficiência de utilização é bastante importante. Como tal, esta eficiência será consideravelmente superior num dispositivo com maior resolução pois permitirá mais informação ser apresentada ao utilizador e um fluxo mais fácil e direto de operações.

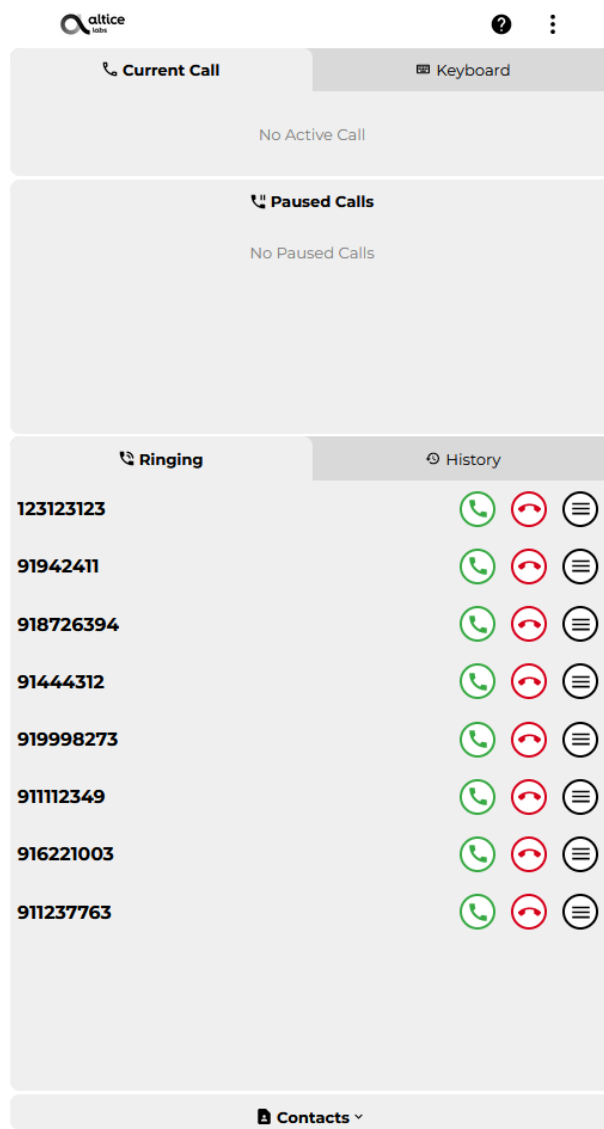


Figura 58: Imagem da interface relativa à aplicação num dispositivo móvel

Através da Figura 58 é possível ver o aspeto da interface quando esta é utilizada num dispositivo móvel. Esta interface foi implementada de forma a que todos os elementos aumentem ou diminuam de tamanho, de acordo com o espaço disponível. No entanto, ao analisar a interface na sua totalidade (Figura 46), é possível dizer que, devido à disposição vertical das várias áreas, esta não possui um desenho adequado para um dispositivo móvel. Consequentemente, quando o utilizador acede à página *web* num dispositivo com uma resolução inferior a seiscentos pixeis, esta passa para uma disposição horizontal, como indicado na figura.

À semelhança do comportamento da interface para dispositivos com maiores resoluções, as diferentes áreas, nesta disposição vertical, dividem o espaço entre si, no entanto, de forma ligeiramente diferente. Como se pode ver na imagem, as diferentes áreas encolhem e crescem de forma a que ocupem o mínimo

espaço necessário (com a exceção de quando há mais espaço disponível do que necessário), de forma a que consigam apresentar toda a informação disponível sem que seja adicionada uma barra de *scroll* dentro de cada área. Isto permite evitar múltiplas barras de *scroll* na aplicação, o que em dispositivos móveis pode acrescentar à dificuldade de utilização. Contudo, isto acontece apenas em casos extremos, quando o operador possui demasiadas chamadas a tocar ou pausadas, de forma a que a altura deste componente não exceda a altura do dispositivo. Desta forma, a aplicação mantém-se simples de utilizar nestes cenários.

Esta implementação é possível pois a área de contactos é, por defeito, minimizada, podendo esta ser expandida pelo operador, de forma a que apresente a lista total de contactos (Figura 59).

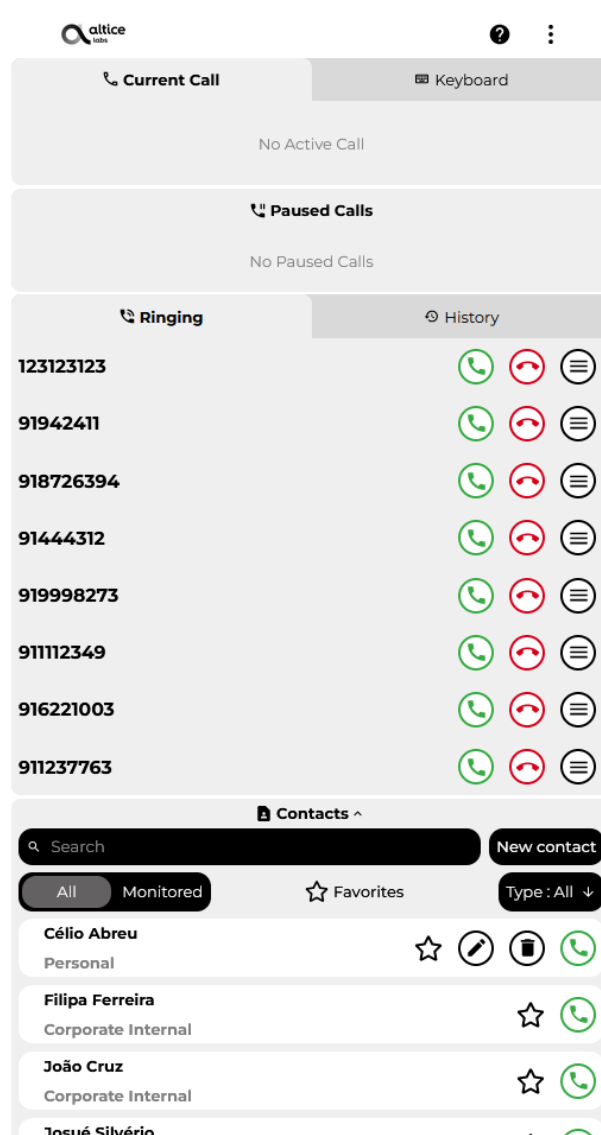


Figura 59: Imagem da interface relativa à aplicação num dispositivo móvel com contactos expandidos

Aqui, como a altura dos componentes excede a altura do dispositivo, é apresentada uma única barra

de *scroll* vertical, que diz respeito ao documento, notando-se também uma pequena redução das outras áreas para alocar espaço para os contactos.

Com este desenho adaptativo adicionado à aplicação, alcança-se uma interface flexível, podendo ser utilizada de forma efetiva em múltiplos dispositivos, implicando uma melhor experiência do utilizador, assim como aumentar a abrangência do público-alvo [136].

4.4 Sumário

Este capítulo tem como objetivo detalhar o processo de desenvolvimento. Este processo teve como base um modelo de desenvolvimento em cascata [137], sendo as áreas da consola desenvolvidas de forma sequencial, de acordo com as prioridades estabelecidas.

Inicialmente são indicadas considerações importantes relativas ao âmbito do trabalho, que inclui uma nova funcionalidade que teve um elevado impacto no protótipo final, relativa ao modo de simulação.

De seguida é exposta a arquitetura de baixo nível do cliente web desenvolvido, explicando a estrutura, o funcionamento, a lógica aplicada e padrões de desenvolvimentos utilizados, sendo os motivos que levaram a estas decisões devidamente indicados.

Por fim são demonstrados os resultados finais da interface através de imagens retiradas do protótipo final, mostrando o seu aspeto final e expondo as várias alterações desta face aos mockups desenvolvidos na fase de planeamento.

No geral, as várias alterações à interface e relativamente às funcionalidades, quando comparadas com o planeado, incluem melhorias do desenho inicial, adição de novas funcionalidades incluindo um novo modo de operação, ou remoção de funcionalidades que, por motivos externos, limitaram ou dificultaram a sua implementação, portanto, o balanço geral do protótipo desenvolvido é positivo.

Capítulo 5

Testes de usabilidade

5.1 Contexto

O objetivo principal da aplicação desenvolvida está relacionado com o auxílio dos operadores de telecomunicações a fazer uma gestão efetiva de um elevado número de chamadas telefônicas. Como tal, é necessário avaliar se o protótipo desenvolvido é capaz de cumprir este objetivo. Para isto, foi realizado um breve teste de usabilidade com o intuito de compreender como o produto é utilizado pelos utilizadores e se permite que estes realizem as tarefas desejadas com eficiência e satisfação [138]. A necessidade destes testes é óbvia, uma vez que permitem avaliar se a plataforma vai de encontro às expectativas dos utilizadores e é utilizado por estes de forma correta, permitindo também obter *feedback* dos utilizadores e, no final, remover potenciais erros e problemas da plataforma[139].

A usabilidade revela-se um fator crucial para qualquer aplicação web, e qualquer produto no geral, tendo fortes implicações na experiência do utilizador, sendo capaz de determinar o sucesso do produto [140], sendo esta importância validada e confirmada nos testes desenvolvidos.

De modo a conduzir estes testes, é necessário decidir o tipo de teste a ser executado. O *Department of Health and Human Services* dos Estados Unidos recomenda vários tipos de testes [141], indicando que o melhor método é escolher um conjunto de participantes que representem um grupo de utilizadores alvo a interagir com um cenário representativo da realidade. Desta forma, é possível recolher dados do sucesso, satisfação e dificuldades dos participantes que sejam o mais exatos possível.

Tendo em conta o contexto em que a aplicação foi desenvolvida, a seleção da metodologia do teste de usabilidade executado foi bastante restritiva, sendo utilizado um *Cognitive Walkthrough* [142]. Este tipo de teste tem como objetivo verificar como a interface suporta um utilizador que está a interagir com esta pela primeira vez e são realizadas diferentes tarefas planeadas.

Estes testes foram realizados com cinco participantes que, como indicado pelo *Nielson Norman Group*, é um número adequado para um elevado número de projetos, permitindo encontrar grande parte dos

problemas de usabilidade [143]. É importante verificar que o nível de experiência dos participantes selecionados com plataformas semelhantes diferiu bastante. Daí que, poderia ser vantajoso considerar um maior número de participantes de forma a criar um grupo para cada nível de experiência (por exemplo, novato e especialista), uma vez que a diferença de experiência mostrou-se como um fator decisivo no sucesso da execução das tarefas. Por motivos de confidencialidade, os participantes selecionados tiveram que ser colaboradores internos da Altice Labs, pelo que foi impossível selecionar um grupo representativo do público-alvo da aplicação.

As tarefas foram definidas de modo a tentar, de certa forma, simular as várias tarefas que seriam realizadas num cenário de utilização real, sendo também efetuada uma breve contextualização sobre o teste antes das tarefas serem expostas ao participante. Estas tarefas foram escritas de forma a que representassem ações específicas realistas suportadas pela aplicação, mas de forma a que não fornecessem qualquer tipo de ajuda e não revelassem como é que esta devesse ser feita [144; 145]. No caso de dificuldade por parte do participante, era fornecida ajuda permitindo que este conseguisse executar a tarefa. Devido às limitações na utilização da consola já referidas anteriormente, foi decidido efetuar estes testes no modo de simulação. Esta decisão permitiu executar as tarefas sem a necessidade de instalação de um *softphone* no dispositivo do participante, uma vez que estes testes foram realizados remotamente, tornando também possível testar as funcionalidades apenas implementadas no modo simulação. Desta forma foi possível utilizar um registo de chamadas pré-definido, assim como uma lista de chamadas a tocar sem ser necessário manipular a *websocket* de outra forma, para que a aplicação recebesse chamadas adicionais.

Durante os testes realizados foram sempre coletados dados qualitativos relativos ao comportamento dos participantes, nomeadamente o sucesso, se necessitaram de algum tipo de orientação para terminar a tarefa, e, no geral, as dificuldades observadas ou indicadas por estes [144].

No final, foi enviado a cada participante um formulário a realizar *online*, anónimo, para que este estivesse totalmente confortável a fornecer *feedback* da aplicação, juntamente com o *feedback* fornecido durante o teste.

5.2 Tarefas

Como foi estabelecido, as tarefas a realizar foram expostas a cada participante de forma a não revelar como é que seria esperado que cada uma fosse realizada. Exemplo disso é a Tarefa 1.1, cujo objetivo é iniciar uma chamada inserindo o número na área do teclado, sendo dado o número a digitar ao participante

por motivos de simplicidade. Foi apenas pedido que este iniciasse uma chamada, não indicando o nome "teclado" que poderia de imediato suscitar o participante a aceder a esta área. Na lista seguinte, são apresentados apenas os objetivos pretendidos na forma de execução da tarefa após esta ser apresentada. Desta forma, é possível ver se a interface é intuitiva e simples de utilizar, uma vez que nenhum dos participantes possuía conhecimento prévio desta aplicação.

Com o objetivo de facilitar a execução das tarefas por parte dos participantes, e como estas foram realizadas no modo simulação, as chamadas efetuadas eram apenas mantidas durante alguns segundos, sendo durante este tempo simulada a execução normal da chamada. Assim, foram planeadas as seguintes tarefas:

Tarefa 1 - Iniciar Chamadas

- 1.1 - Iniciar chamada digitando número no teclado;
- 1.2 - Iniciar chamada através de um contacto com vários números;
- 1.3 - Iniciar chamada através de um contacto com apenas um número;

Tarefa 2 - Gerir Chamadas

- 2.1 - Atender e lidar com várias chamadas recebidas, presentes na lista de chamadas a tocar;
- 2.2 - Alternar entre diferentes chamadas a tocar, pausadas e iniciadas, após uma chamada em conversação, colocar em pausa e iniciar uma outra chamada;

Tarefa 3 - Contactos

- 3.1 - Pesquisar contacto através da barra de pesquisa com o Nome guardado;
- 3.2 - Filtrar contacto por tipo (Corporativo Externo, Corporativo Interno e Pessoais);
- 3.3 - Criar Contacto;
- 3.4 - Editar contacto pessoal;
- 3.5 - Editar contacto corporativo, expandindo o contacto para editar as notas;
- 3.6 - Adicionar contacto ao favorito;
- 3.7 - Apagar contacto;

Tarefa 4 - Gerir Histórico

- 4.1 - Procurar um registo específico no histórico através da data;
- 4.2 - Efetuar chamada para o número desse registo através do menu de contexto;
- 4.3 - Guardar esse número nos contactos através do menu de contexto.

Estas tarefas, como previamente estabelecido, focam-se em tarefas que se prevê que sejam realizadas num cenário do mundo real, mas focam-se também em certas funcionalidades que é previsto que possam dar origem a problemas ou dificuldades por parte do utilizador. Exemplo disto é a Tarefa 3.6, uma vez que não existe uma caixa de diálogo adicional para confirmar a operação, o que pode tornar cliques involuntários bastante inconvenientes, levando à remoção de um contacto acidentalmente. Outro exemplo é a Tarefa 1.3, uma vez que a diferença de comportamento de início de chamada, quando o contacto tem vários números associados, que ao invés de abrir um menu de contexto, inicia de imediato a chamada, podendo levar a chamadas efetuadas acidentalmente.

5.3 Observações

As observações levantadas durante todo o processo de execução de tarefas permitiram levantar retirar conclusões relativamente ao que os participantes, e potencialmente o público alvo, podem achar mais complexo de utilizar. É importante mencionar que todos os participantes possuíam um *background* semelhante, com a exceção do participante 3, que possuía alguma experiência a utilizar produtos de consola de operador semelhantes.

Os resultados obtidos encontram-se compilados na Tabela 3.

Tabela 3: Observações retiradas dos testes efetuados

Participante	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4
Participante 1	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades	Dificuldade para encontrar as notas nos contactos corporativos e notar que é possível expandir os contactos	Dificuldade para encontrar o registo
Participante 2	Dificuldade para encontrar o teclado	Dificuldade ao analisar o estado da chamada em conversação	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades
Participante 3	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades	Dificuldade no formulário de criação de contacto ao interpretar os erros e dificuldade a expandir contactos e notas	Dificuldade para encontrar o registo
Participante 4	Dificuldade ao encontrar o teclado	Tarefa realizada sem dificuldades	Dificuldade a manusear os filtros, o participante não percebeu que era necessário remover depois de estarem aplicados	Tarefa realizada sem dificuldades
Participante 5	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades	Tarefa realizada sem dificuldades

Ao analisar os dados com cuidado é possível ver que, de imediato, a interface pode conter alguns problemas. Foi notada alguma dificuldade a aceder ao teclado. Apesar deste estar num local bastante óbvio, pode ter existido alguma dificuldade a associar o teclado à área onde se insere um número, sendo que uma troca da palavra "teclado" para uma outra que o permitisse identificar mais facilmente poderia diminuir esta dificuldade.

Foram também notadas várias dificuldades na secção dos contactos, nomeadamente a expandir cada contacto, uma vez que o botão que indica que este pode ser expandido apenas é revelado caso o utilizador passe com o cursor por cima. O mesmo acontece ao editar as notas quando o contacto se encontra expandido. Estas dificuldades acontecem por ser a primeira vez que cada participante interagia com a plataforma, mas que seriam facilmente ultrapassadas através da aprendizagem de funcionamento acrescida com alguma experiência de utilização por parte do utilizador. Além disso, foi notada alguma dificuldade a manusear os filtros da lista de contactos, nomeadamente a alterar os filtros depois destes serem seleccionados e, no geral, a seleccionar cada opção. Em particular, verificou-se alguma confusão ao identificar se seleccionar uma opção iria adicionar ou remover essa categoria dos filtros. Novamente, uma dificuldade que facilmente poderia ser ultrapassada depois de algum uso, mas que deveria ser resolvida para eliminar possíveis frustrações de uma utilização inicial.

Por último, algo que também suscitou alguma dificuldade aos participantes foi procurar e encontrar um registo específico no histórico de chamadas. O motivo desta dificuldade é proveniente da longa lista de registo de chamadas colocada para este teste, que incluía chamadas realizadas durante um longo espaço de tempo. Ao procurar o registo, como não está implementado qualquer método de filtração nesta lista, nem paginação, implicou a necessidade de efetuar bastante *scroll* até encontrarem o registo específico de acordo com a data. Este processo seria simplificado se simplesmente fosse possível filtrar por data, ou algum tipo de paginação que permitisse saltar vários registos de uma só vez.

5.4 Questionário

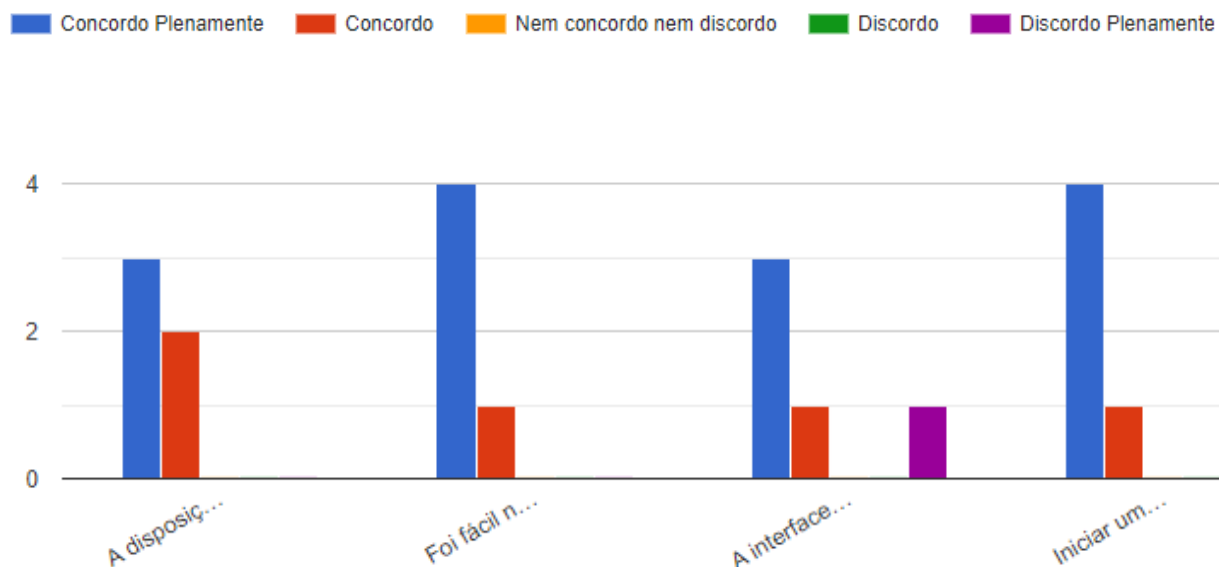
Para finalizar o teste de usabilidade, foi enviado um questionário a cada participante. Este questionário foi criado com recurso ao *Google Forms* [146], com o objetivo permitir o anonimato, evitar a pressão de tempo de resposta, e permitir a cada participante responder de forma livre. Foram colocados dois tipos de questões: questões de escolha múltipla e questões de desenvolvimento. As primeiras questões, de escolha múltipla, são relativas à opinião do utilizador acerca da interface testada, sendo as respostas permitidas entre “Concordo plenamente” e “Discordo plenamente”. Por serem respostas quantitativas, foram compiladas num gráfico que será apresentado mais à frente. As questões de desenvolvimento, permitiu a cada utilizador desenvolver uma opinião relativa à interface ou a áreas específicas desta.

As perguntas de escolha múltipla foram as seguintes :

- A disposição dos menus é intuitiva e fácil de interagir;
- Foi fácil navegar nas diferentes áreas para realizar as diferentes tarefas;

- A interface transmitiu a informação acerca do estado das chamadas de forma clara e simples;
- Iniciar uma chamada através dos contactos foi fácil e intuitivo;
- Pesquisar um contacto foi simples e fácil;
- A interface contribui positivamente para uma busca de contactos efetiva;
- O formulário para criação de um contacto foi simples e compreensível;
- As funcionalidades de edição de contacto funcionam da forma que previa;
- A informação acerca do estado de cada contacto monitorado foi transmitida de forma clara e intuitiva?;
- Foi fácil e direto encontrar uma chamada específica no histórico;
- As operações possíveis nos registos do histórico são claras e intuitivas.

As seguintes respostas foram obtidas, respetivamente:



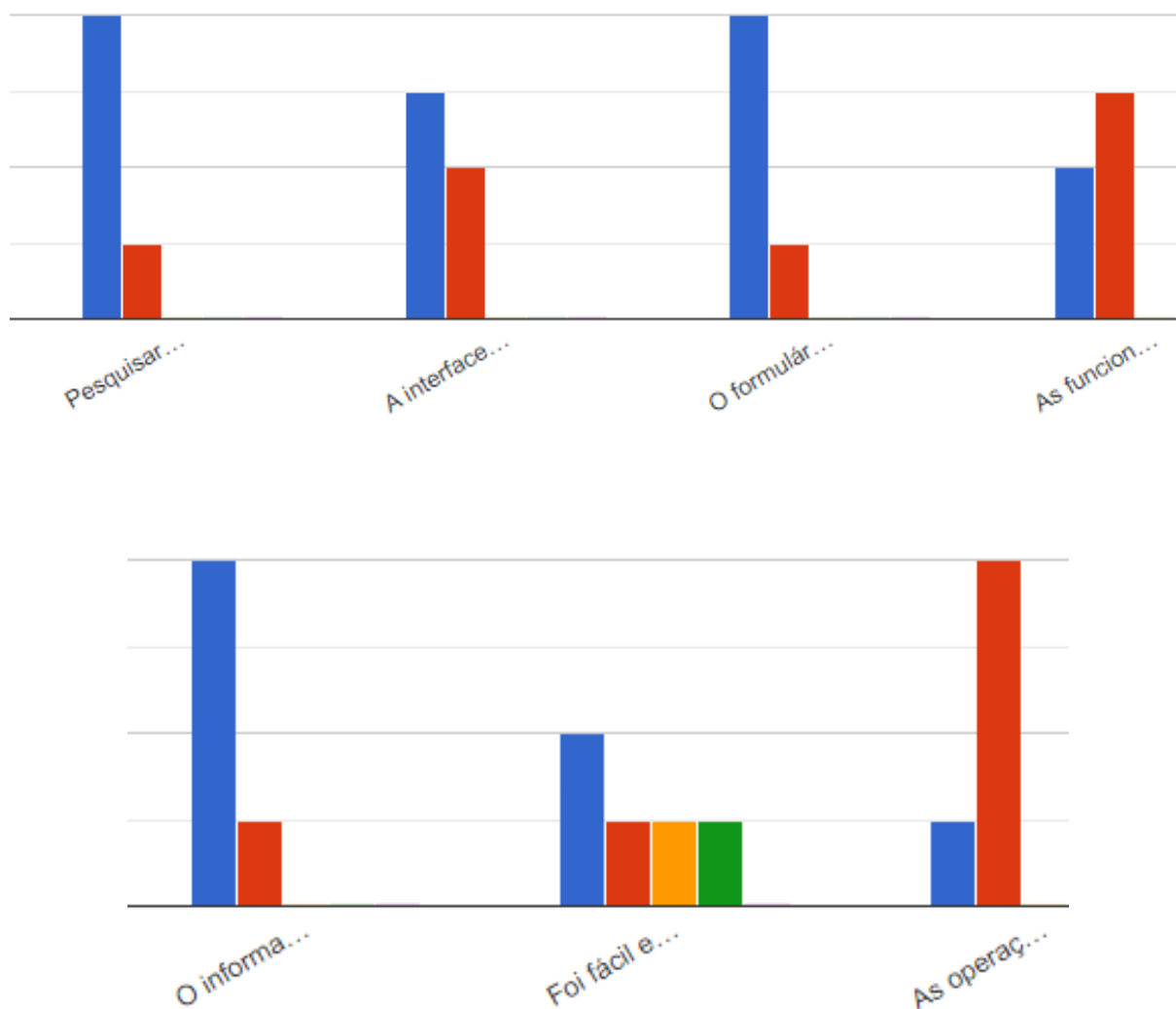


Figura 60: Respostas obtidas as perguntas de escolha múltipla

Após analisar todas as respostas, é possível concluir que, no geral, existe concordância entre os participantes relativamente à interface desenvolvida. No entanto, tal não aconteceu no caso da questão "Foi fácil e direto encontrar uma chamada no registo", o que está de acordo com a explicação já estabelecida na secção 5.3. As dificuldades apresentadas nesta área surgem na presença de uma lista de histórico grande, sendo complicado navegar na lista, uma vez que esta não possui nenhum método de filtragem dos registos. Um método de filtragem de acordo com a data seria suficiente para resolver este problema.

Já a questão "A interface transmitiu a informação acerca do estado da chamada de forma clara e simples" teve uma resposta "Discordo plenamente", considerada uma resposta negativa, indicando que um participante teve dificuldade nesta área, o que já tinha sido observado durante a execução das tarefas, no entanto, não foi claro o motivo desta dificuldade.

Relativamente às questões de desenvolvimento, foram colocadas as seguintes:

- A interface funciona como o esperado?
- Encontrou ou consegue prever alguma dificuldade durante todo este processo ou durante a execução de alguma tarefa?
- Acha que esta plataforma fez um bom trabalho de auxiliar na gestão de um elevado número de chamadas?
- Tem alguma sugestão de alteração que poderia ser feita na interface de modo a melhorar o seu comportamento ou usabilidade?

No geral, estas questões tiveram como objetivo pedir *feedback* geral aos participantes sobre a consola, dando mais liberdade nas respostas com a possibilidade de justificar o motivo das dificuldades encontradas, algo que não era possível nas questões de escolha múltipla.

Resumidamente, as respostas foram, no geral positivas, sendo indicado que a interface é bastante intuitiva e simples de usar. No entanto, foi também indicado que os filtros na área dos contactos não funcionaram como o esperado, tendo sido encontradas dificuldades na sua utilização, especialmente na forma como os filtros aplicados são transmitidos ao utilizador. Adicionalmente, foi também indicado que o registo de chamadas podia introduzir várias dificuldades aos utilizadores, especialmente na presença de uma lista extensa, como já mencionado.

Como sugestões de alterações, foi indicado que a interface poderia beneficiar de uma adição de *tooltips* nos botões que não possuem texto de forma a descrever as operações, como é o caso dos botões das operações nos contactos (editar e eliminar), assim como os botões das operações nas notas de cada contacto (editar e guardar). Esta alteração tornaria estas operações mais claras, uma vez que a utilização de apenas ícones pode levantar algumas dúvidas aos utilizadores. Adicionalmente, foi também sugerido alterar o nome dado à área contendo o teclado, que não é clara ao que se refere.

No geral, as respostas enviadas pelos participantes vão de encontro aos dados coletados durante a execução das tarefas. Este teste de usabilidade, no geral, foi um sucesso, uma vez que permitiu levantar alguns aspetos a melhorar na interface, uns simples e outros mais extensos, de forma a torná-la mais intuitiva e simples de utilizar, e permitiu também averiguar que esta é capaz de auxiliar o operador a gerir um elevado número de chamadas.

De forma a melhorar progressivamente a consola, este processo deveria ser iterativo [141; 147], sendo necessário efetuar alterações para resolver os problemas mencionados e levantados durante todo este processo. Seria adequado realizar novamente um teste, com uma amostra de participantes diferente, mas que, devido à falta de tempo, não foi possível.

5.5 Sumário

Este capítulo documentou os testes de usabilidade a que o protótipo foi submetido, com o objetivo de o colocar à prova e averiguar se este funcionaria conforme o esperado perante atividades do cotidiano.

A abordagem deste teste de usabilidade baseou-se na seleção de cinco participantes com o intuito de realizarem um conjunto de tarefas pré-definidas que se enquadrem num cenário de utilização normal de uma consola de operador, de forma a avaliar se os participantes eram capazes de realizar as tarefas sem dificuldades, e se a interface desenvolvida era capaz de os auxiliar no processo de gestão de chamadas telefónicas.

Foi coletada informação valiosa durante a execução do teste que permitiu identificar aspetos a melhorar no protótipo desenvolvido, indicadas por dificuldades dos participantes na execução de tarefas específicas, assim como sugestões fornecidas por estes durante a execução dessas tarefas.

No final, o questionário realizado pelos participantes permitiu consolidar os resultados obtidos, sendo mais uma vez indicando possíveis aspetos a melhorar, assim como sugestões de alterações que permitiriam melhorar a qualidade da solução final, sendo, no geral, o *feedback* coletado pelos participantes relativo à qualidade do protótipo desenvolvido bastante positivo.

Capítulo 6

Conclusões e trabalho futuro

6.1 Conclusões

Em todo este documento foi analisada a importância de métodos de comunicação efetivos e o impacto negativo que a falta destes pode ter nas organizações. É este elevado impacto que leva a que empresas tentem melhorar os seus processos de comunicação, de forma a manter uma elevada colaboração entre equipas.

É por estes motivos que surgiu a necessidade de melhorar o produto já existente na Altice Labs. Depois de uma fase inicial de tomada de decisões, foi delineado que, tendo em conta o tempo alocado para tal, o mais adequado seria o desenvolvimento de um protótipo de um cliente *web* que funcionasse como consola de operador, utilizando para tal as diferentes **APIs** indicadas para a integração com a telefonia e os contactos.

Durante a fase de investigação, foram aprofundados os conceitos aplicados a esta área, assim como a evolução e as diferentes opções disponíveis no mercado relativo a software semelhante ao que se pretendia desenvolver, indicando as diferenças entre si e entre o produto já desenvolvido pela Altice Labs. Este passo, apesar de ter sido feito numa fase inicial, revelou-se como bastante importante para definir o papel e a relevância do produto no mercado e para definir qual deveria ser o foco do protótipo a desenvolver.

Foi também explicado todo o processo subjacente à escolha da abordagem selecionada, assim como as funcionalidades essenciais que deveriam pertencer à aplicação, de acordo com o ambiente de trabalho e todas as tecnologias aplicadas ao processo de implementação, que foram estudadas e ponderadas de modo a tomar uma decisão acertada.

Posteriormente foi abordado em detalhe todo o processo de planeamento e desenvolvimento, explicando todo o raciocínio aplicado ao comportamento da aplicação. Foi explicado, de forma aprofundada, a arquitetura da camada aplicacional e de visualização, com o aspeto final da interface e as devidas alterações necessárias face ao planeamento.

No geral, é complicado avaliar se o protótipo correspondeu às expectativas e se cumpriu com os objetivos definidos, uma vez que a definição de protótipo é bastante vaga [148]. No entanto, de acordo com as restrições aplicadas ao âmbito do trabalho, provenientes da abordagem selecionada, é de notar que os objetivos principais foram atingidos com este protótipo. Algumas funcionalidades previstas não foram possíveis de realizar como planeado devido à falta de suporte proveniente das **APIs**, tendo sido encontradas alternativas que permitissem, de certa forma, a sua adição ao protótipo. As funcionalidades consideradas de menor prioridade, tais como gravação de chamadas, não foram implementadas uma vez que não houve tempo suficiente que o permitisse.

Além disso, os testes de usabilidade realizados, apesar de identificarem alguns pontos negativos e certos aspetos a melhorar, são compostos por pequenos detalhes que, na grande maioria, eram de fácil resolução. Mesmo assim, os resultados e o *feedback* coletado durante os testes indicam que o protótipo desenvolvido cumpre com as suas funções e objetivos.

Tendo em conta estes dois pontos, é possível concluir que, no final, a aplicação desenvolvida, apesar de ser apenas um protótipo e estar relativamente longe de ser uma aplicação final pronta para produção, cumpre com os objetivos traçados, sendo esta não só uma ferramenta, mas sim um pilar para uma gestão efetiva de um elevado número de chamadas, principalmente no modo simulação. O mesmo se pode dizer acerca do modo online, no entanto, as restrições que são introduzidas pelos serviços integrados implicam alguns problemas, como uma utilização mais complexa, devido à obrigatoriedade de uma ferramenta externa que sirva como *softphone* e a falta de algumas funcionalidades cruciais. No entanto, como a implementação ou resolução destes dois problemas não foi parte do âmbito do projeto, é possível assumir que são fatores externos e que, portanto, o impacto negativo no protótipo não tem um grande peso.

6.2 Perspetiva de trabalho futuro

O trabalho desenvolvido refere-se a um protótipo, e como tal, está longe de ser um produto final e completo, pronto para produção. Por este motivo, há bastante trabalho futuro a ser feito.

Primeiramente, apesar de não serem funcionalidades cruciais, existem algumas suportadas pelas **APIs** que poderiam ser adicionadas ao protótipo, como a gravação de chamadas, que o tornariam numa aplicação mais completa. Esta é a melhoria mais direta e simples de fazer, uma vez que não implicaria nenhuma alteração em nenhuma outra área.

No entanto, existem várias outras melhorias, com um maior impacto, que podem ser efetuadas. Como visto em várias secções deste documento, as **APIs** integradas para as várias funcionalidades implementa-

das introduziram várias restrições. Como tal, estas **APIs** deveriam ser melhoradas de forma a incluir as funcionalidades que foram definidas como cruciais, mas não suportadas. Estas alterações, obviamente, iriam também implicar alterações na consola, de forma a que suportasse estas operações no modo on-line. Estas funcionalidades incluem o pausar, resumir e atender chamadas, mas também a monitoria de contactos, que, como foi visto, é apenas suportado para um utilizador e deveria ser extendido à lista de contactos completa.

Estando estas alterações efetuadas, seria obtida uma consola que, em termos de funcionalidades, estaria ao nível da consola atual da Altice Labs, no entanto, com uma arquitetura bem definida, com tecnologias recentes e como uma interface nova e de acordo com os *standards* de hoje em dia.

No entanto, como foi visto na secção do estado de arte, esta consola ainda se encontra relativamente longe de outros competidores, sendo que seria vantajoso melhorar a solução desenvolvida, nomeadamente, com um *softphone* integrado, o que permitiria diminuir significativamente a complexidade de utilização, e outras funcionalidades consideradas como relevantes.

Além de todas estas adições, existem também algumas alterações que foram levantadas ou sugeridas no seguimento dos testes de usabilidade realizados, que, apesar de não serem alterações significativas, contribuiriam de forma positiva para a aplicação. Como foi explicado no capítulo anterior, apesar de os testes terem sido realizados com um número de participantes considerado suficiente, seria bastante vantajoso considerar um número maior de participantes com diferentes níveis de experiência relativamente a produtos desta área, assim como a realização de novos testes após a implementação das melhorias listadas.

Referências Bibliográficas

- [1] 8x8 University. 8x8 frontdesk. Accessed: 7-9-2022. URL: <https://www.8x8.com/university/catalog/8x8-frontdesk>.
- [2] Landis Technologies LLC. Attendant console for microsoft teams. Accessed: 07-09-2023. URL: <https://landistechnologies.com/microsoft-teams-attendant-console/>.
- [3] React team and external contributors. Passing data deeply with context. Accessed: 29-08-2023. URL: <https://react.dev/learn/passing-data-deeply-with-context>.
- [4] Bilel Msekni. Modern frontend architecture. 2019. URL: <https://medium.com/js-dojo/modern-front-end-architecture-103-92dd1cc3f26c>.
- [5] Sheridan Smalley. Grammarly and harris poll research estimates u.s. businesses lose \$1.2 trillion annually to poor communication | business wire, 1 2022. Accessed: 1-10-2022. URL: <https://www.businesswire.com/news/home/20220125005525/en/Grammarly-and-Harris-Poll-Research-Estimates-U.S.-Businesses-Lose-1.2-Trillion-Annually-to-Poor-Communication>.
- [6] Agência Lusa. Covid-19. sns 24 atendeu mais de um milhão de chamadas em janeiro – observador. 2 2021. Accessed: 30-12-2022. URL: <https://observador.pt/2021/02/17/covid-19-sns-24-atendeu-mais-de-um-milhao-de-chamadas-em-janeiro/>.
- [7] Mary McMahon. What is an attendant console? (with picture), 12 2022. Accessed: 12-09-2022. URL: <https://www.wise-geek.com/what-is-an-attendant-console.htm>.
- [8] See plans and pricing for 8x8 integrated contact center, voice, video, chat, and apis. | 8x8. Accessed: 05-01-2023. URL: <https://www.8x8.com/products/plans-and-pricing>.
- [9] Michaela Goss. What is a call center? everything you need to know, 11 2021. Accessed: 28-12-2022. URL: <https://www.techtarget.com/searchcustomerexperience/definition/Call-Center>.

- [10] Noah Gans, Ger Koole, and Avishai Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing & Service Operations Management*, 5(2):79–141, 2003. arXiv:<https://doi.org/10.1287/msom.5.2.79.16071>, doi:10.1287/msom.5.2.79.16071.
- [11] Morteza Saberi, Omar Khadeer Hussain, and Elizabeth Chang. Past, present and future of contact centers: a literature review. *Business Process Management Journal*, 23(3):574–597, 2017.
- [12] Carolina Ignaczuk. Contact center: o que é e como funciona na prática? - movidesk, 6 2020. Accessed: 29-12-2022. URL: <https://conteudo.movidesk.com/contact-center/>.
- [13] What is an attendant console? | mida solutions. Accessed: 12-10-2022. URL: <https://www.midasolutions.com/knowledge-hub/attendant-console/>.
- [14] 8x8. note = Accessed: 31-10-2022,. URL: <https://www.8x8.com/products/plans-and-pricing>.
- [15] Kate Vogel. Softphone: What is it and why should you use it? | ringcentral. 2022. Accessed: 28-12-2022. URL: <https://www.ringcentral.com/us/en/blog/softphone/>.
- [16] Rob Scott. What is unified communications? - uc today, 2020. Accessed: 12-01-2023. URL: <https://www.uctoday.com/unified-communications/what-is-unified-communications/>.
- [17] About us - uc today. Accessed: 05-01-2023. URL: <https://www.uctoday.com/about-us/>.
- [18] Alissa Irei, Jessica Scarpati, and Ben Lutkevich. What is unified communications (uc)?, 2021. Accessed: 12-01-2023. URL: <https://www.techtarget.com/searchunifiedcommunications/definition/unified-communications>.
- [19] Vodafone one net manual de utilizador attendant console, 2018.
- [20] One net operator console help. Accessed: 03-01-2023. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-getting-started>.
- [21] Vodafone One Net Help Support. Parking calls. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-parking-calls>.

- [22] Vodafone One Net Help Support. Managing call history. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-managing-call-history>.
- [23] Vodafone One Net Help Support. Managing queues and hunting groups. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-managing-queues>.
- [24] Vodafone One Net Help Support. Call queues. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/feature-management-portal/administrator/call-queues>.
- [25] Vodafone One Net Help Support. Managing conference calls. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-conference-calls>.
- [26] Vodafone One Net Help Support. Managing contacts. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-managing-contacts>.
- [27] Vodafone One Net Help Support. Personal contacts. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-personal-contacts>.
- [28] Vodafone One Net Help Support. Managing the error log. Accessed: 1-11-2022. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-managing-error-log>.
- [29] What is crm? - salesforce emea. Accessed: 28-12-2022. URL: <https://www.salesforce.com/eu/learning-centre/crm/what-is-crm/>.
- [30] Integrated contact center, voice, video, chat, and apis. | 8x8. Accessed: 05-01-2023. URL: <https://www.8x8.com/products>.
- [31] Ringcentral plans pricing: Mvp, video, and contact center. URL: <https://www.ringcentral.com/office/plansandpricing.html>.

- [32] 8x8 Support. Call flip: How do i transfer a call from my desk phone to another device? Accessed: 5-11-2022. URL: <https://support.8x8.com/devices-accessories/phones/general-phone-settings/how-to-transfer-call-from-desk-phone-to-another-device-call-flip>.
- [33] RingCentral. Call flip. Accessed: 5-11-2022. URL: <https://www.ringcentral.com/office/features/call-flip/overview.html>.
- [34] How do call queues work? - 8x8 support, 4 2022. Accessed: 05-01-2023. URL: <https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/call-queues/How-do-Call-Queues-work>.
- [35] How to add members to call queues - 8x8 support, 11 2022. Accessed: 07-01-2023. URL: <https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/call-queues/how-to-add-members-to-call-queues>.
- [36] How to enable agents to log in and out of call queues in the admin console - 8x8 support, 1 2021. Accessed: 07-01-2023. URL: <https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/call-queues/how-to-enable-agents-to-log-in-and-out-of-call-queues-in-admin-console>.
- [37] How to change call queue call forwarding rules - 8x8 support, 10 2022. Accessed: 08-01-2023. URL: https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/call-queues/How_to_Change_Call_Queue_Call_Forwarding_Rules.
- [38] RingCentral. Call queue – definition of call queuing and its role in your call center. Accessed: 5-11-2022. URL: <https://www.ringcentral.com/call-queueing.html>.
- [39] Barge-monitor-whisper (bmw) dial codes - 8x8 support, 1 2022. Accessed: 06-01-2023. URL: [https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/barge-monitor-whisper-groups/Barge-Monitor-Whisper_\(BMW\)_Dial_Codes](https://support.8x8.com/cloud-phone-service/voice/admin-console/work-group-settings/barge-monitor-whisper-groups/Barge-Monitor-Whisper_(BMW)_Dial_Codes).
- [40] RingCentral. What is call monitoring? Accessed: 5-11-2022. URL: <https://www.ringcentral.com/call-monitoring.html>.

- [41] 8x8 Support. Hot desk faq. Accessed: 5-11-2022. URL: <https://support.8x8.com/business-phone/voice/admin-console/work-group-settings/hot-desk/hot-desk-faq>.
- [42] RingCentral. Hot desking. Accessed: 5-11-2022. URL: <https://www.ringcentral.com/office/features/hot-desking/overview.html>.
- [43] Enabling voicemail transcription for your company in 8x8 admin console - 8x8 support, 7 2022. Accessed: 10-01-2023. URL: <https://support.8x8.com/cloud-phone-service/voice/admin-console/setup/company-settings/enabling-voicemail-transcription-for-your-company-in-admin-console>.
- [44] How to listen to voicemail - 8x8 support, 11 2020. Accessed: 09-01-2023. URL: <https://support.8x8.com/cloud-phone-service/voice/voice-administration-account-manager/phone-system/voicemail/how-to-listen-to-voicemail>.
- [45] RingCentral. Voicemail for business. Accessed: 5-11-2022. URL: <https://www.ringcentral.com/office/features/voicemail/overview.html>.
- [46] 8x8 Work for Desktop User Help. 8x8 work team messaging. Accessed : 10-11-2022. URL: <https://docs.8x8.com/8x8WebHelp/8x8-work-for-desktop/Content/workd/use-team-messaging.htm>.
- [47] RingCentral. Conversations made better with team chat and messaging. Accessed: 11-11-2022. URL: <https://www.ringcentral.com/teams/overview.html>.
- [48] 8x8 Support. How do i edit my auto attendant schedule in 8x8 admin console? Accessed: 15-11-2022. URL: https://support.8x8.com/business-phone/voice/admin-console/phone-system-settings/auto-attendants/edit_my_Auto_Attendant_Schedule_in_8x8_Admin_Console.
- [49] 8x8 Support. How do i set up my auto attendant in 8x8 admin console? Accessed: 15-11-2022. URL: <https://support.8x8.com/business-phone/voice/admin-console/phone-system-settings/auto-attendants/how-do-I-set-up-auto-attendant-admin-console>.

- [50] RingCentral. Multi-level auto-attendant. Accessed: 11-11-2022. URL: <https://www.ringcentral.com/office/features/multi-level-auto-attendant/overview.html>.
- [51] 8x8 Support. How to set up call forwarding in 8x8 admin console. Accessed: 20-11-2022. URL: <https://support.8x8.com/business-phone/voice/admin-console/setup/users/how-to-set-up-call-forwarding-8x8-admin-console>.
- [52] RingCentral. Call forwarding. Accessed: 20-11-2022. URL: <https://www.ringcentral.com/office/features/call-forwarding/overview.html>.
- [53] 8x8. 8x8 integrations | 8x8 technology partner ecosystem. Accessed: 25-11-2022. URL: <https://www.8x8.com/products/integrations>.
- [54] RingCentral. Ring central app gallery. Accessed: 20-11-2022. URL: <https://www.ringcentral.com/apps/>.
- [55] 8x8. 8x8 work for salesforce. Accessed: 25-11-2022. URL: <https://www.8x8.com/products/integrations/work-for-salesforce>.
- [56] RingCentral. Ringcentral for salesforce. Accessed: 20-11-2022. URL: <https://www.ringcentral.com/apps/salesforce>.
- [57] RingCentral. Ringcentral analytics portal | unlock new insights with powerful analytics. Accessed: 22-11-2022. URL: <https://www.ringcentral.com/analytics.html>.
- [58] 8x8. Business phone analytics. Accessed: 22-11-2022. URL: <https://www.8x8.com/products/business-phone/analytics>.
- [59] About - landis technologies llc. Accessed: 03-01-2023. URL: <https://landistechnologies.com/about/>.
- [60] Landis Technologies LLC. Landis attendant console for microsoft teams is now available. Accessed: 7-12-2022. URL: <https://landistechnologies.com/landis-attendant-console-for-ms-teams-now-available/>.
- [61] Ring Central. Ringcentral for microsoft teams. Accessed: 10-12-2022. URL: https://www.ringcentral.com/apps/microsoft-teams-dialer?utm_source=app-gallery&utm_campaign=featured.

- [62] David Curry. Most popular apps (2022) - business of apps, 1 2023. Accessed: 04-01-2023. URL: <https://www.businessofapps.com/data/most-popular-apps/>.
- [63] 8x8. 8x8 voice for microsoft teams. Accessed: 10-12-2023. URL: <https://www.8x8.com/products/integrations/8x8-voice-for-microsoft-teams>.
- [64] Ringcentral embedded dialer integration for microsoft teams | ringcentral app gallery. Accessed: 20-01-2023. URL: <https://www.ringcentral.com/apps/microsoft-teams-dialer>.
- [65] Ringcentral cloud pbx for microsoft teams | ringcentral app gallery. Accessed: 25-01-2023. URL: <https://www.ringcentral.com/apps/cloud-pbx-for-microsoft-teams>.
- [66] 8x8 voice for microsoft teams | 8x8 teams integration | 8x8. Accessed: 25-01-2023. URL: <https://www.8x8.com/products/integrations/8x8-voice-for-microsoft-teams>.
- [67] 8x8 voice for microsoft teams. Accessed: 25-01-2023. URL: <https://appsource.microsoft.com/en-us/product/office/8x8inc-5383858.8x8-voice-for-microsoft?tab=overview>.
- [68] Landis attendant console for microsoft teams fully ga among first to showcase new microsoft api | landis technologies llc. 1 2022. Accessed:15-01-2023. URL: <https://landistechnologies.com/landis-attendant-console-for-teams-is-now-generally-available-enabling-best-atte>
- [69] Landis Technologies LLC. Call handling. Accessed: 13-12-2022. URL: <https://ac.docs.landis.cloud/daily-usage/call-handling>.
- [70] Landis Technologies LLC. Enable chat consult transfer (preview). Accessed: 07-09-2023. URL: <https://ac.docs.landis.cloud/appendix/enable-chat-consult-transfer>.
- [71] Landis Technologies LLC. Contact card. Accessed: 07-09-2023. URL: <https://ac.docs.landis.cloud/daily-usage/contact-card>.
- [72] Bryan Turner. Best business phone systems of 2023 | techradar, 2022. Accessed: 25-01-2023. URL: <https://www.techradar.com/best/best-business-phone-system>.
- [73] Sarah Shelton, Anne Nagro, Bryce Colburn, and Lauren Swift. Best business phone systems of 2023 | u.s. news, 2022. Accessed: 12-01-2023. URL: <https://www.usnews.com/360-reviews/business/business-phone-systems>.

- [74] Robert C. Martin. *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, 2008.
- [75] Cross platform integrations. Accessed: 22-04-2023. URL: <https://www.imagicle.com/en/integration/other-cross-platform/>.
- [76] Mida Solutions, via San Crispino, 46 35129 - Padua Italy. *Mida Products Compatibility*, 3.3 edition.
- [77] G Kavitha, K T Kanya Kumari, and D R Santhosh Kumar. Internet protocol private branch exchange. volume 1, 2009. URL: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=053cfaca4e3ef3333d29ef154400cedae8394d1>.
- [78] Bitrix24 tools. Accessed: 11-10-2022. URL: <https://www.bitrix24.eu/tools/>.
- [79] Contact center, all-in-one communication platform. Accessed: 15-10-2022. URL: https://www.bitrix24.eu/solutions/tool/contact_center.php#scrolled.
- [80] Bitrix24 for developers. Accessed: 13-03-2023. URL: <https://www.bitrix24.eu/apps/dev.php>.
- [81] Stacy Smith. Connect sip pbx using rest api, 6 2023. Accessed: 13-03-2023. URL: <https://helpdesk.bitrix24.com/open/17454078/>.
- [82] Bitrix24 pricing. Accessed: 15-10-2022. URL: <https://www.bitrix24.eu/prices/>.
- [83] Eino Nieminen. Ringcentral telephony integration app. Accessed: 27-10-2022. URL: <https://helpdesk.bitrix24.com/open/13540808/>.
- [84] Stacy Smith. Examples of webhooks for developers. Accessed: 13-03-2023. URL: <https://helpdesk.bitrix24.com/open/12357038/>.
- [85] Telephony integration tips. Accessed: 13-03-2023. URL: <https://www.bitrix24.eu/apps/webhooks.php>.
- [86] Voice operator panel - professional softphone for operators and receptionists. Accessed: 11-10-2023. URL: <http://www.voiceoperatorpanel.com>.
- [87] Voice operator panel - faq. URL: <http://www.voiceoperatorpanel.com/faq/>.
- [88] Amjad Hudaib, Raja Masadeh, Mais Haj Qasem, and Abdullah Alzaqebah. Requirements prioritization techniques comparison. *Modern Applied Science*, 12, 01 2018. doi:10.5539/mas.v12n2p62.

- [89] RayRay. You don't need a javascript framework. 2020. Accessed: 20-07-2023. URL: <https://betterprogramming.pub/you-dont-need-a-javascript-framework-df2a36c2dd0a>.
- [90] MDN Contributors. Understanding client-side javascript frameworks, 2023. Accessed: 15-08-2023. URL: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks.
- [91] MDN Contributors. Spa (single-page application). <https://developer.mozilla.org/en-US/docs/Glossary/SPA>, 2023. Accessed: 15-08-2023.
- [92] Madhuri A Jadhav, Balkrishna R Sawant, and Anushree Deshmukh. Single page application using angularjs. *International Journal of Computer Science and Information Technologies*, 6(3):2876–2879, 2015.
- [93] Mochammad Fariz Syah Lazuardy and Dyah Anggraini. Modern front end web architectures with react.js and next.js. *Research Journal of Advanced Engineering and Science*, 7(1):132–141, 2022.
- [94] Morgan Persson. Javascript dom manipulation performance: Comparing vanilla javascript and leading javascript front-end frameworks (dissertation). 05 2020. URL: <https://urn.kb.se/resolve?urn=urn:nbn:se:bth-19531>.
- [95] Devographics. State of javascript, 2022. Accessed: 23-11-2022. URL: <https://stateofjs.com/en-US>.
- [96] Devographics. The state of js 2021, 2022. Accessed: 23-11-2022. URL: <https://2021.stateofjs.com/en-US/>.
- [97] Costanza Tagliaferri. How many software developers are there in the world? 2023. URL: <https://distantjob.com/blog/how-many-developers-are-in-the-world/>.
- [98] Gavin Bierman, Martín Abadi, and Mads Torgersen. Understanding typescript. In *ECOOP 2014–Object-Oriented Programming: 28th European Conference, Uppsala, Sweden, July 28–August 1, 2014. Proceedings 28*, pages 1–27. Springer, 2014.
- [99] David Pogue. 5 of the worst user-interface disasters. 5 2016. Accessed: 07-07-2023. URL: <https://www.scientificamerican.com/article/pogue-5-of-the-worst-user-interface-disasters/>.

- [100] Mark Woodroffe Shailey Minocha Debbie Stone, Caroline Jarrett. *User Interface Design and Evaluation*. Morgan Kaufmann / The Open University, 09 2014. URL: https://www.researchgate.net/publication/43642930_User_Interface_Design_and_Evaluation.
- [101] Thomas Metz. The real effects of bad web design. Accessed: 15-07-2023. URL: <https://usabilitygeek.com/real-effects-bad-web-design/>.
- [102] Oshyn. 8 real dangers of poor user experience. 2023. Accessed: 15-07-2023. URL: <https://www.oshyn.com/blog/8-real-dangers-of-poor-user-experience>.
- [103] hexacta Design Team. How good ui design benefits your business. Accessed: 15-07-2023. URL: <https://www.hexacta.com/how-ui-design-benefits-your-business/>.
- [104] A. L. Dzyubenkob V. N. Lukina and Yu. B. Chechikov. Approaches to user interface development. *Programming and Computer Software*, 2020.
- [105] Contact center. voice. video. chat. together | 8x8. Accessed: 30-01-2023. URL: <https://www.8x8.com>.
- [106] Message. video. phone | ringcentral. Accessed: 30-01-2023. URL: <https://www.ringcentral.com>.
- [107] Introducing one net operator console. Accessed: 15-01-2023. URL: <https://onenet.vodafone.com/latest/uk/en/content/topics/cf/operator-console/operator-console-introducing-operator-console>.
- [108] Imagicle : Your communications faster, smarter, easier. Accessed: 15-07-2023. URL: <https://www.imagicle.com/en/>.
- [109] Voice operator panel - professional softphone and attendant console for operators and receptionists. Accessed: 15-01-2023. URL: <http://www.voiceoperatorpanel.com>.
- [110] Marc Rettig. Nobody reads documentation. *Commun. ACM*, 34(7):19–24, jul 1991. doi:10.1145/105783.105788.
- [111] Charles J. Welty. Usage of and satisfaction with online help vs. search engines for aid in software use. In *Proceedings of the 29th ACM International Conference on Design of Communication*, SIGDOC '11, page 203–210, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/2038476.2038516.

- [112] Wyzowl. Customer onboarding statistics 2020. Accessed: 25-01-2023. URL: <https://www.wyzowl.com/customer-onboarding-statistics/>.
- [113] Filip Stål. Interactive user onboarding and its effect on activation rates: A statistical study of feature introductions in applications with complex interfaces (dissertation), 2020.
- [114] Samantha Ferguson. 5 amazing walkthrough examples for apps and websites. 2023. Accessed: 27-01-2023. URL: <https://www.wyzowl.com/walkthrough-examples/>.
- [115] David Garlan, Felix Bachmann, James Ivers, Judith Stafford, Len Bass, Paul Clements, and Paulo Merson. *Documenting Software Architectures: Views and Beyond*. Addison-Wesley Professional, 2010.
- [116] Software Engineering Institute. Software architecture. URL: <https://www.sei.cmu.edu/our-work/software-architecture/>.
- [117] Roger S Pressman. *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.
- [118] Monika Nožinić. How to create powerful frontend architecture for your website. 2023. Accessed: 05-08-2023. URL: https://www.asyncclabs.co/blog/software-development/how-to-create-powerful-frontend-architecture-for-your-website/#Benefits_of_a_well-built_frontend_architecture%20re.
- [119] Hiren Dhaduk. Frontend architecture and how to improve its design. 2023. Accessed: 05-08-2023. URL: <https://www.simform.com/blog/frontend-architecture/>.
- [120] Kevin Pennekamp. How to create a scalable and maintainable frontend architecture. 2021. Accessed: 05-08-2023. URL: <https://dev.to/kevtiq/how-to-create-a-scalable-and-maintainable-front-end-architecture-4f47>.
- [121] Carlos Santana Roldan. *React 17 Design Patterns and Best Practices - Third Edition*. Packt, 2021. Accessed: 07-08-2023.
- [122] Cory Gackenhaimer. *What Is React?* Apress, Berkeley, CA, 2015. doi:10.1007/978-1-4842-1245-5_1.
- [123] Murray Silverstein Christopher Alexander, Sara Ishikawa. *A Pattern Language*. Oxford University Press, 1977.

- [124] James Coplien. Software design patterns. pages 1604–1606, 01 2003.
- [125] Saurabh Barot. React design patterns : You should know in 2023. 2023. Accessed: 20-08-2023. URL: <https://aglowiditsolutions.com/blog/react-design-patterns/>.
- [126] Lawrence Eagles. A guide to react design patterns. 2022. Accessed: 15-08-2023. URL: <https://blog.logrocket.com/react-design-patterns/#provider-pattern>.
- [127] Morten Barklund. React architecture: The react provider pattern. 2020. Accessed: 16-08-2023. URL: <https://mortenbarklund.com/blog/react-architecture-provider-pattern/>.
- [128] React team and external contributors. Extracting state logic into a reducer. Accessed: 29-08-2023. URL: <https://react.dev/learn/extracting-state-logic-into-a-reducer#consolidate-state-logic-with-a-reducer>.
- [129] Redux Team. Redux style guide, 2022. Accessed: 30-08-2023. URL: <https://redux.js.org/style-guide/#reducers-must-not-have-side-effects>.
- [130] Dom tree. 2022. Accessed: 31-08-2023. URL: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction.
- [131] MDN Contributors. Introduction to the dom. Accessed: 31-08-2023. URL: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction.
- [132] Sanchit Aggarwal et al. Modern web-development using reactjs. *International Journal of Recent Research Aspects*, 5(1):133–137, 2018.
- [133] Getting started | axios docs. Accessed: 30-08-2023. URL: <https://axios-http.com/docs/intro>.
- [134] Kristiina Karvonen. The beauty of simplicity. In *Proceedings on the 2000 Conference on Universal Usability*, CUU '00, page 85–90, New York, NY, USA, 2000. Association for Computing Machinery. doi:10.1145/355460.355478.
- [135] Amy Schade. Responsive web design (rwd) and user experience. 2014. Accessed: 02-09-2023. URL: <https://www.nngroup.com/articles/responsive-web-design-definition/>.

- [136] Brett S Gardner. Responsive web design: Enriching the user experience. *Sigma Journal: Inside the Digital Ecosystem*, 11(1):13–19, 2011. Accessed: 02-09-2023.
- [137] Adobe Communications Team. Waterfall methodology: A complete guide. Accessed: 10-10-2023. URL: <https://business.adobe.com/blog/basics/waterfall#:~:text=The%20Waterfall%20methodology%20-%20also%20known,before%20the%20next%20phase%20begins>.
- [138] ISO. Iso 9241-11:1998(en), last visited 14/05/2015. 1998. URL: <https://www.iso.org/obp/ui/iso:std:iso:9241:-11:ed-1:v1:en>.
- [139] Quovantis. Why is it important to do usability testing. 2017. Accessed: 05-09-2023. URL: <https://uxplanet.org/why-is-it-important-to-do-usability-testing-5080a5640df3>.
- [140] Emilio Insfran and Adrian Fernandez. A systematic review of usability evaluation in web development. In Sven Hartmann, Xiaofang Zhou, and Markus Kirchberg, editors, *Web Information Systems Engineering – WISE 2008 Workshops*, pages 81–91, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
- [141] U.S. Department of Health and Human Services. *Research-Based Web Design Usability Guidelines: Current Research-Based Guidelines on Web Design and Usability Issues*. U.S. Government Printing Office, Washington, D.C., 2006 edition edition, 2006. URL: <http://www.usability.gov/pdfs/guidelines.html>.
- [142] Julie Rinder. The importance of website usability testing. 2012. URL: <https://api.semanticscholar.org/CorpusID:56955596>.
- [143] Jakob Nielsen. How many test users in a usability study? 2021. Accessed: 05-09-2023. URL: <https://www.nngroup.com/articles/how-many-test-users/>.
- [144] Kate Moran. Usability testing 101. 2019. Accessed: 05-09-2023. URL: <https://www.nngroup.com/articles/usability-testing-101/>.
- [145] Marieke McCloskey. Turn user goals into task scenarios for usability testing. 2014. Accessed: 05-09-2023. URL: <https://www.nngroup.com/articles/task-scenarios-usability-testing/>.

- [146] 2023. Accessed: 05-09-2023. URL: <https://www.google.com/forms/about/>.
- [147] User Pilot. How does iterative testing help build better products [+test examples]. 2022. Accessed: 05-09-2023. URL: <https://userpilot.com/blog/iterative-testing/>.
- [148] Virginia Ramírez. What is a prototype? 2018. Accessed: 05-08-2023. URL: <https://medium.com/nyc-design/what-is-a-prototype-924ff9400cfd>.

