INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Aplicação Web/Mobile Chat - Novo Chat Bla.AEIOU

JORGE MIGUEL DE CASTRO PINHEIRO
Outubro de 2022

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Aplicação Web/Mobile Chat - Novo Chat Bla.aeiou

Jorge Miguel de Castro Pinheiro

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software

Orientador: Ricardo Almeida

Júri:

Presidente:

Vogais:

Porto, Outubro 2022

Dedicatória

A realização deste projeto é especialmente dedicada aos meus pais, irmão e namorada que sempre me apoiaram, motivaram e possibilitaram com tudo durante todo o percurso académico e com a oportunidade que me proporcionaram para me formar.

Jorge Pinheiro

Resumo

Este projeto foi desenvolvido num ambiente empresarial, onde o autor teve a possibilidade de realizar um estágio profissional, a qual lhe foi dada a oportunidade de desenvolver o projeto, na organização designada Cool Beans, Lda. Este projeto está inserido no ramo de desenvolvimento de uma aplicação web/mobile de chat, e este baseia-se na ideia de atualizar e melhorar o website desenvolvido anteriormente para uma versão atualizada e com mais funcionalidades do que o antigo chat.

A presente dissertação tem como objetivo documentar todo o processo de implementação de um protótipo de framework React, juntamente com Firebase, bem como novas funcionalidades, para que estas possam posteriormente ser integradas no novo chat da organização e ser utilizada por diferentes utilizadores. O objetivo deste projeto é criar um padrão para as diferentes funcionalidades e componentes utilizadas na organização, para que seja possível reduzir a complexidade e intervenção dos desenvolvedores.

O protótipo da framework escolhida é composta por duas grandes aplicações, a aplicação do frontend, que foi desenvolvida em React, e a aplicação responsável pela organização do backend, designada por Firebase.

Durante todo o processo de desenvolvimento do projeto foram utilizados todos os processos de engenharia de software, nos quais se destacam a análise de requisitos e o desenvolvimento de todo o design do respetivo sistema implementado.

Palavras-chave: Firebase, Framework, React, Real-Time Chat

Abstract

This project was developed in a business environment, where the author had the possibility of carrying out a professional internship, which was given the opportunity to develop the project, in the organization called Cool Beans, Lda. This project is part of the development of a chat application, and this is based on the idea of updating and improving the website previously developed to an updated version and with more features than the old chat.

This dissertation aims to document the entire process of implementing a prototype React framework, along with Firebase, as well as new features, so that these can later be integrated into the organization's new chat and be used by different users. The objective of this project is to create a standard for the different functionalities and components used in the organization, so that it is possible to reduce the complexity and intervention of developers.

The chosen framework prototype consists of two large applications, the frontend application, which was developed in React, and the application responsible for organizing the backend, called Firebase.

Throughout the project development process, all software engineering processes were used, in which the requirements analysis and the development of the entire design of the respective implemented system stand out.

Keywords: Firebase, Framework, React, Real-Time Chat

Agradecimentos

Agradeço aos meus familiares, amigos, docentes e ao ISEP todo o apoio e ajuda que me proporcionaram ao longo dos anos.

Agradeço à minha namorada por todo o apoio, força e dedicação que me proporcionou para ser possível a realização da dissertação.

Agradecimento especial ao meu orientador, Ricardo Almeida, pelo tempo disponibilizado e suporte prestado para que esta dissertação tivesse o maior rigor.

Agradecimento especial à Cool Beans, ao meu supervisor Sérgio Carvalho e ao resto da equipa na empresa pelo tempo disponibilizado e conhecimento transmitido.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

API	*Application Programming Interface*
SWOT	*Strengths, Weaknesses, Opportunities and Threats*
UML	*Unified Modelling Language*
XML	*Extensible Markup Language*
HTML	*Hypertext Markup Language*
PWA	*Progressive Web Apps*
MVC	*Model-View-Controller*
MPA	*Multi Page Application*
SPA	*Single Page Application*
HTTPS	*Hyper Text Transfer Protocol Secure*
SSL	*Secure Sockets Layer*
CSS	*Cascading Style Sheets*
NCD	*New Concept Development*
AHP	*Analytic Hierarchy Process*

Lista de Símbolos

β	Largura de banda
π	Pi

1 Introdução

Neste capítulo, é descrita a visão geral do projeto "Aplicação *Web/Mobile* Chat". Nesta primeira fase, é descrito o contexto em que o projeto está inserido. Após a contextualização do projeto, é descrito o problema, demonstrando também os objetivos associados. Neste capítulo, é ainda detalhada a abordagem que foi realizada ao problema e os resultados esperados com o desenvolvimento da aplicação. Por fim, esta secção contêm uma breve descrição de como o presente documento está estruturado.

1.1 Enquadramento/Contexto

O presente projeto foi desenvolvido em ambiente empresarial, na organização denominada de Cool Beans, Lda., onde o autor teve a possibilidade de adquirir novas experiências e táticas de trabalho. A Cool Beans, Lda. é uma organização que desenvolve e explora o comércio de serviços e conteúdos em suporte multimédia e internet, bem como gere e comercializa espaços publicitários em suporte multimédia. Também possui diferentes websites de conteúdo para diferentes utilizadores, bem como um chat, o antigo chat blá, que irá ser o tema de desenvolvimento deste projeto.

O novo chat blá é utilizado neste momento por centenas de utilizadores e é um dos websites que necessita de atenção e atualização, visto que foi criado há mais de 10 anos atrás, as tecnologias e as linguagens de programação foram atualizando e progressivamente melhorando o que fez com que o website começasse a ficar desatualizado. Apesar destes aspetos, os utilizadores diários que o website apresenta, ainda utilizam o mesmo e para melhorar a experiência e evolução do website irá ter de ser feita a atualização e criação do novo chat.

Este projeto surge desta necessidade de atualização e é descrito todo o processo efetuado para o desenvolvimento de um protótipo de uma framework em React e Firebase, capaz de satisfazer as necessidades da organização e dos diferentes utilizadores.

1.2 Problema

A necessidade da realização deste projeto, surgiu devido à falta de interesse dos utilizadores e atualização do website, que fez com que existisse uma perda de utilizadores que iam ao chat diariamente o que poderia levar à total não utilização do chat. A implementação de um novo chat, pode ter um impacto bastante significativo no funcionamento do website e no chamamento de novos utilizadores, tendo em conta também, a fácil correção e atualização do código desenvolvido.

Com isto, a organização sentiu a necessidade de obter um novo chat capaz de trazer novos utilizadores, com as funcionalidades que já existiam e novas, garantindo a total migração de todos os dados dos utilizadores que já existiam no antigo chat para o novo, o que tornará fácil a mudança.

O foco da criação do website é a execução e atualização do novo chat, uma vez que a organização utiliza outras ferramentas e websites, onde algumas das funcionalidades ou linguagens podem ser reutilizadas para a criação de outros websites diferentes. Para além disso, a organização tem como objetivo migrar todos os dados e utilizadores do antigo chat para o novo chat, fazendo com que os utilizadores não se tenham que preocupar com dados perdidos nem conflitos com as suas contas. O código desenvolvido também será de fácil aproveitamento para novas funcionalidades bem como a criação de novos websites dentro da organização.

1.3 Objetivos

Tendo em conta o problema apresentado, o objetivo desta dissertação é a análise de algumas frameworks para a criação de aplicações de chat, uma vez que se pretende desenvolver um novo chat, que será uma aplicação web/mobile. Para além disso, deve ser assegurada a escalabilidade da aplicação, uma vez que, poderão ser adicionadas mais funcionalidades à aplicação e deverá ser relativamente simples fazer alterações. Quando a aplicação estiver numa fase mais completa, será feita a migração dos utilizadores e os respetivos dados do antigo chat para o novo e atualizado chat.

Para isso a abordagem que terá de ser utilizada contará com um estudo das frameworks existentes, identificando e configurando uma simples aplicação inicialmente, em que é necessário ter alguma coisa a funcionar. A partir desse ponto, fazer as devidas alterações e correções, com a criação de uma interface e a configuração do chat, para que seja possível chegar ao resultado final esperado. Esta interface desenvolvida deve ser simples e intuitiva, para que os futuros utilizadores possam familiarizar-se com o sistema de uma maneira mais rápida, bem como, as funcionalidades que existem e poderão vir a ser acrescentadas.

Concluído o protótipo, pretende-se realizar um estudo de satisfação, em que será feito um teste no site em que o utilizador testa e responde a um questionário, no final, acerca da experiência.

1.4 Abordagem

A proposta visa o desenvolvimento de uma aplicação web/mobile de chat capaz de suportar as necessidades dos utilizadores do antigo chat e que permita uma atualização desse chat com novas e atualizadas tecnologias.

Na primeira fase, deve ser realizado um estudo acerca da área de aplicações de chat, com o objetivo de se tornar mais fácil a integração com o problema a solucionar e de forma a adquirir o conhecimento necessário para o desenvolvimento do chat.

Posteriormente, deve ser feita uma sistematização do problema de uma forma mais detalhada de modo a perceber, para a organização, qual o processo que melhor se adequa para a criação da aplicação de chat.

Para além disso, deve ser feita uma análise relativamente às tecnologias e abordagens existentes que melhor se adequam para o problema apresentado.

Numa segunda fase pretende-se desenvolver e desenhar a arquitetura de modo a permitir a modularidade do sistema para que seja possível adicionar novas funcionalidades. Após o desenho, deve ser desenvolvido o protótipo da aplicação de chat, com a arquitetura definida previamente, garantindo assim a modularidade da aplicação.

Seguidamente, deve ser realizado um caso de estudo, acerca da aplicação desenvolvida, que possa validar e confirmar o grau de satisfação dos futuros utilizadores.

Na última fase do processo, os dados recolhidos do estudo devem ser analisados para que seja possível retirar conclusões acerca da aplicação desenvolvida e perceber se é necessário fazer alterações e em que pontos é preciso haver melhorias.

1.5 Estrutura da dissertação

Completados os pontos anteriores que tiveram como objetivo explicar e contextualizar o leitor do objetivo do trabalho realizado, é importante perceber a estrutura da dissertação.

No capítulo 2 é possível encontrar o estado da arte que é um dos aspetos mais importantes da dissertação, visto que é essencial explicar quais as tecnologias mais utilizadas para as aplicações de chat.

O capítulo 3 descreve a análise de valor realizada, onde é identificada a oportunidade que a organização pretende atingir. Seguidamente, foi feita uma breve análise da oportunidade, onde é possível observar a análise SWOT (Strenghts, Weaknesses, Opportunities, Threats), que tem como objetivo captar se a oportunidade apresentada é realmente relevante para ser perseguida. É também explicada a proposta de valor realizada, que demonstram como as necessidades da organização se interligam com os benefícios da aplicação desenvolvida. Por fim, é demonstrado o método de apoio à decisão, o New Concept Development, onde são demonstrados os cálculos feitos para chegar à melhor solução.

No capítulo 4 é possível observar uma descrição detalhada da análise da solução a desenvolver. Neste capítulo é apresentado o modelo do domínio do problema e os requisitos funcionais e não funcionais do problema. Por fim, também é possível ver o desenho da aplicação desenvolvida.

No capítulo 5 encontra-se a descrição da implementação realizada para a resolução do problema, juntamente com imagens e excertos de código da aplicação.

No capítulo 6 é destinado à avaliação da aplicação desenvolvida, onde foi realizado um inquérito de satisfação dos utilizadores. Posteriormente, é feita a análise e interpretação dos resultados obtidos por parte do autor.

Por fim, no capítulo 7, são efetuadas algumas conclusões acerca do projeto implementado. Também são descritas as limitações da solução apresentada, possíveis melhorias e trabalho futuro para o projeto.

2 Contexto e Estado da Arte

Neste capítulo, é possível encontrar uma breve descrição teórica dos conceitos essenciais para a compreensão da área de negócio ao qual está inserido, nomeadamente Aplicações de Chat (mobile/web).
Neste capítulo é explicado também o estado da arte relativo ao tema escolhido para a tese, onde são descritas algumas abordagens existentes para a construção de outras aplicações, bem como, uma comparação entre as outras ferramentas que poderiam ter sido utilizadas para a construção da mesma aplicação.

2.1 Aplicações de Chat

Devido a um dos principais objetivos da presente dissertação ser o desenvolvimento de uma nova aplicação de chat, é necessário entender o conceito de aplicação de chat.

Como é possível perceber, uma aplicação de chat é desenvolvida de maneira a ser criado um sistema instantâneo de mensagens que permite a diferentes utilizadores interagirem entre eles [1]. Conseguir receber mensagens de qualquer parte do mundo é uma das grandes transformações que aconteceram ao longo dos anos, onde é percetível a evolução da tecnologia em relação a este tópico.

Existe uma grande competição relativamente a aplicações de chat, mobile ou web, devido ao facto de os utilizadores procurarem mais funcionalidades diferentes e experiências novas relativamente ao que essas aplicações possam fazer e chamar a atenção dos seus utilizadores. Websites que proporcionem um chat para haver interação entre os diferentes utilizadores do site, obterem um feedback regular é essencial, de modo a ser possível realizar uma melhoria contínua no chat e conseguir assim trazer mais utilizadores e manter os que já utilizam.

2.2 Abordagens para as Aplicações de Chat

O presente subcapítulo visa analisar algumas das abordagens existentes para o desenvolvimento de aplicações de chat. O conhecimento de diferentes abordagens é fundamental para a criação de uma aplicação de chat. Para isso foram analisadas as seguintes abordagens: Aplicações Web, SPA, MPA e PWA.

As subsecções seguintes são destinadas a descrever essas mesmas abordagens, para uma melhor perceção do processo de desenvolvimento de uma aplicação.

2.2.1 Aplicações Web

Aplicações Web referem-se a aplicações acedidas através de um browser utilizando uma certa network e desenvolvidas utilizando linguagens suportadas pelo browser, como HTML e JavaScript. Estas páginas podem ter alguns pontos positivos como ser mais barato o desenvolvimento de aplicações, terem diferentes linguagens antigas e conhecidas pelos desenvolvedores e a atualização destas aplicações é muito mais simplificada [2]. Existem diferentes tipos de aplicações web que maior parte das pessoas conhece atualmente e que provavelmente não sabia, mas o Twitter, o Facebook, o Youtube, são todas aplicações web desenvolvidas de uma forma mais responsiva para que seja possível adaptarem-se a diferentes tipos de telas, como telemóveis, tablets e computadores.

2.2.2 SPA

A single page application é uma abordagem para o desenvolvimento de aplicações web que é usada para desenvolver aplicações mais simples com menos conteúdo [3].

Neste tipo de aplicações as views não são páginas apenas constituídas por HTML. São partes do DOM que fazem com que seja visível as áreas do ecrã. Depois da primeira página inicial ser carregada, todas as ferramentas necessárias para disponibilizar e criar as views são instaladas e prontas para serem utilizadas. Se for preciso ter uma nova view, é gerada localmente no browser e dinamicamente ligada ao DOM através de JavaScript, não sendo necessário atualizar a página.

Com isto, o SPA é uma aplicação web completa com apenas uma página para todas as partes do UI.

2.2.3 MPA

A multi page application é outra abordagem que é mais utilizada para sistemas maiores, com diferentes tipos de serviços e que têm mais tipos de interações com o utilizador [3].

Qualquer mudança no browser envolve receber novas páginas do servidor, os caminhos são assim registados no servidor e ficam guardados na cache. O que isto significa é que cada pedido enviado do servidor irá receber a página a mostrar os resultados que foram pedidos ou então um erro. A lógica da aplicação está no servidor sendo que o cliente é o destino da página pedida.

2.2.4 PWA

Uma aplicação web progressiva é um website criado com tecnologias que maior parte dos desenvolvedores web conhece, HTML, CSS e JavaScript, mas este tipo de website consegue oferecer aos utilizadores uma experiência melhorada [4].

Estas PWA possuem um ficheiro chamado manifest que contêm informação acerca do website, incluindo os símbolos, o fundo, as cores e a orientação. Também contam com uma funcionalidade bastante importante que são os Service Workers que permitem colocar em cache partes do website para proporcionar uma experiência offline.

A diferença que existe destas PWA para um website normal é a possibilidade de adicionar funcionalidades e a experiência do utilizador [5]. Quando um website normal precisa de um utilizador para abrir um browser, escrever o URL e esperar que todo o conteúdo carregue e seja instalado em cada visita, o que é eficiente para caso o website estiver offline, a PWA apenas precisa que o utilizador carregue uma vez a página para que todos os ficheiros estáticos bem como o HTML, CSS, JavaScript, imagens e fontes do website fiquem agora guardados no sistema do utilizador, preparados para serem utilizados offline.

2.3 Frameworks e Tecnologias

Como um dos principais objetivos deste projeto é a criação de uma aplicação de chat com uma simples interface, é necessário que se realize um estudo das principais ferramentas disponíveis para o desenvolvimento do frontend. Uma das principais linguagens de programação utilizadas para o desenvolvimento de aplicações web, JavaScript, é uma das mais populares da última década [6]. Tendo esta informação em conta, realizou-se um estudo entre três das principais Frameworks em JavaScript mais utilizadas, sendo elas ReactJs, AngularJs e VueJs.

Visto que é necessário um serviço para o backend, o Firebase também será explicado neste capítulo pois foi a opção escolhida para o projeto.

2.3.1 ReactJs

Tendo em conta que a empresa, Cool Beans, precisava de uma solução para a atualização do antigo chat, e depois de realizado o estudo aos requisitos recebidos para a realização deste

projeto, surgiu a oportunidade da pesquisa de tecnologias para o desenvolvimento do frontend para o novo chat. A conclusão dos estudos realizados sobre o ReactJs será explicada a seguir.

ReactJs[1] é uma biblioteca JavaScript para desenvolver componentes de User Interface (UI), que ultimamente tem ganho cada vez mais influência, tornando-se numa das ferramentas mais utilizadas para a criação de frontend [7]. Esta tecnologia é utilizada para o desenvolvimento de aplicações, em muitos casos de grande dimensão e complexas, baseadas em Web, em que é possível tornar o processo de criação de UIs mais simples e mais rápida, devido ao facto de as componentes só serem atualizados quando existe alguma alteração. O servidor que é responsável pela parte de renderização do React é o NodeJs[2]. O ambiente de desenvolvimento das componentes de React utilizam geralmente a arquitetura MVC.

O principal bloco de construção de uma aplicação feita em React é as componentes. O método de render() que se encontra em todas as componentes de React delineadas gera o virtual DOM[3], que está explicado na subsecção 2.3.1.1 que depois também é convertido no DOM do navegador que for utilizado. Todas as interfaces são projetadas através da construção em árvore, que é responsável por mapear os vários componentes que estão presentes e desenhados.

O React utiliza uma linguagem designada por JavaScript XML (JSX) para a construção da árvore de componentes. O JSX simplifica a definição dos processos de coordenação e na definição de propriedades, como no caso dos atributos em XML.

Esta tecnologia foi desenhada com o objetivo de suportar um fluxo de dados unidirecional devido a estas componentes terem de ser imutáveis e os dados não puderem ser alterados em nenhuma ocasião. Com isto, os dados apenas podem surgir num sentido e nunca no sentido contrário para que seja possível assegurar a sincronização entre os componentes pais e filhos correspondentes. Por este motivo é que esta tecnologia é uma das mais utilizadas para o desenvolvimento de aplicações Web interativas. Se alguma alteração for feita na ligação de dados upstream, as componentes que têm acesso a esses dados são automaticamente renderizados, guardando assim as alterações registadas.

De seguida, as subsecções têm como objetivo demonstrar e detalhar os aspetos mais importantes do React, e que tornam a tecnologia cada vez mais utilizada no processo de desenvolvimento web [6].

2.3.1.1 Virtual DOM

Para que a aplicação desenvolvida tenha uma melhor capacidade de desempenho com o intuito de ser mais eficiente, uma das caraterísticas do React para realizar esta vantagem é de fornecer um Document Object Model mais leve e eficaz, de maneira que o React interage com o DOM guardado na memória e não com o DOM do navegador [7][8].

[1] https://reactjs.org/
[2] https://nodejs.org/en/
[3] https://developer.chrome.com/docs/devtools/dom/

Um dos problemas das aplicações que não são implementadas com React é o de fazerem a interação direta com o DOM do navegador que estão a utilizar, o que irá afetar a árvore do DOM, quando ocorre algum evento. Por outras palavras, em casos que existem muitos dados para serem modificados nesses eventos, o desempenho da aplicação será afetado. Uma das soluções que o React possui para esse problema é utilização de uma virtual DOM.

Para se perceber melhor o conceito de virtual DOM, a ideia é semelhante ao processo do DOM gerado pelo navegador, mas só que em vez de manipular diretamente a árvore do DOM, a aplicação cria uma versão mais abstrata da árvore e guarda na memória. Esta versão irá fazer uma comparação entre as duas, o virtual DOM e o DOM do navegador, onde as diferenças encontradas serão alteradas apenas no DOM do navegador, eliminando assim a possibilidade de causar problemas no desempenho. Esta solução torna a manipulação do DOM significamente mais rápida quando é agrupado com algoritmos de comparação eficientes e as operações realizadas nas árvores [9].

2.3.1.2 JavaScript XML

A utilização da linguagem JSX torna a utilização do React muito mais simples, que tem caraterísticas muito semelhantes ao XML [7][8]. A utilização desta linguagem não é obrigatória quando se está no processo de desenvolver uma aplicação em React. No entanto, esta linguagem pode ser bastante utilizada entre a comunidade de desenvolvedores devido a facilitar a ligação entre diferentes ações e de facilitar a escrita de marcações dentro das componentes no React.

Posto isto, ao longo do tempo, a utilização desta linguagem tornasse cada vez mais popular, devido a ser de mais fácil adaptação e menos complexa do que muitas Frameworks no mercado.

2.3.2 VueJs

O Vue é um tipo de linguagem em JavaScript que também utiliza métodos para executar ações entre os utilizadores e as aplicações. Estes métodos são acedidos através de uma referência â instância do Vue. Para que o Vue consiga utilizar os métodos adequados, utiliza a expressão function() para esse feito.

Na criação de uma aplicação em VueJs[4] é necessário ter alguns aspetos importantes em conta, um desses aspetos é o template. O template é a caraterística mais importante do Vue e é constituído apenas por atributos especiais e diretivas, num simples ficheiro de HTML. A razão para a existência deste template é que serve para gerar o DOM que substitui o elemento escolhido com a opção especial na página web. Esta opção especial é a propriedade que indica o local específico da instância do Vue na página web. Para a manipulação dos dados, o Vue utiliza uma propriedade chamada filter que contribui para reduzir a duplicação de código e simplificar o mesmo.

[4] https://vuejs.org/

Para guardar informação nova na base de dados, os desenvolvedores precisam de criar instâncias novas porque as propriedades dos dados são todos adicionados ao mesmo tempo num sistema que controla todas as alterações feitas.

Uma das propriedades principais do Vue, que já foi referida no React, mas que também existe no Vue, é a componente. Esta componente contribui para a manutenção e a reutilização do código de uma maneira mais eficiente. Todas as componentes juntas numa aplicação implementada em Vue, trabalham em conjunto para que consigam alcançar uma boa performance no UI e na informação recebida [10].

2.3.3 AngularJs

O AngularJs[5] é uma Open Source framework em JavaScript, que tem como objetivo ajudar os desenvolvedores a criar aplicações em single page. O objetivo da framework é conseguir tornar as aplicações web mais modulares e consequentemente mais fáceis de realizar a sua manutenção. Tem como base a arquitetura padrão Model-View-Controller, o que torna o processo mais fácil para desenvolver, manter e testar [11].

O Angular ajuda a criar aplicações web baseadas em HTML, CSS e JavaScript e na criação de elementos personalizados para os Document Object Model. O principal ficheiro é o index.html que funciona como uma base para a aplicação inteira ou website. Se a aplicação necessitar de ficheiros com scripts ou CSS, a única coisa que é preciso fazer é incluir nesse ficheiro index.html o que for necessário e não é preciso repetir em mais lado nenhum da aplicação, pois ele consegue ir buscar toda a informação de uma maneira eficaz.

Um dos aspetos exclusivos do Angular é de ser totalmente baseado numa Bidirectional-Data-Binding. Esta sincronização é uma maneira automática de atualizar a view quando o modelo sofre alterações e atualizar o modelo sempre que a view é alterada também. Com isto, a framework é automática e imediata no processo.

2.3.4 Comparação das frameworks

Para uma melhor perceção do que cada framework consegue fazer, foi feita uma tabela comparativa entre as três frameworks explicadas anteriormente com algumas perguntas importantes segundo o estudo feito por Marin Kaluža, Krešimir Troskot e Bernard Vukelić [12]. A tabela é dividida em duas classificações, sendo que na primeira coluna aparece de uma forma mais textual (Sim, Não e Parcialmente) a resposta à análise realizada e na segunda coluna aparece de uma forma mais numérica (0, 1, 1.5, 2).

Tabela 1 – Comparação das frameworks ReactJs, VueJs e AngularJs

Descrição	ReactJs	VueJs	AngularJs

[5] https://angularjs.org/

MPA	---------		--------		----------	
Apenas a importação de JS é necessária para começar?	Sim	1,5	Sim	2	Não	0
A framework inclui muita sobrecarga?	No	2	Não	2	Sim	0
O workflow é necessário?	Parcialmente	1	Não	2	Sim	0
Servidor – renderização lateral	Sim	1	Sim	2	Sim	2
SPA	---------		----------		---------	
A capacidade de escrever e manter uma grande quantidade de código complexo	Parcialmente	1	Parcialmente	1	Sim	2
O desenvolvedor deve confiar em pacotes de terceiros?	Sim	0	Parcialmente	1	Não	2
Qual é a possibilidade de comprimir a aplicação, minimizando o tamanho do arquivo?	Sim	2	Sim	2	Sim	2
Manutenção do estado dos dados	Sim	1,5	Sim	2	Sim	1

Após uma breve análise aos resultados é possível perceber que para a primeira parte da tabela (MPA) existe uma diferença entre o Angular e o Vue. A somatória dos valores às primeiras 3 perguntas na tabela do Angular é 0 e no Vue é 6. Isto indica que a framework Vue é muito mais simples que o Angular e pode assim ser utilizada como uma libraria. Em algumas perguntas, o React demonstra que consegue parcialmente concretizar o critério definido. Outra observação é que o Angular e o Vue permitem uma renderização mais do lado do servidor, através de pacotes oficiais, enquanto o React permite uma renderização do lado do servidor através de pacotes de terceiros.

Na segunda parte da tabela (SPA) o Angular é a framework que, comparado com as outras duas frameworks analisadas, é a melhor para escrever e manter muito código complexo. O Vue e o React não têm práticas para nomear os arquivos e as pastas claramente definidas, com isto, permitem apenas parcialmente escrever e manter muito código complexo. O Angular, comparado com o React, não necessita de recorrer a muitos pacotes de terceiros, enquanto o Vue precisa parcialmente. Todas as frameworks permitem uma fácil otimização, enquanto a única framework que permite verificar e gerir o estado é o Vue.

Com esta análise, foi possível perceber melhor o que cada framework é capaz de proporcionar ao desenvolvedor de modo a ajudar, da maneira mais eficaz e eficiente, para que seja possível o desenvolvimento das melhores aplicações.

Posto isto, a framework que vai ser utilizada para o frontend é o React, devido a ser uma das frameworks que a empresa utiliza atualmente, por ser a framework que o autor tem mais experiência e pelo estudo realizado neste tópico que demonstra as caraterísticas positivas e necessárias para a aplicação. Na secção 3.4 é possível observar um método de apoio à decisão

que permite identificar qual a melhor alternativa de framework a utilizar (Angular, React ou Vue), tendo em conta os critérios de performance, modularidade e usabilidade.

2.3.5 Firebase

O Firebase[6] é um serviço de backend, que começou inicialmente por ser uma base de dados de tempo real, mas que se transformou numa plataforma de cloud da Google [13]. Esta plataforma é uma solução para aplicações mobile e web que inclui serviços para construir, testar e gerir as aplicações. O Firebase não deixa de ser uma base de dados em NoSQL, mas guarda os dados numa cloud.

Quando aplicações são desenvolvidas, o desenvolvedor tem que ter a noção que precisa de contar com um serviço de backend que garante o bom funcionamento das aplicações. Para isso existe o Firebase, que é uma excelente opção devido a poupar tempo e permitir ao desenvolvedor focar-se no desenvolvimento da aplicação. A data é guardada num ficheiro JSON, na qual não é necessário escrever nenhum código para o servidor pois já é uma API que está preparada para o desenvolvimento e implementação de aplicações.

O Firebase permite a instalação de pacotes de código predefinidos para que seja mais facilmente automatizar as tarefas comuns de desenvolvimento. Também é uma tecnologia que pode ser facilmente integrada com diferentes ferramentas, como por exemplo a Play Store, o Slack ou o Jira.

Algumas das funcionalidades do Firebase vão ser descritas e explicadas a seguir:

- Base de dados em tempo real: sincroniza a data recebida do remetente para o Firebase, que vai atualizar instantaneamente, e o cliente vai receber a data atualizada em tempo real em qualquer dispositivo;

- Armazenamento de ficheiros: permite guardar ficheiros diretamente do cliente mesmo estando em modo offline. Toda a informação guardada em cloud está protegida pelo sistema de segurança do Firebase;

- Autenticação: disponibiliza o próprio UI customizado de autenticação ou para fazer sign in com diferentes métodos como email e password, Google, Facebook e muito mais;

- Serviço de Hosting: consegue fazer a implementação de páginas estáticas e SPA que são desenvolvidas com HTML, CSS e JavaScript. Para uma maior segurança, o Firebase utiliza os protocolos de HTTPS e SSL;

- Serviço Gratuito: o Firebase não precisa de ser pago para ser possível usufruir dos serviços de base de dados. Apenas é necessário pagar se a memória da base de dados não for suficiente.

[6] https://firebase.google.com/

Tendo estes motivos em consideração, o Firebase vai ser a tecnologia utilizada para o projeto devido a coincidir com as caraterísticas necessárias para o desenvolvimento da aplicação.

3 Análise de Valor

Neste capítulo, é apresentado a análise de valor do projeto realizado, com o objetivo de constatar o valor produzido para a organização, devido a ser um projeto desenvolvido a nível interno.

Para realizar a análise, foi utilizado o modelo New Concept Development [14] com o objetivo de conseguir identificar a oportunidade e fazer a devida análise. Para uma melhor definição da parte inicial do processo de negócio e inovação, é demonstrado na figura 1, o modelo representado, no qual é possível perceber que o NCD está dividido em três partes:

- Fatores de influência: que demonstram os diversos fatores que caraterizam a organização, os fatores externos (como por exemplo clientes, concorrência, etc.) e a estratégia empresarial;
- Elementos: estes elementos podem ser controláveis pela organização, existindo assim cinco elementos chave. Estes elementos são a identificação da oportunidade, análise da oportunidade, inovação e adição de ideias, seleção de ideias e definição do conceito abordado. Para o caso do projeto, apenas foram utilizados a identificação da oportunidade e a análise da oportunidade;
- Motor: é a liderança, cultura e estratégia de negócio da organização que indica os cinco elementos chave que são controláveis pela organização.

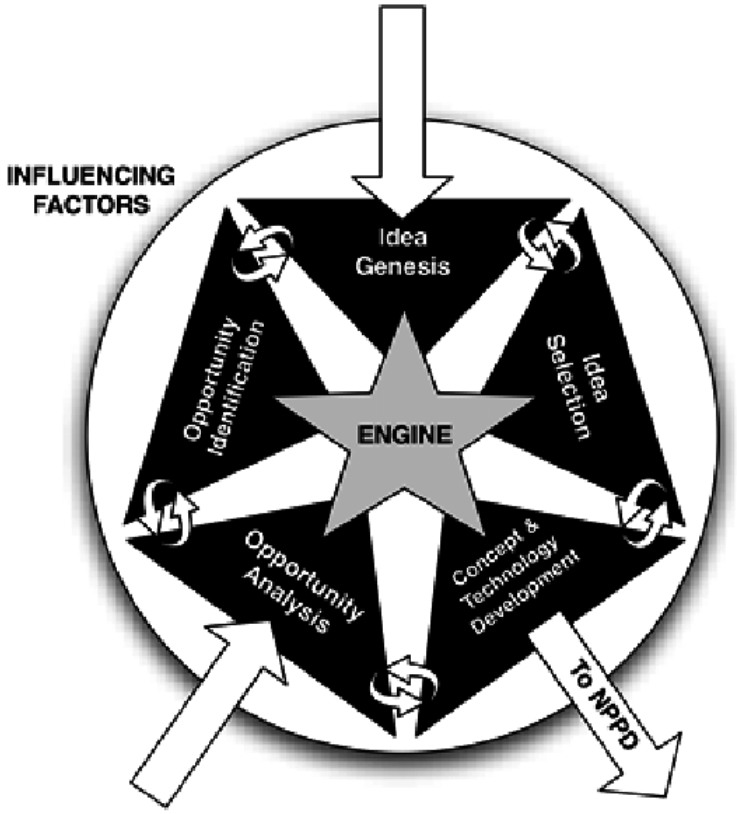


Figura 1 – Modelo New Concept Development[7]

Foi definida uma proposta de valor para o presente projeto, juntamente com a identificação da oportunidade e da análise da oportunidade. Para além disso foi utilizado um método de apoio à decisão, o Analytic Hierarchy Process, que permitiu identificar do estudo realizado no capítulo 2, com base em alguns critérios definidos, a melhor tecnologia para a resolução do problema descrito no projeto.

3.1 Identificação da oportunidade

Para conseguir identificar as oportunidades que a empresa pretende atingir, foi realizado este subcapítulo para uma melhor explicação [14].

A Cool Beans já possuía uma aplicação de chat[8] que já tinha sido criada em 2000, onde a aplicação desenvolvida ainda tem centenas de pessoas que a utilizam diariamente. Porém, devido ao avanço tecnológico que decorreu durante estes anos, verificou-se que as tecnologias utilizadas e algumas funcionalidades do chat começaram a ficar desatualizadas e sem funcionar.

[7] Retirado de: researchgate.net/figure/The-New-Concept-Development-NCD-Model-according-to-Koen-et-al-1-Image-reproduced-by_fig3_281199416
[8] http://bla.aeiou.pt/

Com isto, a organização sentiu a necessidade de criar uma aplicação de chat que fosse capaz de ser utilizada em mobile e em web. O principal objetivo é ter uma aplicação de chat que funcione corretamente e que substitua o desatualizado chat, mas também deve ter algumas funcionalidades do antigo chat bem como ser possível acrescentar novas funcionalidades para atração de novos utilizadores.

Para além disto, devido a existirem constantemente pessoas no chat, é necessário migrar os dados do chat antigo para a nova solução de modo a quem já utilizava a aplicação, apenas tenha que entrar com os seus dados no novo chat. Outro ponto importante é que a solução seja capaz de receber novas funcionalidades de uma maneira simples, para que seja possível acrescentar novas atividades e entretenimentos para os diferentes utilizadores e de maneira que o chat esteja em baixo o mínimo tempo possível.

3.2 Análise da oportunidade

Com a oportunidade encontrada, é necessário avaliá-lo, com o objetivo de definitivamente verificar se é mesmo válido perseguir esta oportunidade. Existem 4 tipos de estratégias diferentes para avaliar sendo a Strategic Framing, o Market Segment Assessment, o Competitor Analysis e o Customer Analysis [15].

Para este projeto, a estratégia utilizada foi a Strategic Framing visto que determina como é que a oportunidade definida encaixa no mercado da organização e nos pontos fortes, fracos e ameaças da tecnologia. Com isto, foi elaborado uma análise SWOT [16], demonstrada na Figura 2, que está dividida em fatores internos (Strengths e Weaknesses) e em fatores externos (Opportunities e Threats).

Figura 2 – Análise SWOT

Como forças é importante realçar que a criação de uma aplicação de chat pode trazer novas e melhoradas aplicações ao chat, de modo que o código desenvolvido seja escalável. Devido ao desenvolvedor, o autor da dissertação, ser quem desenvolveu o código e será quem irá apoiar em problemas técnicos que surjam, o tempo de desenvolvimento de novas tecnologias será mais reduzido de modo a tornar mais rápido qualquer atualização do chat. Como referido anteriormente, a eficácia na correção de qualquer erro será muito maior devido a ter sido o próprio autor a desenvolver o código desenvolvido.

Relativamente às fraquezas, é importante explicar que devido à evolução constante das tecnologias nos dias de hoje, podem surgir problemas na implementação da solução, ao nível do código ou até ao colocar a aplicação disponível para toda a gente. Outra fraqueza importante identificada é a impossibilidade/demora na correção de erros por parte de outros desenvolvedores na organização, devido a ter sido apenas o autor a desenvolver a solução, caso outra pessoa tentar arranjar ou acrescentar alguma funcionalidade, pode se tornar uma tarefa complicada por não conhecer a tecnologia usada ou até o código desenvolvido.

Como o projeto desenvolvido corresponde a um projeto interno de uma organização, foram tomados em conta fatores externos como comportamentos organizacionais. Com o desenvolvimento da solução é sempre importante trazer atualizações e novas funcionalidades para que o chat consiga trazer o máximo de utilizadores possível. Também é necessário ter em consideração que, devido à pandemia, a criação de uma aplicação de chat online é uma oportunidade devido às pessoas recorrerem a outro tipo de entretenimento “mais seguro”, sem ter que sair de casa, aumentando assim o número de utilizadores da aplicação.

Por outro lado, existem algumas ameaças relativamente ao projeto, como o surgimento de aplicações de outras organizações que façam competição com esta solução e a descontinuidade das tecnologias ou sistemas utilizados no desenvolvimento do projeto. Para além disso, a utilização do novo chat e a eliminação do antigo chat pode levar a que os utilizadores não gostem da aplicação ou até que não consigam ter uma fácil adaptação, não utilizando assim o novo chat.

Tendo em conta a análise efetuada, é possível afirmar que esta aplicação traz muito valor para a organização, de uma maneira monetária e produtiva. Para além disso, o desenvolvimento de novas funcionalidades para a aplicação, devido ao código ser escalável, será de mais fácil atualização relativamente a esse aspeto.

3.3 Proposta de Valor

Tendo em conta que se está a desenvolver uma aplicação para uso da organização e distribuição para o resto dos utilizadores interessados, pretende-se demonstrar a interligação existente entre os consumidores/utilizadores e os benefícios que o projeto vai ter com a sua utilização. Foi então definida a seguinte proposta de valor:

“O projeto desenvolvido é uma aplicação de chat mobile/web que será responsável pela atualização do antigo chat blá, onde algumas das funcionalidades serão passadas para o novo chat. Para além disso, tem como um dos objetivos, ser possível a fácil integração de novas funcionalidades sempre que for necessário ou decidido, assegurando a sua escalabilidade. Desta forma, é possível reduzir o tempo de desenvolvimento de novas funcionalidades e de haver uma maior eficácia na correção de erros que possam vir a existir, diminuído o tempo de implementação e tornando assim o código desenvolvido escalável. ”

Para ser possível perceber de que maneira é que o projeto desenvolvido pode trazer valor para todas as partes interessadas foi elaborado um modelo CANVAS [17], descrito na Figura 3, para a proposta de valor.

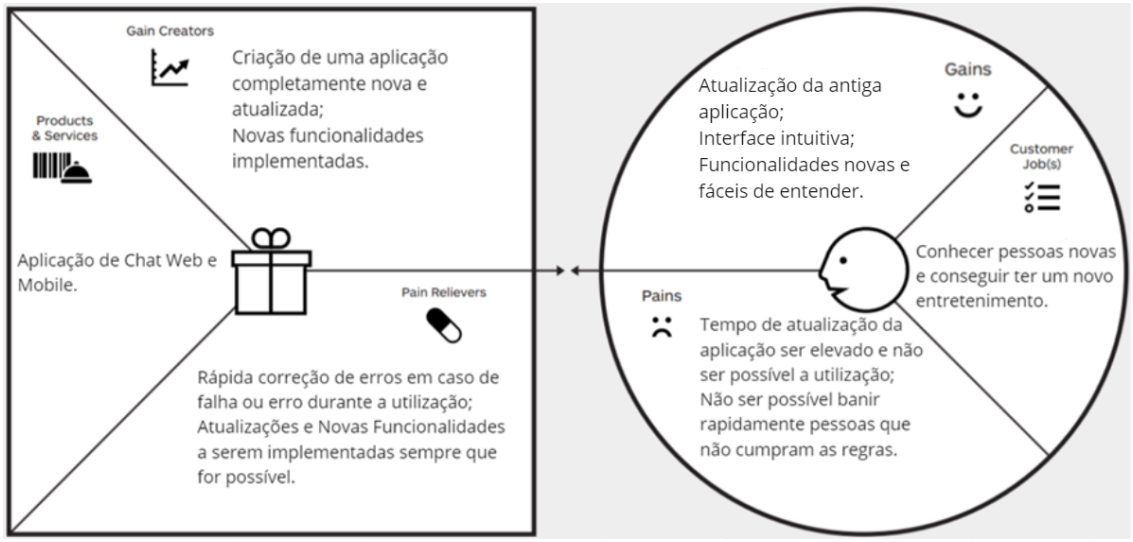


Figura 3 - CANVAS desenvolvido para a Proposta de valor

O modelo é dividido em 6 partes, nomeadamente:

- **Products & Services:** Esta parte mostra que produtos e serviços é que são oferecidos ao cliente;
- **Gain Creators:** Identifica de que maneira o produto desenvolvido contribui para a felicidade do cliente;
- **Pain Relievers:** Identifica de que maneira o produto oferecido alivia os aspetos que o incomodam;
- **Gains:** Esta parte define que fatores é que facilitam o trabalho do cliente na utilização da aplicação;
- **Pains:** Esta secção demonstra que fatores incomodam o cliente;
- **Customer Job:** Identifica que fatores é que permite ao cliente obter a partir da utilização do produto;

3.4 Analytic Hierarchy Process

Para estruturar a decisão do problema é necessário realizar a tarefa mais criativa para tomar decisões que é a escolha dos fatores que são mais importantes para essa decisão. A Analytic Hierarchy Process organiza esses fatores, depois de escolhidos, numa estrutura hierárquica. Este método matemático é utilizado para apoiar na tomada de decisões e na análise das mesmas [18]. Quando se desenvolve este tipo de estruturas hierárquicas, é importante ter estes fatores em mente:

- Representar o problema o mais completo possível, mas não tão completo a ponto de perder a sensibilidade à mudança nos elementos;
- Considerar o ambiente ao redor do problema;
- Identificar os problemas ou atributos que contribuem para a solução;
- Identificar os participantes associados ao problema.

O principal foco desta dissertação foi a criação de uma aplicação de chat, com isto, foi aplicado o método AHP às tecnologias utilizadas com o intuito de perceber qual a melhor framework a adotar para a construção da solução. Visto que foi efetuado um estudo anteriormente, para perceber melhor as tecnologias que podiam ser utilizadas (Angular, React e Vue), o método AHP foi utilizado para ser definido um grau de importância e para auxiliar na escolha da tecnologia a ser utilizada.

Para a análise, foram escolhidos os três critérios mais relevantes para ser feito um estudo com a maior precisão possível, sendo estes:

- A tecnologia suporta uma boa performance;
  (DOM, alocação de memória, tamanho, tempo de arranque)
- A tecnologia suporta uma boa modularidade;
  (Packages, flexibilidade, reutilização do código)
- A tecnologia suporta uma boa usabilidade.
  (Documentação, curva de aprendizagem)

Para ser possível definir o nível de importância de cada critério definido, foi utilizada a escala demonstrada na Figura 4.

Nivel de importância	Definição	Explicação
1	Igual importânica	As duas atividades contribuem igualmente para o objetivo
3	Fraca importância	A experiência e o julgamento favorecem levemente uma atividade em relação à outra
5	Forte importância	A experiência e o julgamento favorecem fortemente uma atividade em relação à outra
7	Muito forte importância	Uma atividade é muito fortemente favorecida em relação a outra
9	Importância absoluta	A evidência favorece uma atividade em relação a outra com o mais alto grau de certeza
2,4,6,8	Valores intermediários	Quando se procura uma condição de compromisso entre duas definições

Figura 4 – Níveis de importância para comparações

Para calcular a razão de consistência (RC) é utilizado um índice aleatório (IR), que tem em conta o número de critérios definidos. Estes valores de IR, para matrizes quadradas de ordem n, estão descritos na Figura 5.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

Figura 5 – Valores de IR para matrizes quadradas de ordem n

Para conseguir efetuar o estudo, foi necessário começar por elaborar a árvore hierárquica, descrita na Figura 6, onde são identificados o problema, os critérios definidos e as alternativas que serão utilizadas para os critérios.

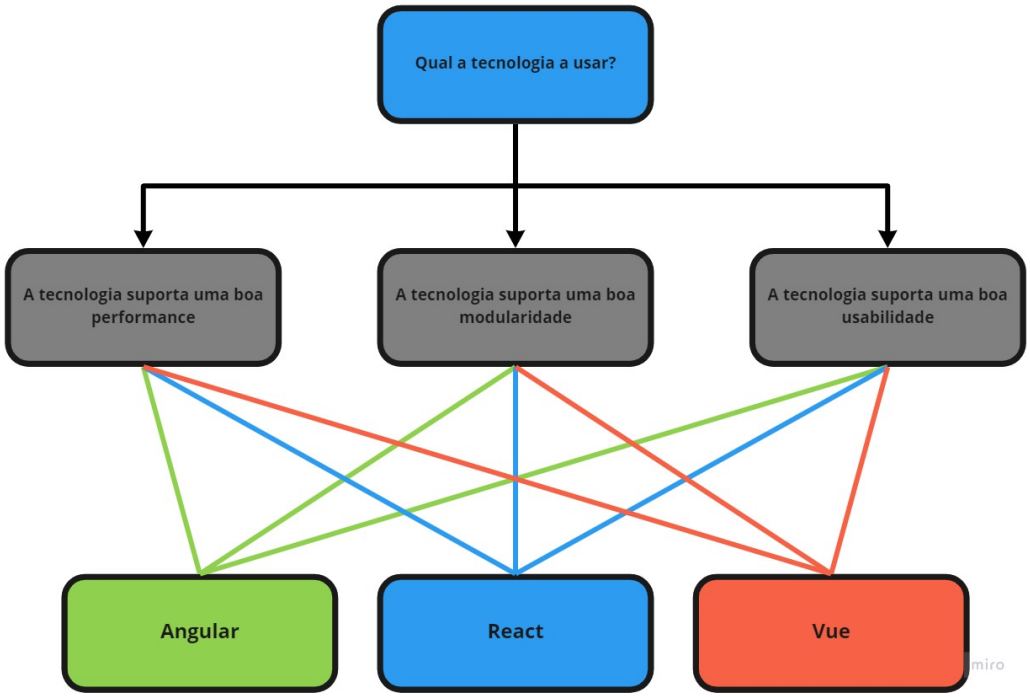


Figura 6 – Árvore hierárquica definida

Seguidamente foi elaborada a matriz par a par, demonstrada na Tabela 2, que é responsável por comparar os critérios definidos e atribuir um valor de intensidade de importância, tendo em conta a escala que foi demonstrada anteriormente.

Tabela 2 – Matriz de comparação par a par para os critérios definidos

Critérios	Performance	Modularidade	Usabilidade
Performance	1	1/3	5
Modularidade	3	1	6
Usabilidade	1/5	1/6	1
Total	4 1/5	1 1/2	12

Tendo a matriz de comparação par a par feita, foi calculada a matriz normalizada, que é resultado da divisão de cada elemento pelo somatório da coluna desse elemento. Na Tabela 3, é possível verificar os valores da matriz normalizada para os critérios definidos e o vetor de prioridades.

Tabela 3 – Matriz normalizada para os critérios definidos e respetivo valor de prioridades

Critérios	Performance	Modularidade	Usabilidade	Vetor de prioridades
Performance	0,2381	0,2222	0,4167	0,2923
Modularidade	0,7143	0,6667	0,5000	0,6270
Usabilidade	0,0476	0,1111	0,0833	0,0807

Subsequentemente, foi elaborado um teste de consistência para os critérios definidos anteriormente.

- **Step 1** – Calcular o vetor próprio

Teste da consistência

	Matriz de comparação par a par				Pesos		
STEP1	1	1/3	5	x	0,2923	=	1
	3	1	6	x	0,627	=	2
	1/5	1/6	1	x	0,0807	=	1/4

Figura 7 - Cálculo do vetor próprio

Calculou-se através da multiplicação da matriz de comparação par a par com o vetor de pesos que foi calculado na parte da normalização.

- **Step 2** – Calcular o vetor intermédio

Os valores encontrados no step anterior são assim divididos pelo respetivo valor do peso normalizado.

$$\frac{1}{0{,}2923} = 3{,}0954$$

$$\frac{2}{0{,}6270} = 3{,}1708$$

$$\frac{0{,}25}{0{,}0807} = 3{,}0193$$

- **Step 3** – Calcular o valor próprio

Com os valores obtidos anteriormente, foi necessário fazer a soma dos três vetores intermédios e dividir por três devido a serem 3 critérios.

$$\frac{3{,}0954 + 3{,}1708 + 3{,}0193}{3} = 3{,}0952$$

- **Step 4** – Calcular o índice de consistência

Para o cálculo do índice de consistência existe uma fórmula:

$$IC = \frac{(\lambda \max - n)}{(n-1)}$$

ou seja, alterando pelos valores já obtidos,

$$IC = \frac{(3{,}0952 - 3)}{(3-1)} = 0{,}047599$$

- **Step 5** – Calcular a razão de consistência

Para o cálculo da razão de consistência existe uma fórmula:

$$RC = \frac{IC}{IR}$$

Como estão definidos 3 critérios, se verificarmos o valor do IR na Figura 5 equivale a 0,58.

$$RC = \frac{0{,}047599}{0{,}58} = 0{,}082067$$

O valor obtido de RC é menor ou igual a 0,1. Com isto pode-se concluir que os valores de prioridades utilizados foram consistentes.

Depois de verificar que os valores utilizados para a comparação entre critérios estão consistentes, foram elaboradas as matrizes de comparação para cada critério (Performance, Modularidade e Usabilidade) considerando cada uma das alternativas que está a ser feita a análise (Angular, React e Vue). A Tabela 4 demonstra os valores da comparação paritária para a parte da performance.

Tabela 4 – Comparação paritária para performance

Performance	Angular	React	Vue
Angular	1	1/2	4
React	3	1	5
Vue	1/5	1/6	1
Total	4,2	1,6667	10

A Tabela 5 demonstra os valores da comparação paritária para a parte da modularidade.

Tabela 5 – Comparação paritária para modularidade

Modularidade	Angular	React	Vue
Angular	1	1/3	3
React	4	1	6
Vue	1/4	1/7	1
Total	5,25	1,4762	10

A Tabela 6 demonstra os valores da comparação paritária para a parte da usabilidade.

Tabela 6 – Comparação paritária para usabilidade

Usabilidade	Angular	React	Vue
Angular	1	1/3	3
React	4	1	5
Vue	1/4	1/6	1
Total	5,25	1,5000	9

Posteriormente foi possível calcular as matrizes de comparação paritárias para cada uma das opções.

A Tabela 7 demonstra os valores da comparação paritária normalizada para a parte da performance.

Tabela 7 – Matriz paritária para performance

Performance	Angular	React	Vue	Pesos
Angular	1/4	2/7	2/5	0,3127
React	5/7	0,6000	0,5000	0,6048
Vue	0,04762	0,1000	0,1000	0,0825
Total	-	-	-	1

A Tabela 8 demonstra os valores da comparação paritária normalizada para a parte da modularidade.

Tabela 8 – Matriz paritária para modularidade

Modularidade	Angular	React	Vue	Pesos
Angular	1/5	2/9	2/7	0,2388
React	0,7619	0,6774	0,6000	0,6798
Vue	0,04762	0,09677	0,1000	0,0815
Total	-	-	-	1

A Tabela 9 demonstra os valores da comparação paritária normalizada para a parte da usabilidade.

Tabela 9 – Matriz paritária para usabilidade

Usabilidade	Angular	React	Vue	Pesos
Angular	1/5	2/9	1/3	0,2487
React	3/4	2/3	5/9	0,6614
Vue	0,0476	1/9	0,1111	0,0899
Total	-	-	-	1

Para verificar a consistência dos dados foram calculadas as razões de consistência para cada um dos critérios, utilizando o mesmo método de raciocínio descrito anteriormente nos steps. A Tabela 10 demonstra a razão de consistência obtida para cada um dos critérios.

Tabela 10 – Valores das razões de consistência para cada critério definido

Critérios	Razão de consistência

Performance	0,067644
Modularidade	0,006129
Usabilidade	0,020803

Realizando uma análise da Tabela 10, verificou-se que as razões de consistência encontradas para cada critério são todas menores ou iguais que 0.1, podendo-se assim afirmar que os valores utilizados são consistentes.

Com isto, foi construída a matriz de prioridades globais para cada um dos critérios definidos, que está demonstrada na Tabela 11.

Tabela 11 – Matriz com prioridades globais

Prioridades Globais	Performance	Modularidade	Usabilidade
Angular	0,3127	0,2388	0,2487
React	0,6048	0,6798	0,6614
Vue	0,0825	0,0815	0,0899

Por último, foi calculada a prioridade composta para cada uma das alternativas, através da multiplicação da matriz com as prioridades globais obtidas anteriormente e os pesos calculados na normalização da comparação entre os critérios definidos, obtendo-se os valores apresentados na Tabela 12.

Tabela 12 – Valor da prioridade composta para cada alternativa

Angular	0,2612
React	**0,656393**
Vue	0,08247

Posto isto, com os dados apresentados Tabela 12, verifica-se que a alternativa mais indicada é o React, com aproximadamente 0.66, tendo em conta os critérios definidos e as respetivas importâncias.

4 Análise e desenho da solução

Neste capítulo será apresentado a análise e desenho do problema descrito na seção 1.2. Inicialmente, realiza-se uma breve análise ao domínio do problema proposto. De seguida são especificados os requisitos que resultam do problema, sendo estes funcionais e não funcionais. Para ser possível perceber como o sistema será implementado, após a fase de análise do problema, é possível definir desenhos da solução a desenvolver apresentando assim alguns diagramas UML para uma mais fácil compreensão.

4.1 Domínio do problema

Um modelo de domínio é uma maneira de representar entidades do mundo real e as relações entre elas [19]. Estas entidades são responsáveis por representar o domínio do problema. Os modelos de domínio são derivados do entendimento dos requisitos a nível de sistema, e a identificação de entidades e a representação de relações entre entidades fornece a base que permite a compreensão do problema.

O modelo de domínio representado na figura 8 demonstra a análise feita do problema e identifica as respetivas entidades do negócio e as relações.

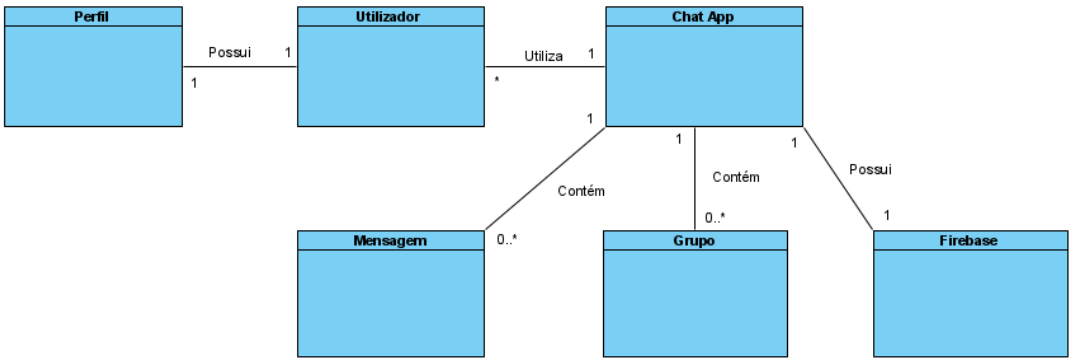


Figura 8 - Modelo de Domínio

Derivado da análise feita é possível identificar que quem vai utilizar a aplicação são os utilizadores, eles terão acesso total a partir do momento que efetuam o login na aplicação.

O principal objetivo desta aplicação é o de permitir que as pessoas possam utilizar o chat de uma forma livre, enviando assim mensagens ou ficheiros entre eles e até mesmo criar grupos para conversarem livremente.

No entanto, esta aplicação deve ser vista como algo mais genérico permitindo a adição de novas funcionalidades futuramente.

4.2 Requisitos funcionais e não funcionais

A compreensão clara do problema apresentado torna-se bastante complicada devido ao facto de o sistema ser novo, como é o caso. Para isso, a identificação específica do que o sistema pode fazer, as funções e as restrições, têm um papel fundamental no desenvolvido inicial e contínuo da aplicação. Neste subcapítulo apresenta-se detalhadamente todos os requisitos do sistema desenvolvido.

4.2.1 Requisitos funcionais

Este tipo de requisitos demonstra as funções que o sistema deve ser capaz de realizar [20]. São responsáveis pela definição do comportamento do sistema, por outras palavras, representa o processo que o software efetua para gerar saídas relativamente às entradas. Neste documento, os requisitos funcionais foram demonstrados através de casos de uso, devido a ser possível, de uma forma mais simples, explicar os processos do sistema e perceber de que forma é que o utilizador interage com o sistema para atingir o objetivo.

Os casos de uso são uma forma muito utilizada de representar graficamente para representar os requisitos funcionais de diferentes sistemas. Este tipo de diagramas corresponde a uma perspetiva externa do sistema, representando assim os atores, os casos de uso e as relações que existem entre ambos.

A figura 9 representa o diagrama de casos de uso correspondente ao problema apresentado, identificando os utilizadores do sistema e as ações que podem realizar.

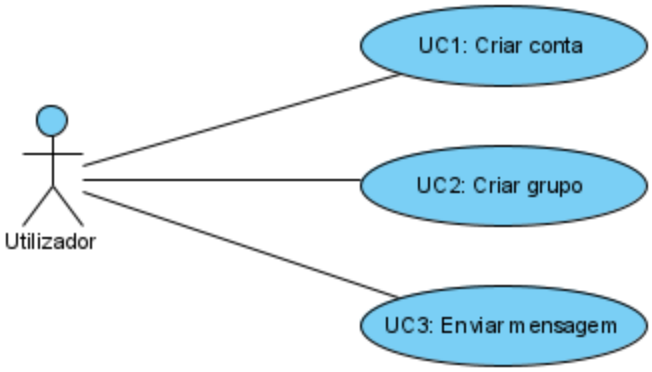


Figura 9 - Diagrama de Casos de Uso

Para os casos de uso demonstrados na figura anterior, é de seguida apresentada uma descrição dos mesmos.

4.2.1.1 UC1: Criar Conta

Este caso de uso é realizado por qualquer utilizador do chat. Permite ao utilizador criar uma conta na aplicação de chat, introduzindo os seus dados para conseguir realizar o registo na aplicação. Seguidamente o sistema é responsável por confirmar se os dados de registo são válidos e por disponibilizar todas as informações necessárias para que o utilizador consiga desfrutar da aplicação.

Na tabela 13 é apresentada uma demonstração mais aprofundada do caso de uso.

Tabela 13 - UC1: Criar conta

OBJETOS	**DESCRIÇÃO**
ATOR	Utilizador.
INTERESSE	Pretende criar uma conta na aplicação, introduzindo os seus dados no registo.
PRÉ-CONDIÇÕES	O sistema tem de estar preparado para conseguir receber novos utilizadores.
PÓS-CONDIÇÕES	Os dados são validados pelo sistema para efeito de autenticação. A utilizador terá de aparecer a todos os utilizadores autenticados na aplicação.
FLUXO	1. O utilizador entra na aplicação no registo; 2. A aplicação solicita ao utilizador para introduzir os seus dados; 3. O utilizador insere os dados necessários; 4. O sistema valida se os dados são válidos; 5. O utilizador é guardado na base de dados; 6. O sistema apresenta a homepage da aplicação.

4.2.1.2 UC2: Criar Grupo

Esta funcionalidade pode ser realizada por qualquer utilizador. Este caso de uso permite ao utilizador criar um grupo, escolhendo um nome e escolhendo que utilizadores é que pretende que sejam inseridos nesse grupo. Posteriormente, quando o grupo é criado, será apresentado a todos os utilizadores que estejam nesse grupo.

Para perceber de uma maneira mais aprofundada o intuito do caso de uso, na tabela 14 é apresentada uma representação do de um formato de caso de uso completo.

Tabela 14 – UC2: Criar Grupo

OBJETOS	**DESCRIÇÃO**
ATOR	Utilizador.
INTERESSE	Pretende criar um grupo na aplicação, atribuindo um nome e escolhendo os utilizadores para o grupo.
PRÉ-CONDIÇÕES	O sistema tem de estar preparado para conseguir receber novos utilizadores; O utilizador tem que estar autenticado.
PÓS-CONDIÇÕES	Os dados escolhidos para o grupo são guardados na base de dados. O grupo tem de ser apresentado a todos os utilizadores que estão inseridos no grupo.
FLUXO	1. O utilizador entra na página de grupos; 2. A aplicação solicita ao utilizador que insira um nome e escolha os utilizadores que pretende atribuir ao grupo; 3. O utilizador insere os dados pedidos; 4. O sistema guarda as seleções do utilizador; 5. O grupo é guardado na base de dados; 6. O sistema apresenta a página de grupos existentes da aplicação.

4.2.1.3 UC3: Enviar Mensagem

Esta caso de uso pode ser realizado por qualquer utilizador. Permite ao utilizador enviar mensagens para outros utilizadores que estejam na aplicação. Quando a mensagem é enviada, o utilizador que recebe a mensagem na aplicação consegue ver a mensagem recebida, sendo de um utilizador ou mais.

Na tabela 15, é possível ver uma descrição mais detalhada do caso de uso, através de um formato completo que explica o fluxo e as condições necessárias para o bom funcionamento.

Tabela 15 – UC3: Enviar Mensagem

OBJETOS	DESCRIÇÃO
ATOR	Utilizador.
INTERESSE	Pretende enviar uma mensagem.
PRÉ-CONDIÇÕES	O sistema tem de estar preparado para conseguir enviar mensagens para outros utilizadores; O utilizador tem que estar autenticado na aplicação.
PÓS-CONDIÇÕES	A mensagem tem de aparecer do lado de quem enviou e do lado de quem recebe a mensagem. Todas as mensagens são guardadas na base de dados.
FLUXO	1. O utilizador escolhe um utilizador; 2. O utilizador escreve uma mensagem e envia para o utilizador selecionado; 3. O sistema guarda a mensagem na base de dados; 4. O sistema mostra a mensagem do lado do utilizador que vê a mensagem e do lado de quem recebe a mensagem.

4.2.2 Requisitos não funcionais

Para identificar os requisitos não funcionais foi utilizado o modelo FURPS+ [21]. Este modelo surge da atualização do modelo FURPS [22], que classifica os requisitos em cinco categorias diferentes, denominados:

- Confiabilidade (Reliability)
- Desempenho (Performance)
- Suportabilidade (Supportability)
- Funcionalidade (Functionality)
- Usabilidade (Usability)
- Outros (+)

Depois da atualização do modelo FURPS, o modelo passou a ser designado por FURPS+, passando assim a existir esta nova classificação dos requisitos não funcionais. Este "+" permite identificar as restrições que são aplicadas ao sistema.

4.2.2.1 Confiabilidade

A confiabilidade do sistema refere-se à integridade e conformidade da aplicação. Neste parâmetro os requisitos a serem considerados são a gravidade das falhas e a possibilidade de recuperação. Para o projeto desenvolvido não foram identificados quaisquer tipo de requisitos não funcionais de confiabilidade.

4.2.2.2 Desempenho

O desempenho é responsável dos requisitos que tratam da avaliação do desempenho do software, como por exemplo, o tempo de resposta, utilização de recursos e a capacidade de leitura. Para o presente projeto não foram identificados requisitos não funcionais de desempenho.

4.2.2.3 Suportabilidade

Esta categoria de suportabilidade demonstra vários tipos de requisitos, como: escalabilidade, compatibilidade, adaptabilidade, requisitos de testabilidade, entre outros. Para o presente projeto foram identificados os seguintes requisitos de suportabilidade:

- O sistema desenvolvido deve permitir a integração de novas funcionalidades;
- O sistema deve permitir a escalabilidade.

4.2.2.4 Funcionalidade

A funcionalidade permite identificar funcionalidades que não é possível relacionar com os casos de uso. Relativamente ao projeto desenvolvido foi identificado o seguinte requisito de funcionalidade:

- O chat deve guardar a informação na base de dados do Firebase.

4.2.2.5 Usabilidade

Esta categoria de usabilidade é responsável pela avaliação da interface, ou seja, todos os requisitos que dizem respeito às interações do utilizador com o sistema. Para o projeto foi identificado o seguinte requisito de usabilidade:

- Pretende-se que a aplicação desenvolvida possua uma interface simples, que seja de rápida utilização por parte dos utilizadores.

4.2.2.6 Outros (+)

Este parâmetro permite a distribuição de novos requisitos não funcionais que permitem assegurar a qualidade de software. Este contém quatro categorias: requisitos físicos, requisitos de interface, requisitos de implementação e restrições de design. Para o projeto apresentado, foi identificado o seguinte requisito:

- A aplicação deve ser desenvolvida com a utilização de processos iterativos.

4.3 Desenho

O presente subcapítulo tem como objetivo demonstrar a forma como a aplicação está estruturada, apresentando assim, a vista lógica, vista de implementação, vista de implantação e vista de cenários.

4.3.1 Vista lógica

O principal objetivo deste tipo de diagrama de componentes é o de ilustrar a relação entre as diversas componentes do sistema e as respetivas interfaces. Na figura 10 é possível observar a representação do diagrama de componentes. Tendo em conta a granularidade do sistema é possível destacar as duas componentes existentes, que funcionam com aplicações distintas. Uma das componentes é responsável por guardar todos os dados e informações importantes para a aplicação, sendo que a outra componente é responsável pela camada de apresentação.

A separação das duas aplicações distintas pode acrescentar algumas vantagens à aplicação, uma vez que, visto que toda a lógica de aplicação é feita consoante a linguagem de React, a alteração futura da aplicação bem como da linguagem tornam-se mais simples, visto que todas as informações relativas a utilizadores estão guardadas noutra aplicação.

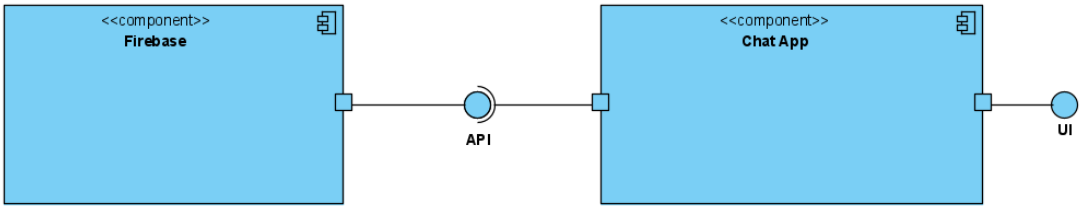


Figura 10 - Vista lógica

4.3.2 Vista implementação

Na figura 11 é possível encontrar uma representação da estrutura da aplicação implementada. Este diagrama de pacotes divide o sistema por lógica, mostrando também as suas dependências. Tendo em conta a granularidade do sistema destaca-se dois pacotes, um que é responsável pela camada de apresentação e lógica da aplicação, ou seja, toda a componente visuais do sistema desenvolvido e outro pacote responsável por guardar toda a informação da aplicação.

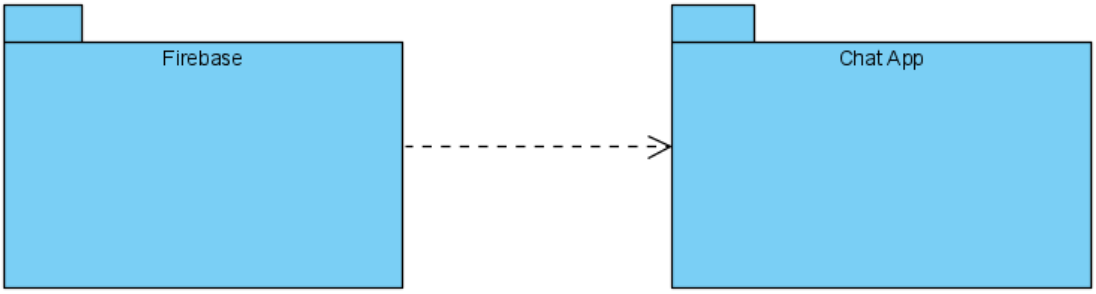


Figura 11 - Vista de Implementação

4.3.3 Vista implantação

É possível mapear a arquitetura do hardware de acordo com as necessidades do software a ser implementado através de diagramas de implantação. Para isso prevê-se que exista dois servidores responsáveis pelo armazenamento dos dados e pelo armazenamento da camada de apresentação e lógica da aplicação.

Esta separação entre as duas componentes diferentes, consegue acrescentar algumas vantagens ao sistema. Por existir esta separação, não existe uma competição pelos mesmos recursos, assim sendo, se pudessem existir cargas para o servidor, estas são eliminadas para que não exista a possibilidade de interferir no desempenho da aplicação. Para além disso, a separação entre as duas componentes diferentes assegura a escalabilidade do sistema implementado, o que faz com que a integração de novas funcionalidades ou de alteração da linguagem ou da aplicação em si se torne mais fácil de implementar.

Na figura 12 estão representadas as dependências entre as componentes do sistema.

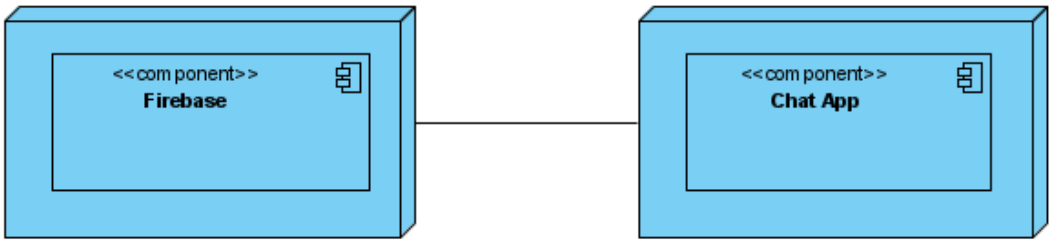


Figura 12 - Vista de Implantação

4.3.4 Vista de cenários

Nesta subsecção são apresentadas as vistas de cenários para os casos de uso desenvolvidos. Para uma melhor compreensão do funcionamento de algumas funcionalidades do sistema, foram assim representados em diagramas, os fluxos que o utilizador e o sistema realizam até concluir os objetivos definidos para cada caso de uso.

É necessário ter em conta que em todos os diagramas apresentados nesta subsecção, a componente Chat App é responsável pela apresentação dos dados, bem como, a ligação entre como é feita a apresentação e como o sistema foi desenvolvido, por outras palavras, a lógica que define de que maneira o sistema deve responder relativamente a uma determinada interação com o utilizador. Por sua vez, visto que o sistema não era capaz de funcionar sem este sistema externo, a componente Firebase é responsável por guardar todas as informações introduzidas na aplicação.

UC1: Criar Conta

Na figura 13 é possível encontrar uma representação gráfica do diagrama para o caso de uso de criar uma conta.

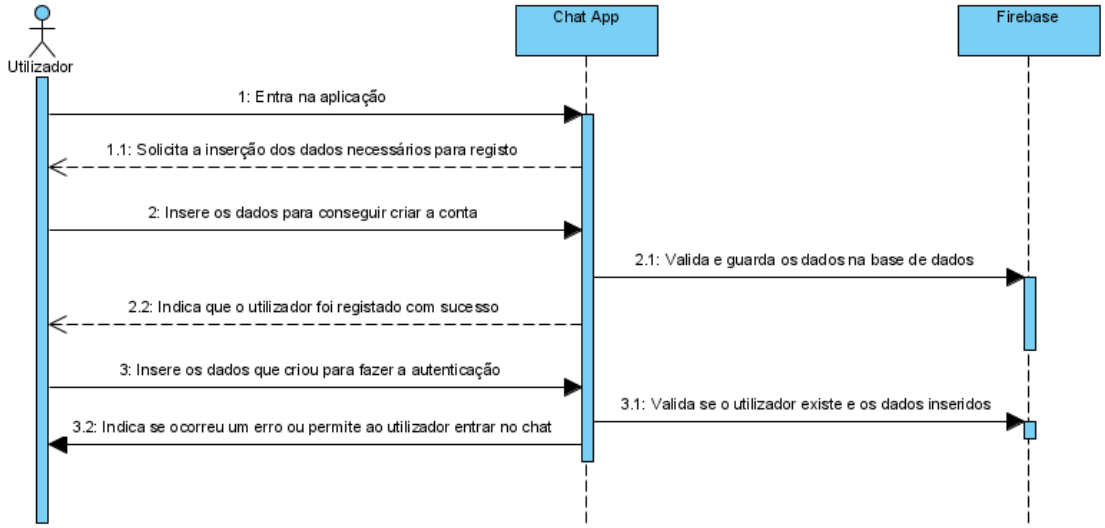


Figura 13 – Diagrama do UC1: Criar Conta

UC2: Criar Grupo

Na figura 14 corresponde à representação gráfica do diagrama para o caso de uso de criar um grupo.

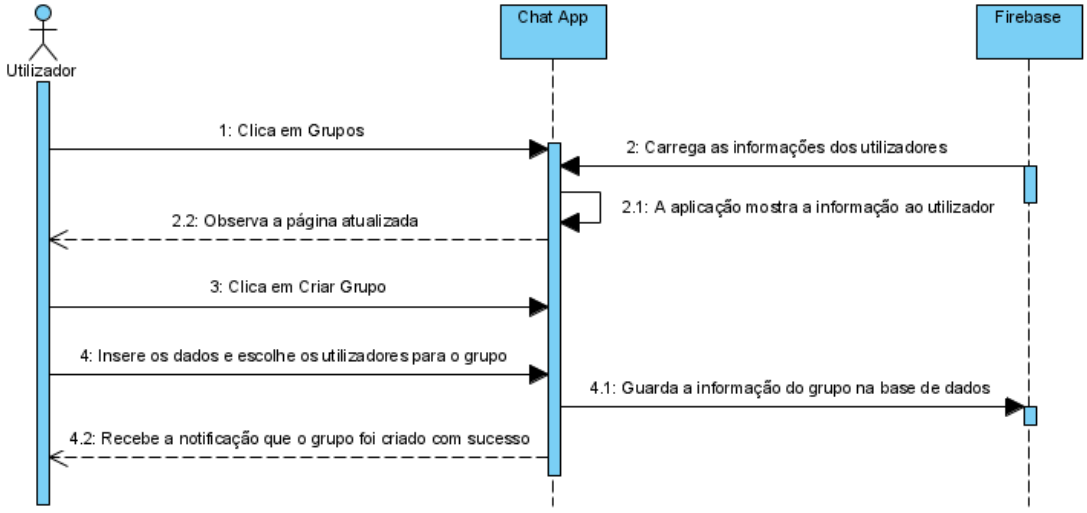


Figura 14 – Diagrama do UC2: Criar Grupo

UC3: Enviar mensagem

Na figura 15 encontra-se uma representação gráfica do diagrama para o caso de uso de enviar uma mensagem.

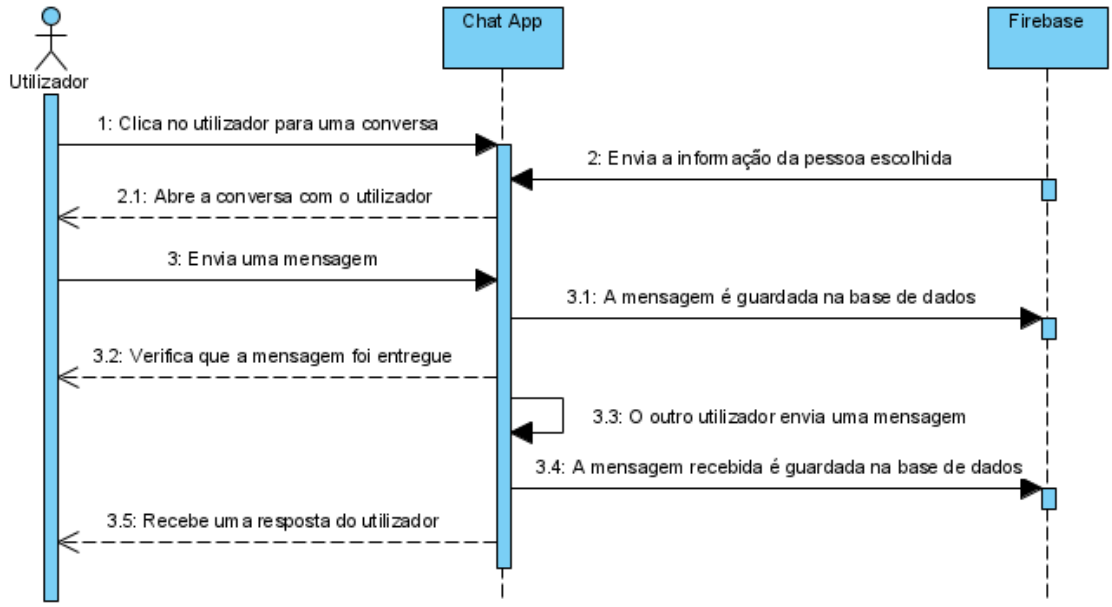


Figura 15 – Diagrama do UC3: Enviar Mensagem

5 Implementação da Solução

No presente capítulo é apresentado a descrição da implementação da solução juntamente com uma breve análise crítica ao trabalho desenvolvido.

5.1 Descrição da implementação

Neste capítulo é possível encontrar uma descrição mais detalhada da implementação da solução. Para uma mais fácil compreensão do trabalho realizado, ao longo da explicação é possível encontrar alguns excertos de código e imagens da aplicação, para conseguir acompanhar o que foi feito.

É importante demonstrar que o sistema que foi desenvolvido foi dividido em duas partes, Frontend e Backend. A primeira parte é a que trata da camada de apresentação ao utilizador, ou seja, é tudo o que vai ser apresentado ao utilizador e que ele consegue interagir. A segunda parte é a que trata de guardar os dados que são introduzidos e que contem a lógica por detrás da aplicação.

5.1.1 Firebase

Para começar a construir a aplicação foi necessário gerar uma nova aplicação em React e para isso é necessário ter o npm instalado no computador e de seguida colocar num terminal o seguinte comando:

```
npx create-react-app my-app
cd my-app
npm start
```

Figura 16 - Criar aplicação base

O que o primeiro comando faz é criar uma aplicação base onde conseguimos começar a construir a aplicação que desejamos e iniciar a mesma com o npm start.

Para ser possível utilizar o Firebase, para servir de base de dados e autenticador para a aplicação, é necessário criar um projeto novo. Na figura 17 é possível ver o aspeto da consola do Firebase.

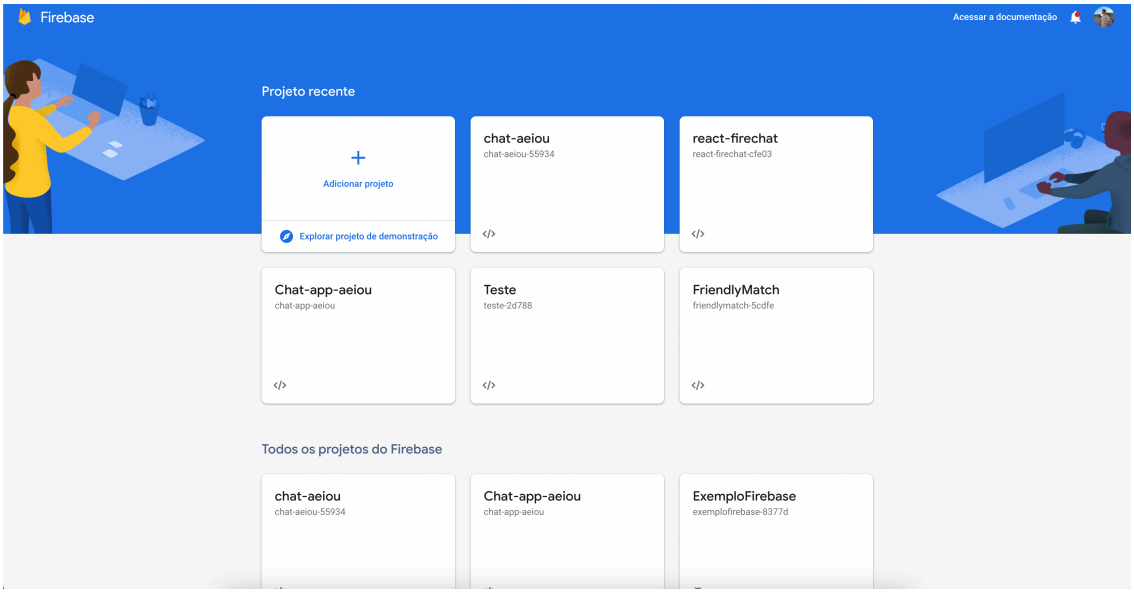


Figura 17 - Consola para criação da aplicação

É necessário atribuir um nome ao projeto que estamos prestes a criar, como é pedido na figura 18.

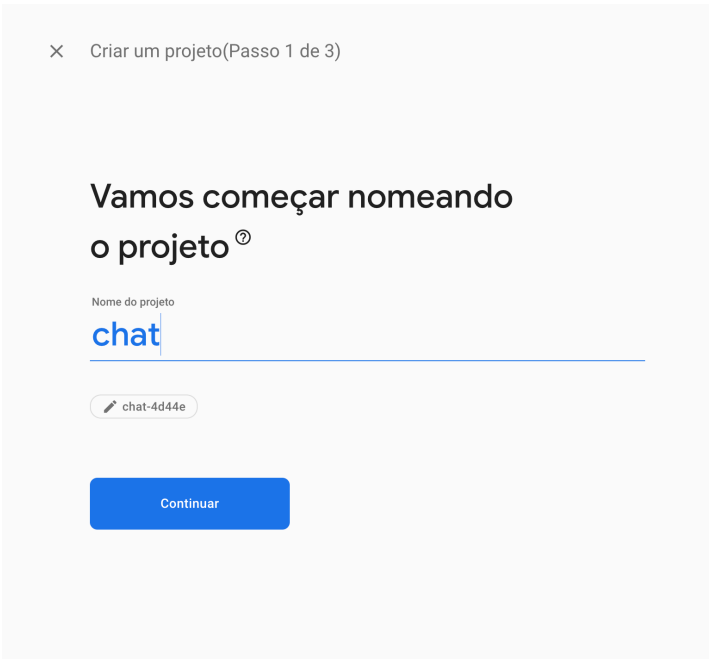


Figura 18 - Atribuir nome ao projeto

Era possível utilizar o Google Analytics para retirar dados acerca do tráfico do site, mas que para o sistema que foi desenvolvido, devido a estar numa fase inicial, não necessitava de utilizar esta ferramenta. Para continuar simplesmente desativamos a opção do “Ativar o Google Analytics neste projeto” e carregamos em “Criar Projeto”.

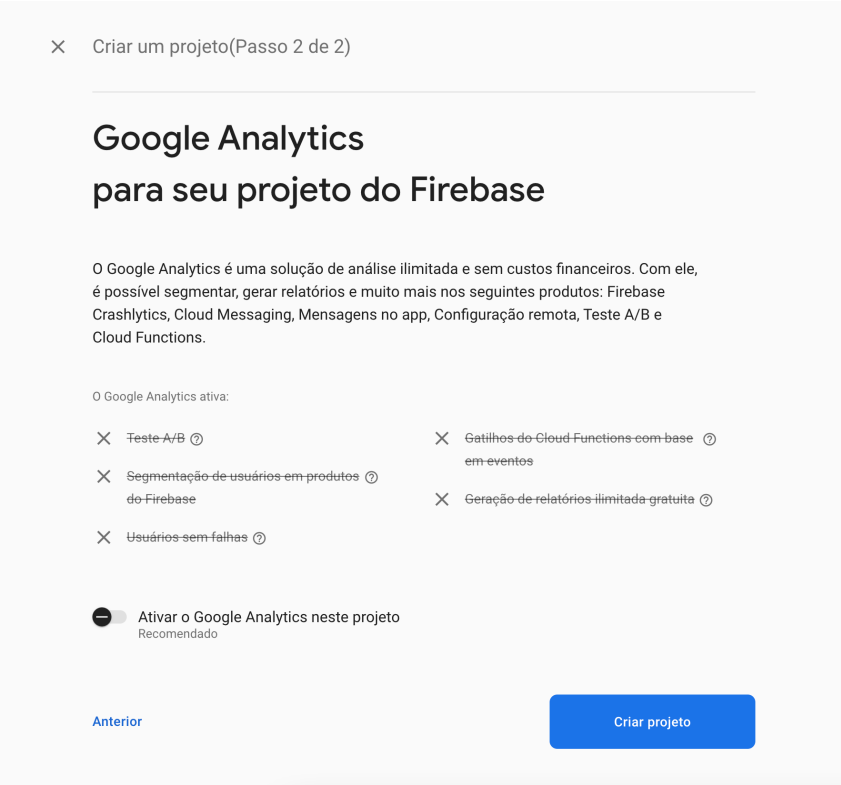


Figura 19 - Google Analytics

Devido ao facto de o sistema ser um chat, é importante ter autenticação para o utilizador ser identificado. Para isso, no Firebase existe uma opção chamada “Authentication” que permite ativar o tipo de autenticação que precisamos no projeto. Para o chat apenas precisamos de ativar o Email/Senha para se tornar mais simples, mas podemos futuramente ativar outros tipos de autenticação. Na figura 20 mostra o botão para ativar a funcionalidade de autenticação.

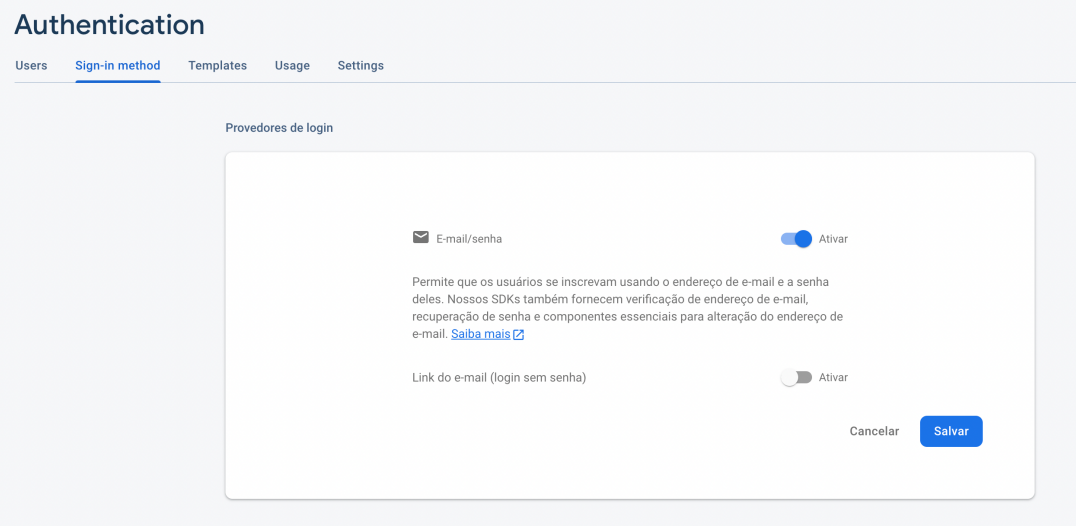


Figura 20 - Autenticador por Email e Senha

Para guardar os dados dos utilizadores e as mensagens que enviam no chat, foi importante ativar o Cloud Firestore no Firebase para esse efeito. Esta funcionalidade permite guardar os dados de uma forma mais intuitiva e rápida. Utiliza um efeito em árvore, quando os dados são guardados, que torna tudo mais simples e de fácil compreensão para quem trata da base de dados da aplicação, neste caso o desenvolvedor. Esta base de dados tem dois tipos de modos, o de produção e o de teste. Inicialmente é utilizado o modo de teste para efeitos de debug e testagem da aplicação. Esta base de dados permite também incluir um conjunto de regras que podem ser modificadas a qualquer momento para que os dados sejam guardados em segurança e permitir a leitura e gravação de dados. Na figura 21 é possível ver a criação da Cloud Firestore.

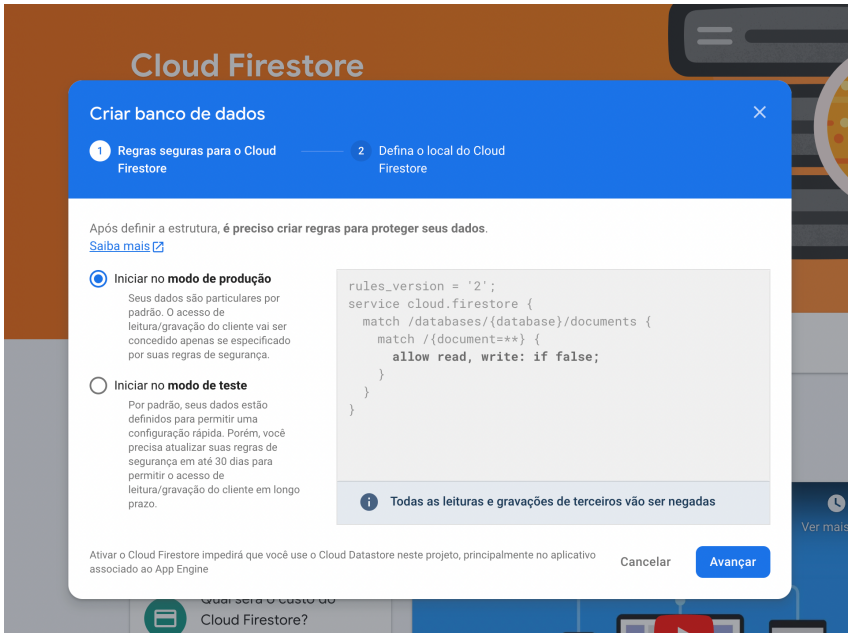


Figura 21 - Criação da Base de Dados

Depois de criada a base de dados e de gerar alguns dados, é possível ver o efeito em árvore que foi referido anteriormente, e que só é capaz de realizar isso devido às coleções criadas. Estas coleções são as que produzem organização dentro da base de dados do Firebase. Quando o código está em desenvolvimento, é possível importar librarias que já existem no Firebase e criar coleções automaticamente. Na figura 22 é demonstrado o aspeto de uma base de dados vazia.

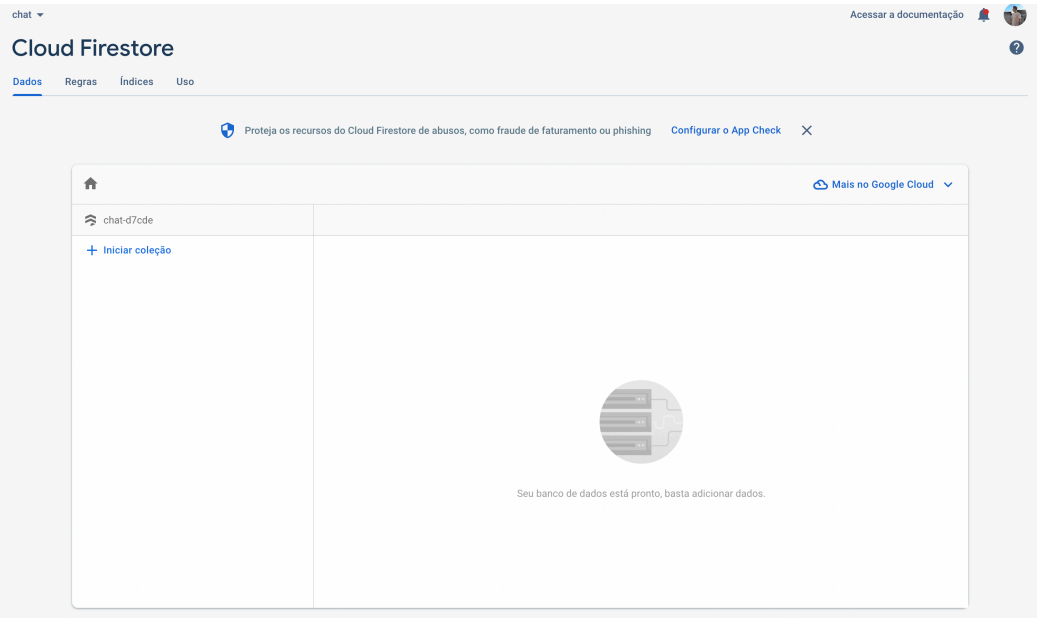


Figura 22 - Base de dados da Cloud Firestore

Para ser possível utilizar a base de dados e a autenticação que foi criado anteriormente, é importante registar a aplicação que pretendemos utilizar. Para isso é necessário abrir as configurações do projeto no Firebase e carregar na plataforma que desejamos, como na figura 23. De seguida, atribuir um nome ao projeto para registar a aplicação e carregar em “Registar app”, como é possível observar na figura 24.

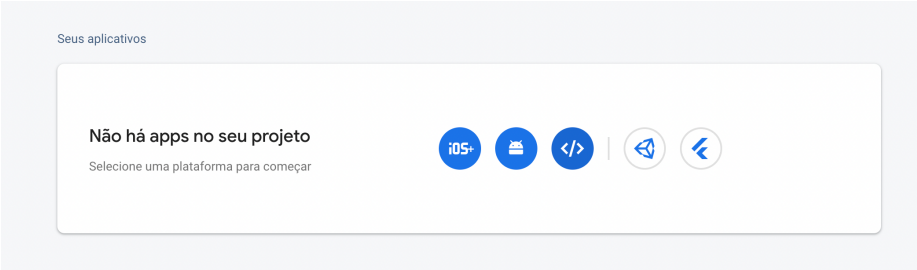


Figura 23 - Plataforma para projeto

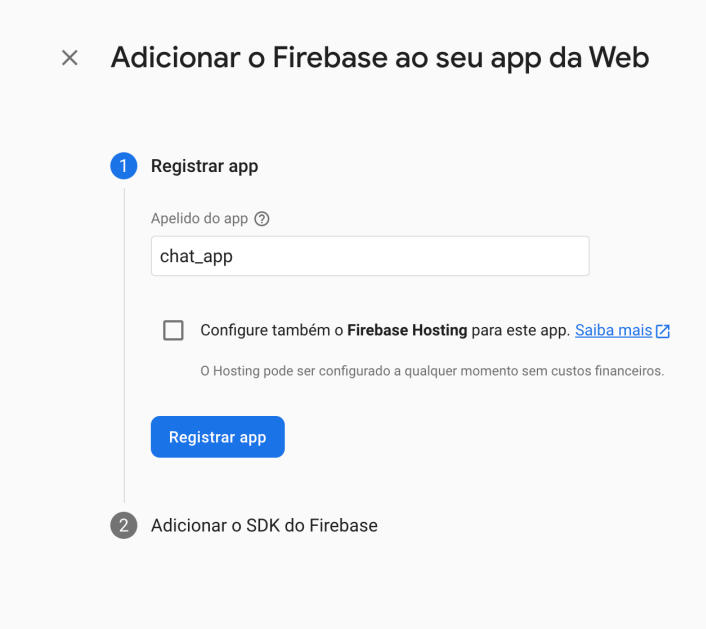


Figura 24 - Registar aplicação

Concluídos estes passos, serão gerados SDK's importantes para a utilização da aplicação. Na figura 25 é demonstrado o que é necessário importar para o projeto para ser possível utilizar o Firebase. No caso do projeto desenvolvido, foi criado um ficheiro .env que contem as chaves para o correto funcionamento da aplicação.

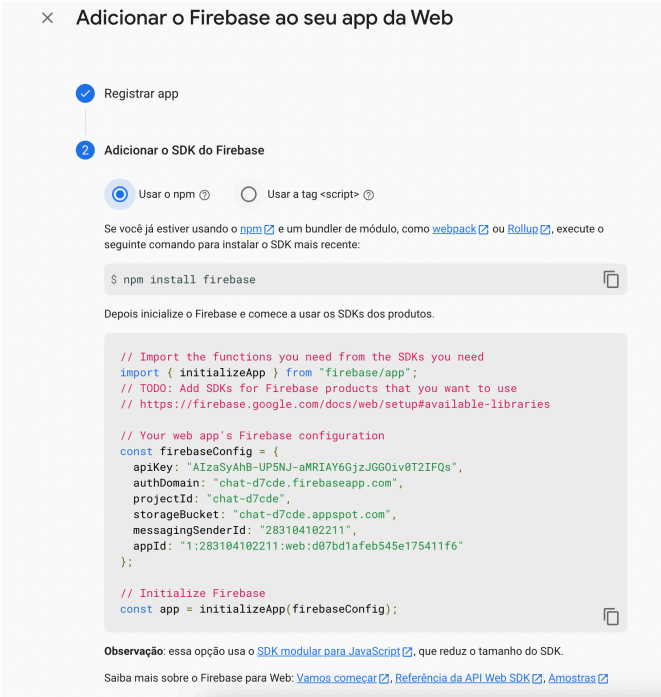


Figura 25 - SDK para a aplicação

Posto isto, nos próximos subcapítulos serão apresentados alguns tipos de excertos de códigos de ficheiros que apresentam conteúdo importante para o sistema desenvolvido.

5.1.2 React e excertos de código

Esta secção descreve as decisões tomadas em relação à implementação da solução bem como uma descrição mais detalhada de alguns elementos essenciais da aplicação. As componentes foram desenvolvidas com base numa biblioteca de JavaScript designada de ReactJS[9], que foi analisada detalhadamente no subcapítulo 2.3.1.

Existem algumas caraterísticas do React que são importantes de serem referidas como os componentes e os estados. Os componentes do React permitem que a interface gráfica seja dividida em partes lógicas, tornando assim cada componente separável e reutilizável. Desta forma, cada componente pode ser manipulado como um bloco independente sem dependências externas.

As componentes React são funções de JavaScript que aceitam dados de entrada, sendo que esses dados funcionam como propriedades para a sua construção, retornando assim elementos React que descrevem a estrutura da interface gráfica. Estas componentes também são responsáveis pela manipulação dos dados apresentados ao utilizador e asseguram a interação do utilizador com o sistema. Na figura 26 é possível encontrar um excerto de código de uma componente React utilizada na implementação da solução.

```
import React from "react";
import Attachment from "./svg/Attachment";

const MessageForm = ({ handleSubmit, text, setText, setImg }) => {
  return (
    <form className="message_form" onSubmit={handleSubmit}>
      <label htmlFor="img">
        <Attachment />
      </label>
      <input
        onChange={(e) => setImg(e.target.files[0])}
        type="file"
        id="img"
        accept="image/*"
        style={{ display: "none" }}
      />
      <div>
        <input
          type="text"
          placeholder="Enter message"
          value={text}
          onChange={(e) => setText(e.target.value)}
        />
      </div>
      <div>
        <button className="btn">Send</button>
      </div>
    </form>
  );
};

export default MessageForm;
```

Figura 26 - Excerto de uma componente React

[9] https://reactjs.org/

Após uma breve análise da figura anterior é possível verificar que as componentes React funcionam como funções de JavaScript, sendo que recebem dados de entrada (neste caso são o “handleSubmit”, “text”, “setText”, “setImg”) e retornam elementos React escritos com uma sintaxe muito parecida ao XML, designada por JSX[10] e já mencionada no subcapítulo 2.3.1.2.

Uma outra caraterística do React são os estados, que são objetos JavaScript que começam com um valor padrão quando um componente está a ser construído. No entanto, os estados React ao longo do tempo vão sofrendo alterações que resultam de eventos despoletados pelos utilizadores do sistema.

Cada componente possui os dados próprios iniciais, mas tendo em conta o problema apresentado, era necessário que o componente guardasse os dados inseridos pelo utilizador e enviasse os mesmos para o Firebase (o que é possível através das librarias que o Firebase permite utilizar).

Para ser possível reutilizar os estados do React para cada utilizar e torná-los mais simples e reutilizáveis, decidiu-se o que é possível observar na figura 27.

```
import React, { useState } from "react";
import { createUserWithEmailAndPassword } from "firebase/auth";
import { auth, db } from "../firebase";
import { setDoc, doc, Timestamp } from "firebase/firestore";
import { useHistory } from "react-router-dom";

const Register = () => {
  const [data, setData] = useState({
    name: "",
    email: "",
    password: "",
    error: null,
    loading: false,
  });

  const history = useHistory();

  const { name, email, password, error, loading } = data;

  const handleChange = (e) => {
    setData({ ...data, [e.target.name]: e.target.value });
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    setData({ ...data, error: null, loading: true });
    if (!name || !email || !password) {
      setData({ ...data, error: "All fields are required" });
    }
```

Figura 27 - Excerto de um estado React

Na figura 28 verifica-se os dados que são passados para o estado do React, armazenados no Firebase, relativamente ao utilizador que se encontra autenticado na conta.

[10] https://reactjs.org/docs/introducing-jsx.html

Figura 28 - Excerto de um utilizador autenticado

Após uma breve análise do código é possível verificar que existem librarias como o "useState" do React e o "auth" do Firebase que tornam o desenvolvimento do registar um utilizar muito mais simples e intuitivo. É também possível ver que o estado inicial do useState é vazio até ao momento em que o utilizador introduz dados válidos na aplicação, o que faz com que o utilizador consiga aceder ao chat e os dados inseridos sejam automaticamente guardados na base de dados do Firebase.

Explicando outras caraterísticas importantes da aplicação desenvolvida, para uma melhor compreensão da aplicação, é possível um dos ficheiros mais importantes da aplicação chamado App.js. Este ficheiro é como se fosse um container para o resto de todas as componentes do sistema tendo também um ficheiro chamado App.css que engloba e proporciona estilos para o resto dos componentes.

O ficheiro index.js é um ficheiro JavaScript que corresponde a um index.html. O código que existe dentro deste ficheiro serve apenas para indicar que a componente App tem de ser carregada para um elemento html com o id root, resumindo, será um elemento div no ficheiro index.html. Os estilos presentes dentro do ficheiro index.css servem para indicar as fontes globais utilizadas na aplicação.

No caso da aplicação desenvolvida, existe a capacidade de indicar que tipo de Route é que feito, por exemplo, se o utilizador não estiver autenticado ele não permite entrar no chat, apenas permite ir à página de registar e entrar na conta. Para este efeito, também foi implementado um PrivateRoute que apenas é permitido o Route para páginas em que o utilizador é permitido e está assim autenticado. Existem assim 3 parâmetros para o Route: o component, o path e o exact. O component recebe qual é o componente que vai ser mostrado quando se acede aquela página. O path é o URL que precisa de ser mostrado para mostrar a component selecionada, que é definido pelo component. O exact vai determinar qual é o component que vai ser exibido sendo necessário estar em conformidade com o que se encontra dentro de aspas. Na figura 29 é demonstrado alguns Route e PrivateRoute da aplicação.

Figura 29 - Exemplo de Route e PrivateRoute da aplicação

Na figura 30 é demonstrado o código do login desenvolvido para a aplicação.

```
const Login = () => {
  const [data, setData] = useState({
    email: "",
    password: "",
    error: null,
    loading: false,
  });

  const history = useHistory();

  const { email, password, error, loading } = data;

  const handleChange = (e) => {
    setData({ ...data, [e.target.name]: e.target.value });
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    setData({ ...data, error: null, loading: true });
    if (!email || !password) {
      setData({ ...data, error: "Todos os campos são obrigatórios." });
    }
    try {
      const result = await signInWithEmailAndPassword(auth, email, password);

      await updateDoc(doc(db, "users", result.user.uid), {
        isOnline: true,
      });
      setData({
        email: "",
        password: "",
        error: null,
        loading: false,
      });
      history.replace("/");
    } catch (err) {
      setData({ ...data, error: err.message, loading: false });
    }
  };
  return (
    <section>
      <h3>Entre na sua conta</h3>
      <form className="form" onSubmit={handleSubmit}>
        <div className="input_container">
          <label htmlFor="email">Email</label>
          <input
            type="text"
            name="email"
            value={email}
            onChange={handleChange}
          />
        </div>
        <div className="input_container">
          <label htmlFor="password">Password</label>
          <input
            type="password"
            name="password"
            value={password}
            onChange={handleChange}
          />
        </div>
        {error ? <p className="error">{error}</p> : null}
        <div className="btn_container">
          <button className="btn" disabled={loading}>
            {loading ? "Logging in ..." : "Login"}
          </button>
        </div>
      </form>
    </section>
  );
};

export default Login;
```

Figura 30 - Código da página de Login

Analisando a figura, para conseguir implementar o Login, foi importante importar as librarias que são necessárias para o funcionamento desta página. Estas librarias possuem funcionalidades já criadas, por exemplo do Firebase, que permitem a utilização de certas funções, tornando o código mais simples e reutilizável. No princípio é definido um useState inicial para ter sempre um estado vazio atribuído ao email, password, um estado null ao error e um estado false ao loading. Este tipo de useState são importantes para a aplicação devido

ao facto de ser possível utilizar o estado definido até ao momento em qualquer lugar do código da aplicação.

O useHistory é uma funcionalidade do React Router, que permite aceder à instância de uma história, permitindo assim redirecionar os utilizadores para outra página.

O handleChange é uma variável que irá atribuir valores ao estado, utilizando para isso um useState, a partir do momento que os valores alteram no formulário de login.

O handleSubmit é uma variável que apenas submete os dados introduzidos pelo utilizador realizando assim as validações necessárias aos dados introduzidos. Em caso de erro irá mostrar uma mensagem ao utilizador de maneira que ele perceba o que é necessário alterar para conseguir aceder à sua conta. No caso de ser bem-sucedido será guardado na base de dados do Firebase, criando assim uma coleção chamada de users (no caso de não existirem utilizadores na base de dados, senão apenas guarda na coleção) e guarda o utilizador gerando assim um código para esse utilizador. De seguida o state da data será atualizado para voltar a estar vazio. Na parte do return, tudo o que está dentro é o que é possível ver pelo utilizador, o que significa que é a parte que é renderizada para ser visível no Frontend. Na figura 31 é demonstrada a página de Login do utilizador.

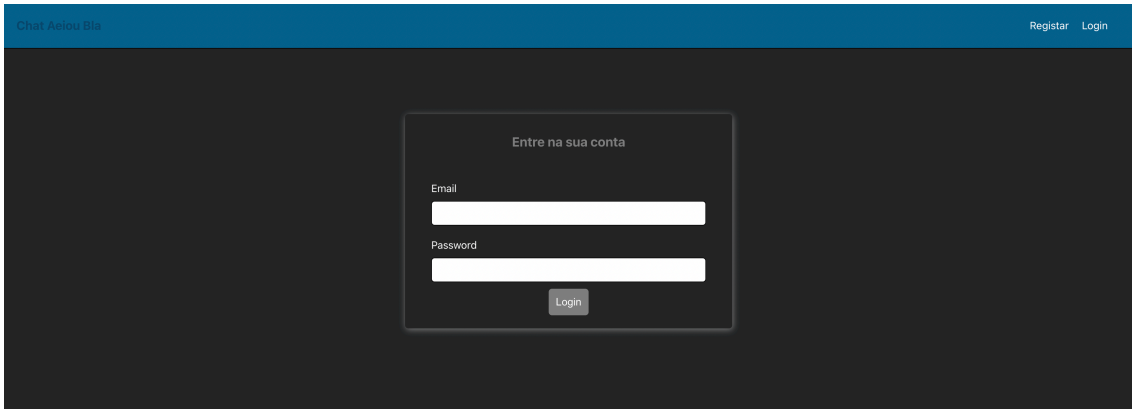


Figura 31 - Página de Login

No caso de o utilizador ser bem-sucedido a fazer o Login, ele é redirecionado para a página Home onde poderá escolher as pessoas com quem quer conversar, podendo assim iniciar uma conversa, como demonstrado na figura 32.

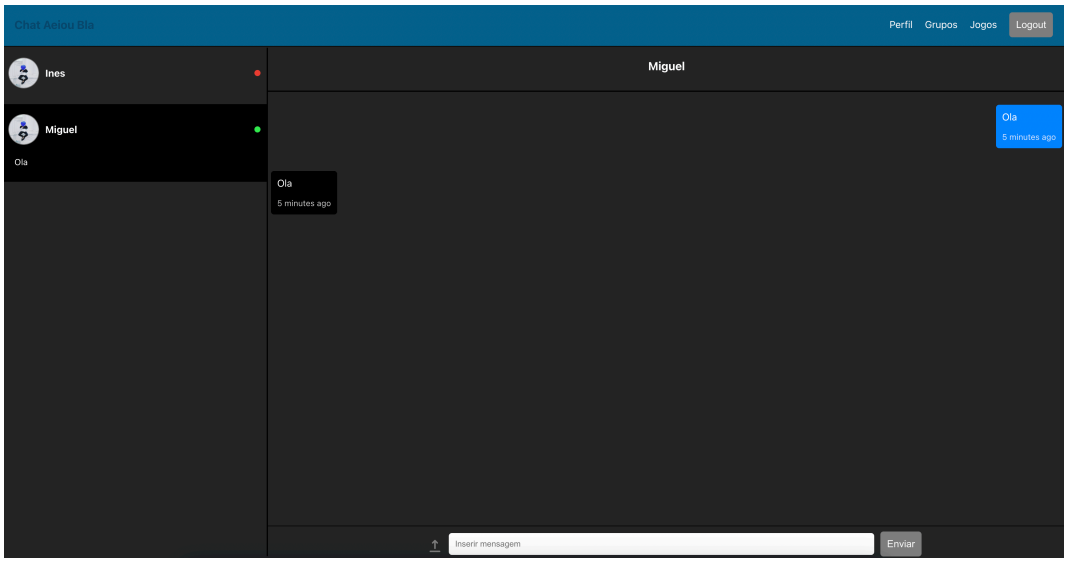


Figura 32 - Chat com outro utilizador na página Home

Na figura 33 e 34 é possível observar excertos do código desenvolvido para a aplicação desenvolvida, que tinha como objetivo a criação de grupos para o chat.

```
const Groups = () => {
  const [users, setUsers] = useState([]);
  const [groups, setGroups] = useState([]);
  const [open, setOpen] = React.useState(false);
  const [chat, setChat] = useState("");
  const [text, setText] = useState("");
  const [img, setImg] = useState("");
  const [state, setState] = React.useState([]);
  const [msgs, setMsgs] = useState([]);
  const handleOpen = () => setOpen(true);
  const handleClose = () => setOpen(false, setState([]));
  const [groupName, setGroupName] = useState('');

  const style = {
    position: 'absolute',
    top: '50%',
    left: '50%',
    transform: 'translate(-50%, -50%)',
    width: 400,
    backgroundColor: '#0084ff',
    color: 'white',
    border: '2px solid #000',
    boxShadow: 24,
    p: 4,
  };

  const user1 = auth.currentUser.uid;

 useEffect(() => {
  const groupsRef = collection(db, "groups");
  const unsubgroup = onSnapshot(groupsRef, (querySnapshot) => {
    let groups = [];
    querySnapshot.forEach((doc) => {
      groups.push(doc.data());
    });
    setGroups(groups);
    });
  return () => unsubgroup();
  }, []);

  useEffect(() => {
    const usersRef = collection(db, "users");
    // create query object
    const q = query(usersRef, where("uid", "not-in", [user1]));
    // execute query
    const unsub = onSnapshot(q, (querySnapshot) => {
      let users = [];
      querySnapshot.forEach((doc) => {
        users.push(doc.data());
      });
      setUsers(users);
    });
    return () => unsub();
  }, []);

  const selectGroup = async (group) => {
    setChat(group);

    const group2 = group.uid;
    const id = user1 > group2 ? `${user1 + group2}` : `${group2 + user1}`;

    const msgsRef = collection(db, 'messagesGroup', id, "chat");
    const q = query(msgsRef, orderBy("createdAt", "asc"));

    onSnapshot(q, (querySnapshot) => {
      let msgs = [];
      querySnapshot.forEach((doc) => {
        msgs.push(doc.data());
      });
      setMsgs(msgs);
    });

    //get last msg by logged in user and selected group
    const docSnap = await getDoc(doc(db, "lastMsgGroup", id));
    // if last msg exists and message is from selected group
    if (docSnap.data() && docSnap.data().from !== user1) {
      //update last msg doc, set unread to false
      await updateDoc(doc(db, "lastMsgGroup", id), { unread: false});
    }
  };
```

Figura 33 - Código da página de Group

```
const handleSubmit = async (e) => {
  e.preventDefault();

  const group2 = chat.uid;
  const id = user1 > group2 ? `${user1 + group2}` : `${group2 + user1}`;

  let url;
  if (img) {
    const imgRef = ref(
      storage,
      `images/${new Date().getTime()} - ${img.name}`
    );
    const snap = await uploadBytes(imgRef, img);
    const dlUrl = await getDownloadURL(ref(storage, snap.ref.fullPath));
    url = dlUrl;
  }

  await addDoc(collection(db, "messagesGroup", id, "chat"), {
    text,
    from: user1,
    to: group2,
    createdAt: Timestamp.fromDate(new Date()),
    media: url || "",
  });

  await setDoc(doc(db, "lastMsgGroup", id), {
    text,
    from: user1,
    to: group2,
    createdAt: Timestamp.fromDate(new Date()),
    media: url || "",
    unread: true,
  });

  setText("");
  setImg("");
}

const handleGroupNameChange = event => {
  setGroupName(event.target.value);
};


const handleChange = (event) => {
  // verify if name is selected == true
    if (event.target.checked === true){
      // verify if name already exists in array
      var indexTrue = state.findIndex((item) => {
        return item.name === event.target.name && item.checked === event.target.checked
      });
      if (indexTrue === -1) {
        setState([ ...state, { uid:event.target.id, name:event.target.name, checked: event.target.checked} ]);
      }
    // verify if name is not selected == false
    } else {
      // verify if name already exists in array
      var indexFalse = state.findIndex((item) => {
        return item.name === event.target.name && item.checked === event.target.checked
      });
      if (indexFalse === -1) {
        setState([ ...state,{ uid:event.target.id, name:event.target.name, checked: event.target.checked} ]);
      }
    }
  }

const error = state.filter((c) => c).length !== 2;

let uid = (Math.random() + 1).toString(36).substring(2);

const removeUsers = () => {
  setState(state.filter(state => {
      return state.checked !== false;
    })
  );

  setGroups({groupName, state});
  addDoc(collection(db, "groups"), {
    uid,
    groupName,
    users:state.map(id => id.uid),
    createdAt: Timestamp.fromDate(new Date()),
  });
  handleClose();
}
```

Figura 34 - Código da página de Group

Analisando as figuras anteriores, para todo o código demonstrado funcionar corretamente, inicialmente foi necessário importar as librarias necessárias devido à utilização de bibliotecas do material-UI[11], do Firebase e do React. A utilização do material-UI foi necessária para a construção do modal que aparece quando o adicionar grupo é carregado pelo utilizador, a utilização do Firebase foi para ser possível realizar queries e guardar os dados utilizados para o chat na base de dados e, por fim, a utilização do React para ser possível tirar partido do useState e do useEffect.

Como é demonstrado na figura 33, é visível a utilização do useEffect que permitiu realizar uma procura na base de dados acerca dos grupos existentes, mas também permitiu que se executasse uma query que fosse buscar todos os utilizadores excluindo o utilizador que se encontra atualmente autenticado. Estes useEffect são capazes de comunicar com o React indicando que precisam de fazer ações depois de tudo ser renderizado. O que o React vai fazer é lembrar-se do que foi passado nestes useEffect e chamá-los mais tarde depois de atualizar o DOM.

[11] https://mui.com/pt/

De notar que existe também um selectGroup, na figura 33, que é o que trata de fazer um setChat que indica qual é o grupo que o utilizador clicou. Dentro do grupo selecionado é feita uma query para ser possível guardar as mensagens enviadas naquele grupo ordenadamente, na base de dados na coleção messagesGroup(guardando na coleção messagesGroup com um certo id e na coleção chat), como também é guardado noutra coleção as lastMsgGroup(guardando nas lastMsgGroup com o id criado nas messagesGroup para saber onde foram enviadas as mensagens), que permitem obter as últimas mensagens no caso do utilizador ter saído da aplicação como também indica que mensagens é que não foram lidas, (aparecendo assim uma notificação a indicar que existem mensagens novas por ler) atualizando por mim o atributo unread na base de dados no caso das mensagens não terem sido enviadas por parte do utilizador autenticado.

Na figura 34, existe um handleSubmit que trata como é que as mensagens são enviadas e guardadas na base de dados. Inicialmente começa com um preventDefault que evita que a ação default do evento não irá ocorrer. De seguida existe uma variável url que trata dos ficheiros de imagens que são enviados para o chat, que utiliza uma das funcionalidades do Firebase, o storage. Esta funcionalidade storage irá guardar as imagens enviadas no Firebase numa pasta images com um certo nome. Existe também nesta função um addDoc e um setDoc, este addDoc permite guardar os dados inseridos pelo utilizador na base de dados, contendo assim na coleção messagesGroup, text com o conteúdo da mensagem enviada, pelo user1 sendo o utilizador autenticado, para o group2 que é o id do grupo em que o utilizador enviou a mensagem, createdAt sendo a data que a mensagem foi enviada e a media contendo o url do storage da imagem enviada (caso tenha sido enviada alguma, se não tiver nada terá o campo vazio). O setDoc irá atualizar a coleção lastMsgGroup na base de dados com todos os dados referidos anteriormente e atualizar o campo unread para true, para o caso de outro utilizador ver a mensagem e receber a mensagem no seu chat no caso de ainda não ter lido a mensagem.

É possível ver uma função handleGroupNameChange que guarda o nome do grupo inserido pelo utilizador no momento da criação do grupo na variável global groupName.

Seguidamente, existe a função handleChange que tem como objetivo atribuir ao state os utilizadores que o utilizador escolheu para inserir no grupo no momento da criação do mesmo. Começa por validar se no atributo checked está a true ou a false onde de seguida irá verificar que index é que aquele state possui, de forma a validar se aquele utilizador já existe no array criado, se não existir irá guardar no array um objeto com o uid do utilizador, o nome do utilizador e se está a true. O mesmo acontece no atributo checked se estiver a false, as validações que foram feitas anteriormente ocorrem em ambos os casos devido ao facto de existir uma função chamada removeUsers que tem dois propósitos. Primeiramente efetua uma filtração dos utilizadores que têm o checked a false, retornando assim apenas os que foram selecionados. Em segundo lugar, a função guarda na variável groups o groupName e os utilizadores que foram criados pelo utilizador autenticado, onde seguidamente guarda na base de dados o grupo criado, que contêm um uid do grupo (gerando um valor random), o groupName, o id dos utilizadores selecionados e a data em que o grupo foi criado. O

handleClose efetua uma limpeza dos campos necessários para no caso de o utilizador desejar criar mais grupos.

Finalizando o código do grupo, é retornado o html necessário com as variáveis criadas ao longo do código explicado anteriormente, de forma que seja possível a criação de um grupo, a seleção de um grupo e o envio de mensagens.

Na figura 35 é possível ver a página de grupos, quando o utilizador se encontra no modal que permite introduzir o nome do grupo e escolher os utilizadores do chat que pretende no grupo.

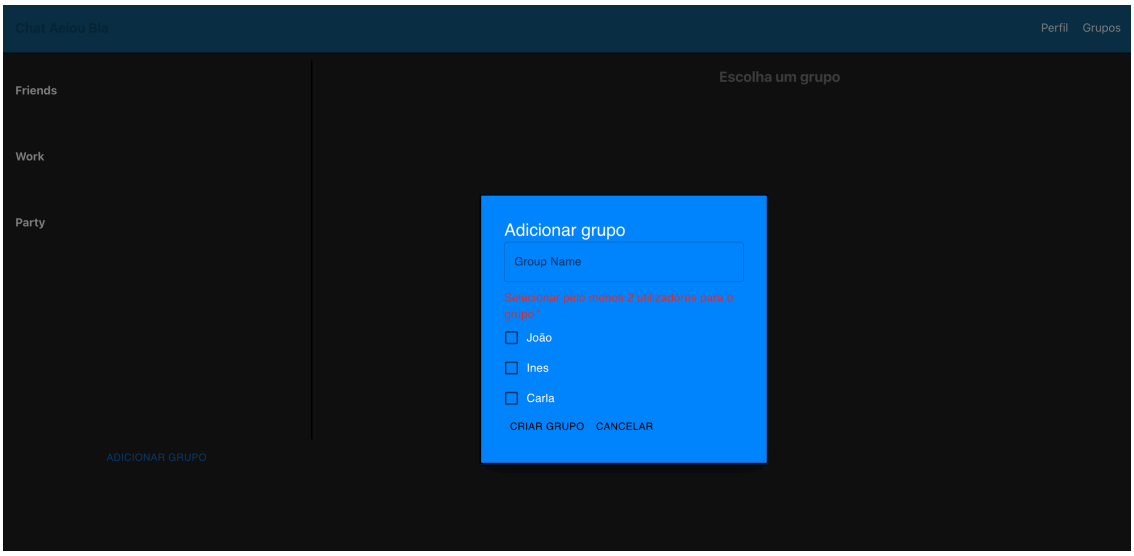


Figura 35 - Adicionar um grupo novo ao chat

6 Experimentação e avaliação

No presente capítulo é descrito, para o problema e objetivos definidos, quais foram as hipóteses e indicadores, metodologia de avaliação e resultados que representam o ponto inicial para a avaliação do projeto implementado.

6.1 Hipóteses e Indicadores

Para verificar se a solução apresentada poderá vir a ser uma mais-valia para a organização para a qual foi desenvolvida, é necessário definir métricas contra as quais será avaliada. Os indicadores definidos para a avaliação são:

- Performance/Desempenho – através deste indicador é possível perceber se o tempo de resposta do sistema implementado é aceitável;
- Modularidade – este indicador permite verificar qual é o nível de flexibilidade para a alteração de funcionalidades;
- Funcionalidade – pretende-se perceber se as funcionalidades do sistema implementado vão de encontro aos objetivos definidos inicialmente e às expetativas dos utilizadores;
- Usabilidade – através deste indicador pretende-se verificar qual é a facilidade que o utilizador apresenta ao utilizar o sistema desenvolvido.

A hipótese que se pretende testar é : (i) perceber se o grau de satisfação dos utilizadores é o ideal, face à aplicação desenvolvida.

6.2 Metodologia de Avaliação

Relativamente à avaliação realizada, foi necessário recorrer a um inquérito de satisfação (consultar Anexo A), que se destina aos utilizadores futuros da aplicação, mas que, neste caso,

foram respondidos por pessoas da organização que contêm mais experiência em websites e chats. Esta metodologia foi escolhida devido ao facto de ser um projeto interno/externo, mas apontado para todo o tipo de pessoas, tornando assim a opinião e o grau de satisfação dos possíveis utilizadores futuros muito relevante no processo de avaliação da solução desenvolvida.

O inquérito elaborado encontra-se dividido em 4 indicadores (o indicador de modularidade aparece vazio devido a não ser relacionado com o utilizador, mas mais com o código, logo não foram encontradas perguntas para este indicador), apresentados no subcapítulo 6.1, e é constituído por 13 perguntas. Na tabela 16 encontram-se descritas as perguntas do inquérito de avaliação.

Tabela 16 - Perguntas do inquérito de avaliação

INDICADOR	PERGUNTA
Usabilidade	1 – A aplicação é simples?
	2 – A aplicação é atrativa?
	3 – A aplicação é intuitiva?
	4 – As cores da aplicação são adequadas?
	5 – Consegue-se adaptar facilmente à aplicação desenvolvida?
Funcionalidade	6 – Consegue criar uma conta?
	7 – As funcionalidades do chat vão de encontro às suas necessidades?
	8 – Consegue enviar uma mensagem?
	9 – O chat é de fácil utilização?
	10 – Consegue criar um grupo?
	11 – As funcionalidades de criar um grupo vão de encontro às suas necessidades?
Performance/Desempenho	12 – O desempenho da aplicação é aceitável?
	13 – Consegue receber e enviar mensagens rapidamente?
Modularidade	----------

Para o utilizador responder a cada questão do inquérito, foi utilizada uma escala de avaliação constituída por cinco níveis, representada na tabela 17.

Tabela 17 - Escala de classificação utilizada no inquérito

ESCALA	DESCRIÇÃO
1	Discordo totalmente
2	Discordo
3	Nem concordo, nem discordo
4	Concordo
5	Concordo totalmente

6.3 Resultados do Inquérito

Neste subcapítulo é demonstrada a análise de resultados, que tem um papel fundamental na avaliação da solução desenvolvida. Os dados obtidos do inquérito permitem determinar se a aplicação desenvolvida vai de encontro a certos objetivos definidos inicialmente e para validar a hipótese definida anteriormente.

A partir do inquérito realizado foram obtidas 10 respostas e os resultados encontram-se na tabela 18.

Tabela 18 - Resultados do inquérito de avaliação

PERGUNTA	CLASSIFICAÇÃO				
	1	***2***	***3***	***4***	***5***
1	0%	0%	0%	40%	60%
2	0%	0%	0%	70%	30%
3	0%	0%	0%	50%	50%
4	0%	0%	20%	50%	30%
5	0%	0%	0%	40%	60%
6	0%	0%	0%	80%	20%
7	0%	0%	0%	50%	50%
8	0%	0%	0%	30%	70%

9	0%	0%	0%	40%	60%
10	0%	0%	50%	50%	0%
11	0%	0%	60%	40%	0%
12	0%	0%	0%	80%	20%
13	0%	0%	0%	70%	30%

Analisando os resultados obtidos e tendo em conta a associação entre respostas aos indicadores definidos inicialmente é possível verificar:

- **Usabilidade -** De uma forma geral, os resultados obtidos a este indicador são bastante positivos. Grande parte dos futuros utilizadores, consideram que a aplicação é bastante simples, atrativa e intuitiva. Para além disso, consideram que as cores escolhidas na aplicação são adequadas, apesar que 20% indicou que não discorda nem concorda, mas no geral concordaram que se adaptavam facilmente à aplicação.
- **Funcionalidade –** Relativamente às respostas relativas às funcionalidades da aplicação também foram positivas. Os possíveis utilizadores concordam que efetivamente as funcionalidades implementadas vão de acordo com as suas expectativas, uma vez que as classificações às perguntas 6 a 9 variam entre o nível 4 e 5. No entanto nas perguntas 10 e 11, 50% e 60% respetivamente, responderam com nível 3. Verificam-se estes resultados nestas perguntas acerca dos grupos possivelmente devido aos utilizadores notarem algumas inconsistências na criação de grupos e no envio de mensagens. Contudo, de um modo geral, conclui-se que as funcionalidades do sistema têm uma avaliação positiva.
- **Performance/Desempenho –** No que diz respeito ao indicador de performance/desempenho, apenas foram colocadas 2 perguntas. Após uma análise das classificações obtidas às perguntas 12 e 13, podemos verificar que 80% respondeu com o nível 4 e 20% respondeu com o nível 5 à pergunta 12 e que 70% respondeu com o nível 4 e 30% respondeu com o nível 5 à pergunta 13. Desta forma, podemos concluir que os futuros utilizadores consideram a performance/desempenho da aplicação bastante aceitável.
- **Modularidade –** Este indicador tem como objetivo perceber se o sistema implementado tem o nível de flexibilidade para alteração de funcionalidades ideal. Não foi possível receber nenhuma resposta acerca deste indicador devido ao facto que queremos avaliar como é que os utilizadores futuros se sentem ao utilizar a aplicação, o que este indicador necessita é de perguntas mais relacionadas com o código e respondidas por developers (com experiência). Posto isto, não foi feita nenhuma pergunta no inquérito para o utilizador futuro responder.

6.4 Análise de Resultados

Tendo em conta as respostas obtidas através do inquérito de avaliação, pode-se afirmar que os resultados obtidos foram positivos. Mesmo que a amostra de possíveis utilizadores, que responderam ao inquérito, tenha sido reduzida pode-se concluir que estes apresentaram um grau de satisfação elevado face à aplicação apresentada.

Os resultados obtidos no inquérito serviram também para validar a hipótese testada. Posto isto, é demonstrado que os futuros utilizadores (neste caso pessoas da organização) consideram que a aplicação desenvolvida possui potencial para ser utilizado e substituir o antigo chat. É possível indicar que a hipótese é ideal devido ao facto de as funcionalidades implementadas estarem de acordo com os objetivos definidos inicialmente e devido ao facto que os utilizadores consideraram que a aplicação era bastante simples, atrativa e intuitiva, o que contribui para que a aplicação, depois de feitas as correções necessárias, pode substituir o antigo chat.

Seria interessante, numa fase mais avançada da aplicação, devido ao número reduzido da amostra de possíveis futuros utilizadores, voltar a repetir um novo inquérito de avaliação com o objetivo de perceber se, utilizando uma amostra maior, os resultados iriam variar.

7 Conclusão

Este último capítulo da dissertação está encarregue de avaliar o balanço do projeto desenvolvido. Aqui são apresentados os objetivos alcançados. Para além disso, uma explicação acerca das limitações do trabalho desenvolvido e possíveis trabalhos que possam vir a ser realizados com este projeto. Por fim, é feita uma apreciação final do trabalho desenvolvido.

7.1 Objetivos Alcançados

Inicialmente foi realizado um estudo bibliográfico do que era aplicações de chat e que tipo de abordagens eram possíveis para a criação do chat. Elaborou-se uma pesquisa tecnológica das frameworks que poderiam ser utilizadas para o desenvolvimento, onde foram listadas, através de uma comparação, os pontos fortes e fracos das frameworks encontradas de maneira a ser possível encontrar a mais indicada. A explicação acerca de como seria utilizado o Firebase também foi demonstrada. Depois de idealizada a framework para o desenvolvimento, recorrendo aos diagramas UML, é que foi possível começar a implementação da solução descrita neste documento.

Assim, foi possível atingir o objetivo principal determinante do presente trabalho que era a criação de um novo chat para a organização, onde é possível ao utilizador interagir com outros utilizadores, tirando o melhor partido da aplicação e utilizando a interface web criada, e que a integração de novas funcionalidades não seja complexa, mas mais simplificada. Para além disso, foi necessário criar uma interface web simples e intuitiva para que fosse utilizada pelos utilizadores da aplicação, onde foi comprovado através de um inquérito de satisfação, que permitiu concluir que a maior parte dos inquiridos se mostraram satisfeitos com a aplicação/interface desenvolvida, considerando-a de fácil utilização.

Neste inquérito, onde foi realizado um questionário de satisfação aos possíveis utilizadores do protótipo desenvolvido, foi possível avaliar o sistema desenvolvido segundo indicadores de

performance/desempenho, modularidade, funcionalidades e usabilidade. Após a realização do questionário, o feedback obtido relativamente aos indicadores, foi positivo.

Na tabela 19 são demonstradas as relações entre os objetivos definidos inicialmente e os respetivos graus de cumprimento.

Tabela 19 - Relações entre objetivos definidos e grau de cumprimento

OBJETIVO	GRAU DE CUMPRIMENTO
LEVANTAMENTO DA OFERTA DE FRAMEWORKS PARA APLICAÇÃO CHAT	Atingido
IDENTIFICAÇÃO E INSTALAÇÃO DE FRAMEWORKS CHAT	Atingido
ESCOLHA E CONFIGURAÇÃO DE OPÇÕES FUNCIONAIS NA APLICAÇÃO	Atingido
DESENVOLVIMENTO DE NOVAS FUNCIONALIDADES SE NECESSÁRIO (OPCIONAL)	Não atingido
CRIAÇÃO DE INTERFACE WEB PARA A APLICAÇÃO CHAT	Atingido
ESTUDO FUNCIONAL DO WEBCHAT BLA.AEIOU.PT	Parcialmente atingido
MIGRAÇÃO DE UTILIZADORES BLA.AEIOU.PT PARA A APP	Não atingido
INTEGRAÇÃO DE FUNCIONALIDADES E PROPRIEDADES DO WEBSITE BLA.AEIOU.PT NA APP	Parcialmente atingido

7.2 Limitações e Trabalho Futuro

No presente subcapítulo são apresentadas as limitações ao projeto desenvolvido, bem como futuras implementações na aplicação desenvolvida.

A organização pretende utilizar a aplicação desenvolvida para atualizar o chat antigo para ferramentas mais modernas e atualizadas, como no caso da utilização de Firebase e React, e para tornar a integração de funcionalidades mais simples. Deste modo, é necessário que o chat seja de fácil utilização para o utilizador final, mas também para o programador que terá de acrescentar funcionalidades futuramente. Para isso, a aplicação foi desenvolvida de uma forma que o programador terá de conhecer Firebase e React para que seja mais eficaz na introdução de novas funcionalidades e atualizações que sejam necessárias para a aplicação.

Para tornar a aplicação mais completa, seriam interessantes o estudo funcional, a migração e a total integração das funcionalidades que já existiam no chat antigo, ficarem implementados de forma que os utilizadores presentes no chat antigo estejam familiarizados com o chat, devido a ter funcionalidades e caraterísticas do chat, mas para que fosse possível chamar utilizadores novos a descobrir o novo chat.

Outro requisito opcional, que é necessário notar devido à sua importância futura para a aplicação seria colocar a aplicação online e não apenas na máquina de quem programou a aplicação. Devido a aplicação ser para utilização por todas as pessoas que desejam utilizar a aplicação de chat, era importante colocar a aplicação online para esse feito. Este requisito é opcional devido à aplicação ser de elevada importância para a organização, o que implica a que a aplicação esteja recetiva a centenas ou milhares de utilizadores. Para este projeto era importante inicialmente ter várias funcionalidades a funcionar para futuramente ser possível implementar ainda mais com o intuito de colocar a aplicação com o máximo de funcionalidades funcionais e apelativas para os utilizadores finais.

Para além disso, por ser um dos objetivos opcionais para a conceção do chat, seria uma mais-valia a descoberta de funcionalidades chamativas para a implementação no chat, de forma que os antigos utilizadores, mas também os novos utilizadores, possam aproveitar ao máximo o tempo que usam a aplicação, mas que fosse um desafio interessante para a aprendizagem do programador que implementasse as mesmas.

Contudo nos grupos, com vista a ser trabalho futuro, devido a não estar totalmente funcional, seria importante corrigir o envio de mensagens. O utilizador consegue enviar mensagens para o grupo, mas os utilizadores não conseguem ver as mensagens como se fossem enviadas por outro utilizador, apenas como se fossem enviadas pelo utilizador que se encontra autenticado. Para além disso, quando o utilizador tenta criar um grupo, a página deixa de renderizar e só fica corrigido quando se atualiza a mesma, este erro foi detetado, mas não foi corrigido totalmente, devido ao tipo de dados não ser um array. Mesmo assim, o utilizador final consegue criar grupos mesmo com esse erro.

7.3 Apreciação Final

O autor teve a oportunidade de realizar um estágio profissional em ambiente empresarial que permitiu adquirir novas competências para projetos pessoais e profissionais futuros, o que permitiu o seu desenvolvimento a nível tecnológico. Para além do desafio tecnológico, por ter sido feito em ambiente empresarial, o estágio profissional desafiou o autor a um nível pessoal, sendo capaz de desenvolver algumas capacidades de espírito de decisão, aprendizagem e preparação.

Devido a ser possível interagir e ter o acompanhamento com profissionais da área de software, também foi um fator importante que contribuiu para o crescimento pessoal e profissional do autor, permitindo o alargamento e aprofundamento dos seus conhecimentos.

Tendo em conta que o projeto foi bem sucedido, o autor teve a oportunidade de integrar na sua primeira experiência profissional na Sooma.

Referências

[1] Thakur, A., & Dhiman, K. (2021). Chat Room Using HTML, PHP, CSS, JS, AJAX. arXiv preprint arXiv:2106.14704.

[2] Al-Fedaghi, S. (2011). Developing web applications. International journal of software engineering and its applications, 5(2), 57-68.

[3] Sireteanu, N. A., & Homocianu, D. (2021). Front-End Frameworks for Development of Spa and Mpa Web Applications. Available at SSRN 3987838.

[4] Hume, D. (2017). Progressive web apps. Simon and Schuster.

[5] Majchrzak, T. A., Biørn-Hansen, A., & Grønli, T. M. (2018). Progressive web apps: the definite approach to cross-platform development?.

[6] A. Pano, D. Graziotin, and P. Abrahamsson, "Factors and actors leading to the adoption of a JavaScript framework," Empir. Softw. Eng., vol. 23, no. 6, pp. 3503–3534, Dec. 2018, doi: 10.1007/s10664-018-9613-x.

[7] S. Aggarwal, "Modern Web-Development using ReactJS," vol. 5, no. 1, p. 5, 2018.

[8] A. Fedosejev, React. Olton Birmingham: Packt Publishing Ltd, 2015.

[9] Voutilainen, J. P., Mikkonen, T., & Systä, K. (2016, June). Synchronizing application state using virtual DOM trees. In International Conference on Web Engineering (pp. 142-154). Springer, Cham.

[10] Linh, T. (2019). Building websites with Laravel and VueJS.

[11] Jadhav, M. A., Sawant, B. R., & Deshmukh, A. (2015). Single page application using angularjs. International Journal of Computer Science and Information Technologies, 6(3), 2876-2879.

[12] Kaluža, Marin & Vukelic, Bernard. (2018). Comparison of front-end frameworks for web applications development. Zbornik Veleučilišta u Rijeci. 6. 261-282. 10.31784/zvr.6.1.19.

[13] John, S. (2020). Chat App With React JS And Firebase.

[14] Trott, P. (2008). Innovation management and new product development. Pearson education.

[15] Belliveau, P., Griffin, A., & Somermeyer, S. (Eds.). (2004). *The PDMA toolbook 1 for new product development*. John Wiley & Sons.

[16] Benzaghta, M. A., Elwalda, A., Mousa, M. M., Erkan, I., & Rahman, M. (2021). SWOT analysis applications: An integrative literature review. Journal of Global Business Insights, 6(1), 55-73.

[17] Osterwalder, A., Pigneur, Y., Bernarda, G., & Smith, A. (2015). Value proposition design: How to create products and services customers want (Vol. 2). John Wiley & Sons.

[18] Saaty, T. L. (1988). What is the analytic hierarchy process?. In Mathematical models for decision support (pp. 109-121). Springer, Berlin, Heidelberg.

[19] BOUSETTA, B., Omar, E.B. and GADI, T., 2013. Generating operations specification from domain class diagram using transition state diagram. International Journal of Computer and Information Technology (IJCIT) January, 2(1), pp.p 29-36.

[20] Hull, E., Jackson, K. and Dick, J., 2005. Requirements engineering in the solution domain (pp. 109-129). Springer London.

[21] Jamwal, D. (2010). Analysis of software quality models for organizations. International Journal of Latest Trends in Computing, 1(2), 19-23.

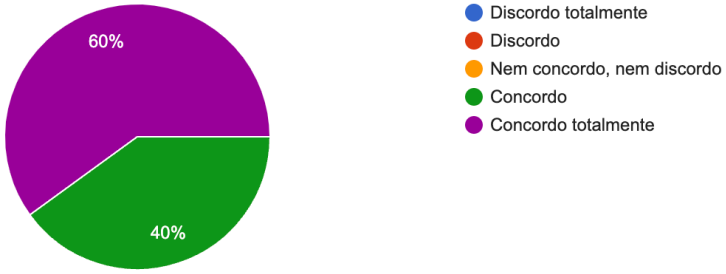
[22] Janošcová, R. (2012). Evaluation of software quality. IMEA 2012

Anexo A – Perguntas do Inquérito para Avaliação

A aplicação é simples?

10 respostas

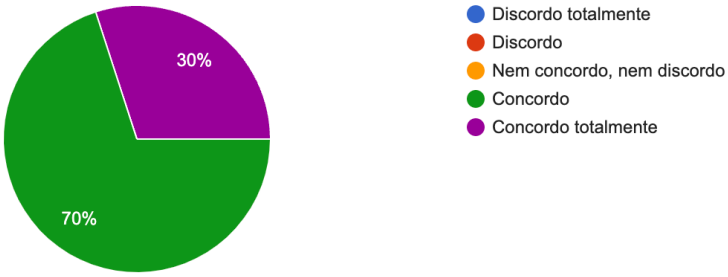
Copiar



A aplicação é atrativa?

10 respostas

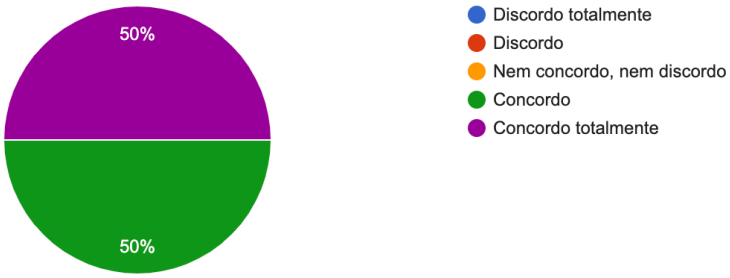
Copiar



A aplicação é intuitiva?

10 respostas

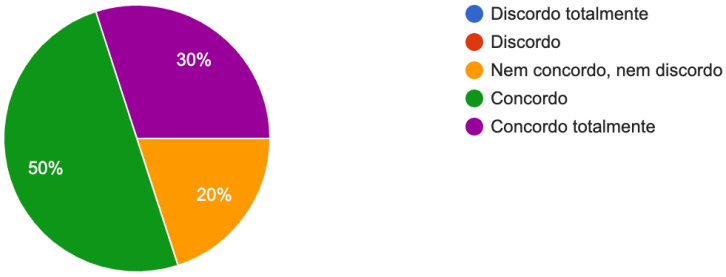
Copiar



As cores da aplicação são adequadas?

Copiar

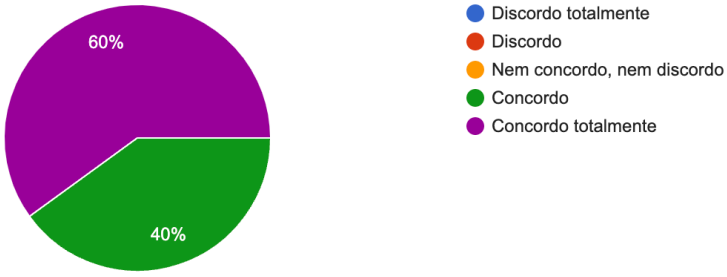
10 respostas



Consegue-se adaptar facilmente à aplicação desenvolvida?

Copiar

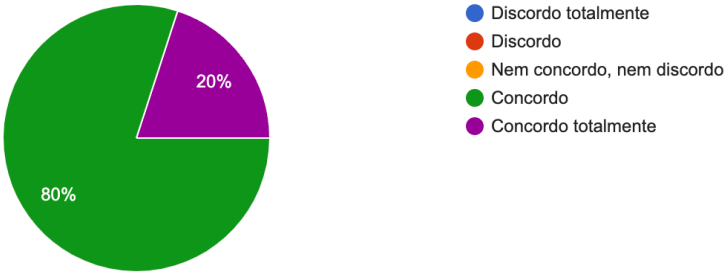
10 respostas



Consegue criar uma conta?

Copiar

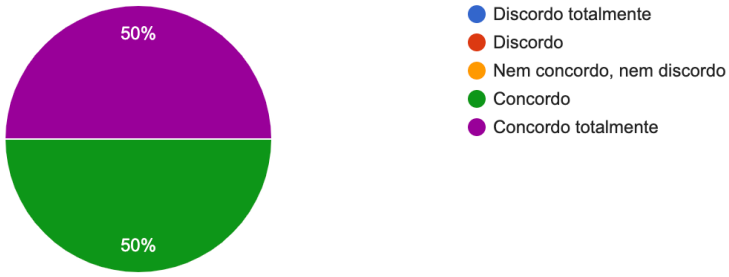
10 respostas



As funcionalidades do chat vão de encontro às suas necessidades?

Copiar

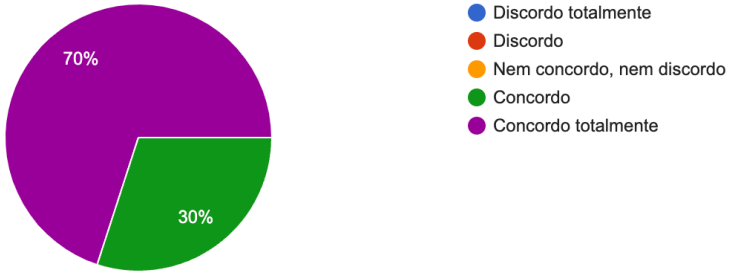
10 respostas



Consegue enviar uma mensagem?

Copiar

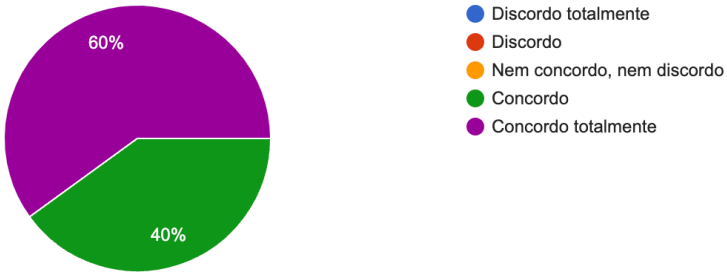
10 respostas



O chat é de fácil utilização?

Copiar

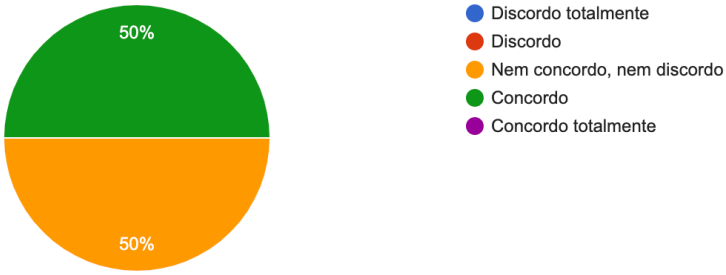
10 respostas



Consegue criar um grupo?

Copiar

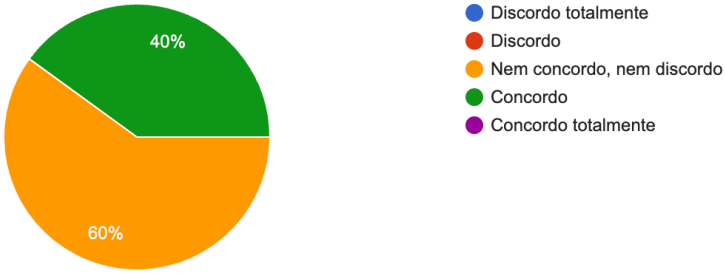
10 respostas



As funcionalidades de criar um grupo vão de encontro às suas necessidades?

Copiar

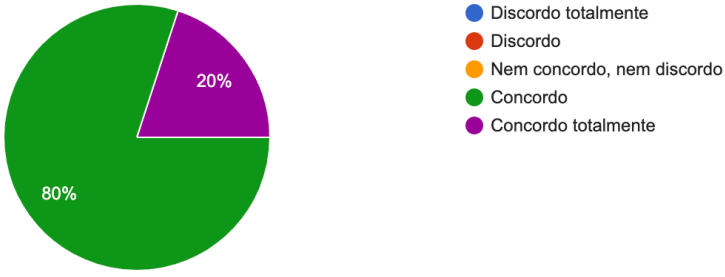
10 respostas



O desempenho da aplicação é aceitável?

Copiar

10 respostas



Consegue receber e enviar mensagens rapidamente?

Copiar

10 respostas