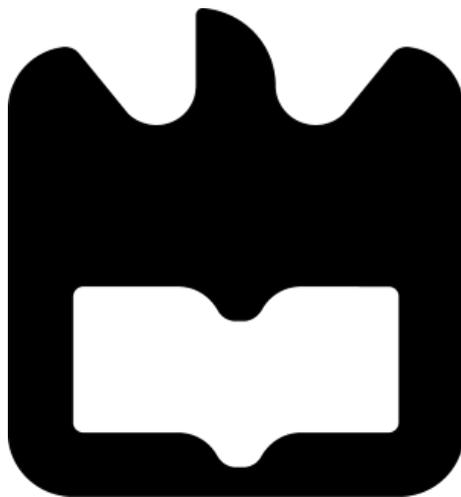




**Ana Sofia
Dos Santos Góis**

**Aplicação móvel para estender a plataforma de
negócios centroproduto**

**Mobile channel to extend centroproduto's business
platform**





**Ana Sofia
Dos Santos Góis**

**Aplicação móvel para estender a plataforma de
negócios centroproduto**

**Mobile channel to extend centroproduto's business
platform**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Ilídio Fernando de Castro Oliveira, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

Dedico este trabalho aos meus pais e ao meu irmão porque sempre me apoiaram e deram força para avançar, à minha avó por sempre me motivar a chegar a onde estou hoje, à minha tia Filomena que apesar da distância está sempre comigo e aos meus tios Catarina, Cândida e José Augusto pelo grande apoio que me têm dado durante este longo percurso.

o júri

Presidente

Professor Doutor Joaquim João Estrela Ribeiro Silvestre Madeira
professor auxiliar da Universidade de Aveiro

Arguente Principal

Professor Doutor Telmo Eduardo Miranda Castelão da Silva
professor auxiliar da Universidade de Aveiro

Orientador

Professor Doutor Ilídio Fernando de Castro Oliveira
professor auxiliar da Universidade de Aveiro

agradecimentos

Quero agradecer a orientação do Prof. Ilídio Oliveira, por me ter ajudado a avançar com este trabalho de dissertação.

Também quero agradecer ao Eng. Pedro Rêgo, por me ter dado a oportunidade de desenvolver este projeto, em contexto de empresa.

Muito mais, agradeço aos meus colegas do centroproduto que me ajudaram durante o desenvolvimento da aplicação.

Por último e mais importante, agradeço aos meus pais por me apoiarem e por me terem dado a oportunidade de estar cá hoje a tirar este curso.

palavras-chave

Aplicação móvel, Android, centroproduto, *smartphone*, sistema de mensagens, *chat*, diretório de empresas.

resumo

A empresa centroproduto opera um Portal web utilizado como um diretório de empresas e fórum de negócios, que permite às empresas em geral anunciar os seus produtos e serviços ou procurar potenciais fornecedores. Os fluxos de contacto entre empresas são executados no portal e podem ser complementados com a troca de mensagens, por *email*.

Este trabalho visou a conceção e desenvolvimento de uma aplicação móvel para complementar os canais de contacto do centroproduto, permitindo uma utilização mais pervasiva. A obtenção de requisitos resultou de um contexto de imersão em empresa, considerando as necessidades de uma plataforma de negócios concreta.

A solução proposta baseia-se na utilização de uma aplicação móvel nativa para a plataforma Android, com integração de serviços de distribuição de mensagens (*push notifications*). Também o ecossistema de soluções já existente foi considerado no desenho da arquitetura, uma vez que a nova aplicação comunica com o *back-end* através de uma camada de serviços.

A aplicação resultante está integrada no portfólio da empresa, e virá a estar, num futuro próximo, disponível no portal do centroproduto e na loja virtual Google Play. Esta aplicação permitirá às empresas utilizadoras enviar mensagens, seguir conversas, receber notificações diretamente no telemóvel, mantendo uma forte coesão com o portal, tanto quanto à experiência de utilização, como em relação à continuidade dos processos de trabalho.

keywords

Mobile application, Android, centroproduto, smartphone, messaging system, chat, business directory.

abstract

centroproduto is a company operating a web portal used as a business directory, allowing other companies to announce or look for products and services. The supported workflows were performed on a browser and could be complemented with email exchange. Mobile devices offer a new channel for internet business to engage costumers and, in particular, push messages in an opportunistic way, for example, to deliver chat-like interactions or new offer announcements. In this work, the requirements and the implementation of a mobile application are outlined in order to provide centroproduto with a new communication channel that supports their portal. The requirements elicitation was performed using the real business context and the system architecture constrained by the existing ecosystem in which this work was integrated. Several smartphones and scenarios were included in the testing and quality assurance activities.

The resulting application is integrated in the company's code base portfolio and will be available in the near future through centroproduto's portal and in Google Play store.

Content

List of Figures	17
List of Tables	19
1. Introduction	1
1.1. Context and motivation	1
1.2. Objectives.....	2
2. Related work and state of the art	3
2.1. Similar Platforms and services	3
2.1.1. Facebook	3
2.1.2. LinkedIn	5
2.1.3. Alibaba.....	6
2.1.4. Summary	8
2.2. Development Technologies.....	9
2.2.1. Android platform.....	9
2.2.2. Push Notification Service	10
2.2.3. Image Loaders for Android.....	16
3. System Requirements	19
3.1. Business context.....	19
3.2. Supported scenarios.....	19
3.2.1. User Authentication	21
3.2.2. Messaging and Notification related Use Cases	23
3.3. Cross-cutting requirements.....	26
3.4. Planned application outlined	26
3.5. Summary	27
4. System Architecture and implementation	29
4.1. System Architecture	29
4.2. Android Implementation.....	31
4.2.1. Targeted versions.....	31
4.2.2. Layouts	32
4.2.3. Amazon SNS	33
4.2.4. Chat Head Service	34
4.2.5. Activities	35
4.2.6. Library for image optimization and access.....	38
5. User Interactions Design	41

5.1.	Android prototype.....	41
5.1.1.	Colors and Icons	41
5.1.2.	Launcher Icon and Splash Screen.....	42
5.1.3.	Authentication.....	42
5.1.4.	Conversation List.....	44
5.1.5.	Drawer menu.....	50
5.1.6.	Configuration of Settings.....	51
5.1.7.	Chat Head	53
5.1.8.	Conversation	53
5.1.9.	Feedback on internet connection	58
5.2.	Collaboration with the Design team	61
6.	Validation.....	63
6.1.	Unit testing with Espresso.....	63
6.2.	centroproduto's Tests	64
6.3.	Pilot Users	65
7.	Conclusion and Future Work.....	67
7.1.	Achievements.....	67
7.2.	Lessons Learned	67
7.3.	Future Work	68
References	69

List of Figures

Figure 1 - Possible System Architecture.....	9
Figure 2 - Google Cloud Messaging connection	11
Figure 3 - MQTT protocol.....	12
Figure 4 - Amazon SNS, Push Notification Service	15
Figure 5 - Use case diagram	20
Figure 6 - Activity diagram - Sign In.....	22
Figure 7 - List conversations	24
Figure 8 - Application's menu	27
Figure 9 - Overall System Architecture	29
Figure 10 - Class diagram	36
Figure 11 - Launcher button - Main Screen	42
Figure 12 - Launcher button - List of Applications	42
Figure 13 - Splash Screen.....	46
Figure 14 - Sign In - Required Field (Password)	46
Figure 15 - Sign In - Verify email and password	46
Figure 16 - Recover Password – Success.....	46
Figure 17 - Sign in	47
Figure 18 - Loading Bar.....	47
Figure 19 - List of Conversations Screen	47
Figure 20 - Conversation - One user	48
Figure 21 - Conversation - Two users.....	49
Figure 22 - Conversation - Three users	49
Figure 23 - Conversation - Four users	49
Figure 24 - Conversation - More than Four users	49
Figure 25 - Refresh.....	50
Figure 26 - Menu	52
Figure 27 - Default logo (Left) and default photo (Right).....	52
Figure 28 - Settings.....	52
Figure 29 - Notification (Left) and Chat Head (Right).....	53
Figure 30 - Informative Message – Short Subject.....	55
Figure 31 - Subject Extended	55
Figure 32 - Message List	56
Figure 33 - Message Received	56
Figure 34 - Message Sent	56
Figure 35 - Change Company.....	57
Figure 36 - Menu - Option	57
Figure 37 - User's list	57
Figure 38 - Sign In - Connection Lost	58
Figure 39 - List of Conversations - No connection.....	58
Figure 40 - List of Conversations - Connection Lost.....	59
Figure 41 - Conversation - No connection.....	59
Figure 42 - Conversation - Connection lost.....	59
Figure 43 - List of users - Connection lost.....	59
Figure 44 - Settings - Connection lost - Left: No company, Right: List of companies.....	60
Figure 45 - Test results – Unit test.....	64

List of Tables

<i>Table 1 - Applicable features</i>	8
<i>Table 2 - Push Notification Services</i>	15
<i>Table 3 - Image Loaders</i>	18
<i>Table 4 - Cross-cutting requirements</i>	26
<i>Table 5 - Requirements importance</i>	28
<i>Table 6 - Exert of the Gradle configuration</i>	32
<i>Table 7 - Exert of Density metrics code</i>	32
<i>Table 8 - Density Metrics - Value</i>	33
<i>Table 9 - strings.xml – Example: Sender ID</i>	33
<i>Table 10 - Device Registration to GCM</i>	34
<i>Table 11 - ShadowBuilder Code</i>	35
<i>Table 12 - Example ViewHolder Class</i>	37
<i>Table 13 - Example Row Inflation - getView Function</i>	37
<i>Table 14 - Picasso and crop transformations</i>	38
<i>Table 15 - Exert of Espresso test</i>	63

1. Introduction

1.1. Context and motivation

Modern digitalization of business includes a profound transformation of companies, from digitalization of the productive process to new business propositions, based on digital products and services [1].

Technology plays an important role when clients want to find products and services (e.g. web stores) and when business are willing to find partners (e.g. on-line marketplace).

The match-making of business/companies can benefit from specialized platforms, such as centroproduto [2] which is a company that provides Business-to-Business (B2B) services to companies worldwide through a centralized web portal. This portal, allows these companies to register and take advantage of the functionalities of the platform, such as offer products and/or services, post new offers, among others. Also, it offers a way of communication among these companies based in a messaging system, which means that they can send messages to each other in order to request budgets or to make an order for one or more products, or even services. Through the portal, the manager of each account can create the company's own business network. This portal can be viewed in 7 different languages and gives its users the possibility of searching products by category, by image and by company. Furthermore, centroproduto also has a mobile web portal [3] where the users can keep up with their business even when they are away from the computer.

However, there are different ways in which technology is used to provide the clients and businesses with advantages. Two of them could be the use of websites and mobile applications, being mobile applications perhaps, the most important in the present day. This because, unlike websites they provide many advantages such as: keeping connection with family and friends at any time and place, being allowed to keep their business going without actually being at work or sitting in front of a computer, setting reminders, and much more. These advantages are so important to many people that they end up relying on mobile applications. In fact, they rely so much in these applications that somehow the necessity to use a computer or a laptop regularly is not as fundamental as it used to be. An example of a platform where the number of mobile users active is higher than the number of mobile and desktop users is Facebook since the third quarter of 2015 [4]. Another example is Google that in May, 2015 revealed that more Google Searches take place on mobile devices rather than computers in 10 different countries [5].

Moreover, mobile devices can be taken anywhere; they can be carried in a pocket or held with a single hand which cannot be done with a computer or a laptop.

Having said that, centroproduto is looking for the possibility of investing in a mobile application that adds value to the system, i.e. a mobile application that uses features which are different from any other in centroproduto's system and most importantly that provides a better user experience to potential users.

1.2. Objectives

The main goal of this dissertation is to develop and provide a mobile application that works as a messaging system in which centroproduto's users can exchange information. An application the users can rely on and that will ease their life and how they do business.

The work focuses mainly on the following objectives:

1. Assess the advantages and disadvantages of developing a mobile application functionality for centroproduto that bring a new opportunity to improve their business.
2. Study how related mobile applications work.
3. Develop a mobile application to facilitate business interaction.
4. Integrate the new mobile application functionality in the company's routine workflows.

2. Related work and state of the art

2.1. Similar Platforms and services

Part of the present work was to identify and propose features that can be learned from existing products and for a mobile app that would bring value for the B2B platform operated by centroproduto.

The main goal of analyzing these other platforms is to identify the features they use and which of those can be applied to the development of a mobile application for centroproduto. Three solutions were analyzed in total, namely: Facebook [6], LinkedIn [7] and Alibaba [8]. From the most well-known platforms, these are the most similar to centroproduto's platform since all of them work as networks (social and/or professional). Each one of these will be described, as well as their pertinent characteristics.

For the three cases, one will present a brief comparison between what is available in the mobile application and web application, and an overview of features potential that are relevant to include in this work.

2.1.1. Facebook

Today, Facebook is the biggest social network with more than 1,600 million users by the first quarter of 2016 [9]. This platform, intended for any user who wants to keep in contact with old and new friends and/or to promote their business, offers a huge amount of functionalities, such as sharing information, photos and videos with friends or publicly. The user can also play games, create pages and groups. Moreover, it offers a system where the users can create adverts to publicize their business.

Aside from these functionalities, Facebook offers different means so the user can take advantage of them. These means are Facebook's website and mobile application. Since Facebook's website is huge, it is expected and certainly true that the mobile application does not allow the user to do all the things he/she can do on the website. The main reason why this happens is because one single application is not capable of supporting so many varied functionalities (e.g. creating and managing new pages). Consequently, Facebook started splitting the main mobile application into several mini applications, each of these focusing on a specific functionality. Currently, Facebook has more than 10 applications [10] which are:

- Facebook which is Facebook's main application.
- Facebook Lite which is a lighter version of the main application.
- Facebook Messenger that works as a messaging system.
- Facebook at Work which is a system developed for companies, to allow communication among the employees during work.
- Work Chat which is a spinoff of Facebook at Work.
- Page manager to help the user managing his/her page(s).
- Facebook Groups to allow the users to manage their groups in one place.
- Ad's Manager which allows to create and manage ad's for small and mid-sized companies.
- Moments which allows managing and sharing photos with friends.
- Facebook, also has other extensions for Facebook Messenger, such as: Stickered for Messenger and Sound Clips for Messenger.

From all the above, Facebook Messenger [11] stands out. This is because the application is based on Facebook's website chat and it is a mandatory requirement of the main application to be able to chat with the user's Facebook friends. Messenger, as an application itself, offers the same characteristics of the chat on the Facebook website, as well as other characteristics/features, by using a different method of notification, specifically, the Chat Heads.

Besides Chat Heads, there are characteristics/features that Facebook uses in their mobile application and Messenger. These are:

- Facebook makes use of the user's friend list.
- Both main mobile application and Messenger make use of the phone's camera and/or the photo gallery, allowing the user to share photos by post or chat.
- The phone's GPS allows the user to share his/her location.
- Facebook uses the user's email and phone contacts to find people that might potentially turn out to be friends.
- Facebook will not work without an internet connection.
- Users can share voice notes through the chat.
- The language of the device is used to detect what the language of the application should be.
- The user gets notified every time he/she receives a message or every time someone posts on their profile, or even when someone tags them on photos and comments.

From all the features specified above, it can be said that the most important feature which can be used for the development of this mobile application is receiving a message through Chat Heads, because every time someone sends a message, the Chat Head will open on the front of the current screen in use to let the user know that a message was received and it will catch his/her attention. However, before considering using this feature it was necessary to find out whether this great asset could or not be used by any other rather than just Facebook. As result, many applications using Chat Heads where found in Google Play [12] meaning that the feature can be applied to other applications. Also, centroproduto's application would also require internet connection to be able to receive messages.

2.1.2. LinkedIn

LinkedIn is a professional network, used not only by people looking for new job opportunities, but also by recruiters looking for good and skilled employees. By the first quarter of 2016, LinkedIn was bought by Microsoft Corp for \$26.2 billion [13]. By that time, LinkedIn already had more than 400 million users [14] and the prospect is for that number to keep growing. Through LinkedIn, the users can upload their professional Curriculum Vitae (CV) and also indicate in their profile their past job experiences, acquired skills, written papers and achievements.

To allow the users to perform these actions, LinkedIn offers bot a website and a mobile application that when compared, it becomes clear that the website is more complete than the mobile application. However, this does not mean it is a poor application. The application itself offers enough capabilities and functionalities so the user can do exactly what is expected from this network, like looking and applying for a job, receive and send messages to and from their connections and recruiters, and also upload new information.

Unlike Facebook, LinkedIn does not have any other application for messaging. Instead, the messaging system is integrated in the main application. Regardless of this situation, LinkedIn has also decided to start splitting their applications, having today nine mobile applications [15], each one for a specific functionality. These are:

- LinkedIn which is LinkedIn's main application.
- LinkedIn Job Search that allows the users to search for job opportunities
- LinkedIn SlideShare which provides presentations, videos and infographics to the users.
- Lynda – Online Training provides courses to learn new skills that are on-demand.

- LinkedIn Lookup simplifies the way people can be found and contacted even when they are not connected on LinkedIn.
- LinkedIn Sales Navigator to keep up with the user's accounts and conexions.
- LinkedIn Recruiter to help recruiters finding people that may be potential hires.
- LinkedIn Pulse provides daily news about the professional world.
- LinkedIn Elevate used by companies to provide their employees to discover articles and insights about information they are interested in.

One of the approaches that LinkedIn also uses is that by providing a mobile application the user can be notified at any moment –if he/she has internet– about new messages or interactions with current and new connections. Moreover, it also has a list of characteristics/features that were used in their main application. Some of these are:

- They make use of the user's friend list.
- The application makes use of the phone's camera and/or the photo gallery, allowing the user to share photos by post or chat.
- The phone's GPS allows the user to share its location in its profile.
- It uses the user's email and phone contacts to find people that can be turn out to be his/her friends.
- It requires internet connection to work.
- The language of the device to detect what should be the language of the application.
- The user gets notified every time it receives a message or every time someone likes one of its posts or upload, or even when someone sends a connection request.

It should be noted that, this list is very similar to the one previously presented in the previous section (see section [2.1.1.](#)).

2.1.3. Alibaba

In the present-day, Alibaba is considered to be the biggest e-commerce business-to-business platform [16]. Its main goal is to connect many suppliers with buyers so they can do business [17]. Also, it provides a system that helps giving their targeted users the confidence they need when it comes to payment transactions by keeping the buyer's money safe until they receive their orders, and only after, the money will be handed over to the supplier. Alibaba has now several websites, among them: Taobao [18] which is a shopping website, Tmall [19] for online sales of branded

products and AliExpress [20] for retail consuming. By the first quarter of 2016, Alibaba had over 420 million online end-users [21] using their platform. Part of those end-users not only use Alibaba's website also their mobile application. However, Alibaba has considerably simplified their main mobile application, making it impossible for the users to do everything they need or want to do through the application. Notwithstanding, this does not change the fact that a user has the ability to do some specific things as: managing orders, messaging and make requests to suppliers or see and buy new products, i.e. the application has enough functionalities to allow their users to do business.

As Facebook and LinkedIn, Alibaba also has 2 other mobile applications but not exactly for the same reasons. As it was previously mentioned, they also have other platforms, among them they have 2 applications aimed for AliExpress which is a Business-2-Client (B2C) e-commerce [22] and another application which is an auxiliary application of the main application but targeted to suppliers only, and is named AliSuppliers [23].

Taking LinkedIn's example (see section [2.1.2.](#)), Alibaba also provides, not only in their main application but in all of them, the possibility of receiving notifications from answers to requests or messages sent by other users when they are doing business. In fact, Alibaba offers others features besides the possibility to receive notifications. These are:

- They make use of the user's connections list.
- The application makes use of the phone's camera and/or the photo gallery, when making requests.
- It requires internet connection to work.
- Their application uses the language of the device to detect what the language of the application should be.
- The user gets notified every time it receives a message or a request.

The similarities among this list and the previous ones for Facebook and LinkedIn are not by chance. All of them use nearly the same things to approach their targeted users, making it hard to tell or add a different characteristic to the ones added in the previous sections (see sections [2.1.1.](#) and [2.1.2.](#)) that could be used when developing centroproduto's mobile application.

2.1.4. Summary

From the previous lists (see sections [2.1.1.](#) and [2.1.2.](#)) there are features and/or components that can be used by the application of centroproduo (see Table 1). The high-level analysis was specifically requested by centroproduo as an input for the discussion of the effective requirements for this work.

Table 1 - Applicable features

Features	Facebook	LinkedIn	Alibaba	Interest centroproduo	Applied to centroproduo
User's friend list	✓	✓	✓		
Phone's camera	✓	✓	✓		
Photo gallery	✓	✓	✓		
User's location	✓	✓			
User's email	✓	✓			
Phone contacts	✓	✓			
Microphone	✓				
Internet connection	✓	✓	✓	✓	Would be used to keep the application active and be able to interact with the system.
Device's language	✓	✓	✓	✓	Would be used to install the application in any of the languages supported by centroproduo.
Notifications	✓	✓	✓	✓	Would be used to acknowledge the user about new information in the system.
Chat Head	✓			✓	Would be used to catch the attention of the user and engage him/her when new information is received.

2.2. Development Technologies

To develop an application that receives information from the back-end of centroproduto it is necessary to disclose to which platform the application is going to be developed and also what kind of service should be integrated in the application to receive that information. In Figure 1, it is possible to see a representation of the infrastructure already used by centroproduto, which is an Amazon Infrastructure and both the representation of the mobile application and the Push Notification Service that should be used to establish communication among them. In this case, the mobile application should communicate with the Back-end of centroproduto through a REST service [24], and the back-end should have integrated a Push Notification Service that will send new information to the mobile application.

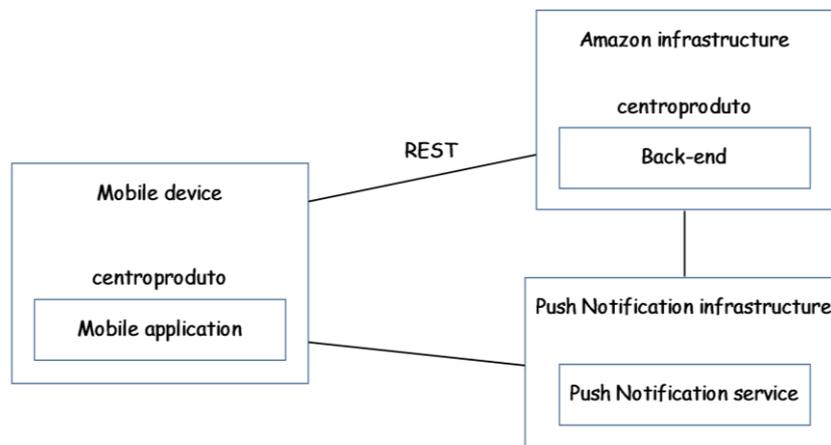


Figure 1 - Possible System Architecture

2.2.1. Android platform

Android OS, released in September 2008 [25], is an open source operating system [26]. This means that anyone can get the code and create customized versions of Android OS which is a great benefit for many device manufacturers, since they can install and use it when building their smartphones, being in the present-day one of the most used operating systems. Today, Android has more than 1.4 billion active users [27]. Furthermore, this operating system offers a store called Google Play [10] from which all the smartphone users can download many of the 2.000.000 existing applications [28]. Aside from being able to use other stores, such as: Amazon App Store for Android and Samsung Galaxy Apps.

Considering the users rather use Google Play, not only they can download applications from the store, but they can upload new applications starting by paying a fee of \$25 once in a lifetime [29]. Also, it provides a Development mode for alpha and beta testing, allowing developers to upload their applications and have them tested by other users for a period of time [30]. Moreover, at the release moment, the application will be online in few hours.

Developing an application for Android OS is quite complicated, not because of the programming languages but because of the quantity of Android versions. Today, Android has 23 APIs [31] that can be used to develop and support applications for those versions. However, this is not the only problem. As there are many manufacturers creating several different devices, is difficult to design a single interface for all of them and there is when the screen size problem comes in. In such case, Android Developer [32] offers a guideline on how to support multiple screen sizes and densities [33]. An example for this is that although the code is the same, the screen size of a tablet offers the possibility of showing more information in screen than a smartphone.

2.2.2. Push Notification Service

A push notification service [34] is a service that notifies the users about new information while they are not using the application. Today, there is a high demand for these services. Therefore, many developers offer services that work similarly but have a different background, which means that there are several of them to be considered when developing an application. Considering that for this application to work properly it will need to receive notifications from the Back-end, one had to study different services to decide which one would be more useful. These services will be explained subsequently.

2.2.2.1. *Google Cloud Messaging (GCM)*

Google Cloud Messaging [35] service as its name says is provided by Google [36] and like any other, it has advantages and disadvantages. When this service was first released it would only work for Android Push Notifications, but in its latest version, "GCM 3.0", Google included support for iOS and Google Chrome as well. GCM not only supports push notifications, or downstream messaging, as it also supports upstream messaging [37].

The main advantage of using a service as GCM is that it will not be necessary for the application to have an open connection with the application server, which would be time and power consuming.

Instead, the application server will detect when new information has arrived and then acknowledge GCM about it. Afterwards, GCM will take care of sending it [38].

As referred there are 3 components which communicate among them. These are the application server, the service (GCM) and the mobile phone; and the communication works in the following way (Figure 2):

- 1) The mobile phone registers to GCM.
- 2) GCM provides a “registrationID” to the mobile phone.
- 3) The mobile phone sends the “registrationID” to the application server.
- 4) If the application server, which is connected to GCM, detects the new information and notifies GCM about it.
- 5) GCM gets the information and sends it to the mobile phone.

Please note that the message (or information) will only reach the destination if the “registrationID” of the phone exists, if not the “registrationID” must be refreshed in order to start receiving messages again [39].

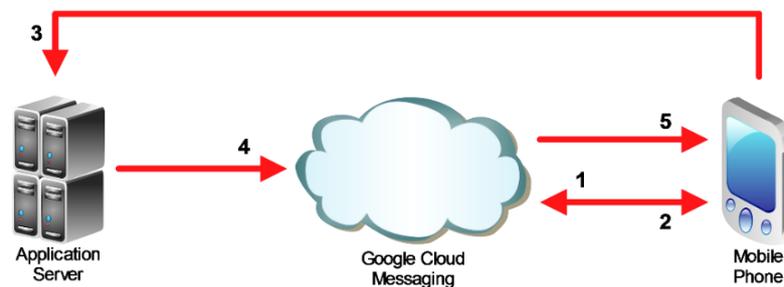


Figure 2 - Google Cloud Messaging connection

2.2.2.2. Message Queue Telemetry Transport (MQTT)

MQTT [40] is a protocol that operates in a publish-subscribe context and works asynchronously. Unlike GCM, this protocol uses a methodology where control packets have to be exchanged in order to work properly. These packets will be sent between a MQTT Server and a MQTT client in an orderly manner (Figure 3).

It is important to disclose that the main reason why this protocol was initially considered as an option was due to the fact that Facebook uses it in their own applications. In fact, Facebook’s developer Lucy Zhang expressed that this form of transportation was especially developed to avoid using more bandwidth than necessary and to save the power of a mobile phone’s battery while

transmitting data from a long distance [41]. Moreover, MQTT is extremely reliable, since it has three levels of Quality of Service (QoS) [42]: At most once, at least once and exactly once. This is important because it does not matter how unreliable is the transmission of the information, MQTT will make sure that the information gets to its destination [43]. Also, HiveMQ, which is an enterprise MQTT broker, explains that MQTT's great advantage is to support QoS because thanks to those 3 levels it can ease the exchange of information among the participants in the connection, even when the network is unreliable.

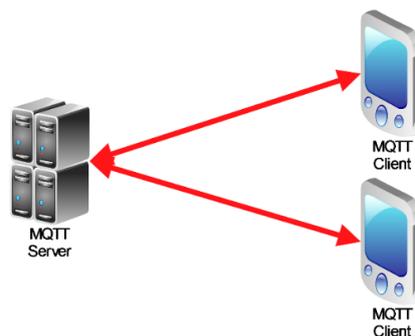


Figure 3 - MQTT protocol

The way how the communication between the server and the client works, not only depends on the packets mentioned above, it also depends on the type of Quality of Service attached to the published message. In such case, the communication will flow like this:

1. The client sends a request to connect to the server.
2. The server sends and acknowledgment to accept.
3. The client sends a subscribe request to the server.
4. The server responds with a subscribe acknowledgment.
5. Publish message:
 - The client publishes a message:
 - a. QoS - At most once: The server does not respond.
 - b. QoS – At least once: The server responds with a publish acknowledgment.
 - c. QoS – Exactly once:
 - i. The server responds with a publish received.
 - ii. The client responds to the server with a publish release.
 - iii. Finally, the server responds with a publish complete.
 - The server publishes a message
 - a. QoS - At most once: The client does not respond.

- b. QoS – At least once: The client responds with a publish acknowledgment.
- c. QoS – Exactly once:
 - i. The client responds with a publish received.
 - ii. The server responds to the client with a publish release.
 - iii. Finally, the client responds with a publish complete.

2.2.2.3. *Pushy*

Pushy, as its developers named it, works similarly to GCM, reason why and according to them is very easy to migrate from one to the other [44]. Moreover, Pushy is presented with the advantage of using MQTT protocol which take ones back to what the advantages of MQTT are (see section [2.2.2.2.](#)). Currently, it only supports Android OS and downstream messaging [45]. Its performance is so similar in fact, that it can be represented by the same steps as GCM, which in this case are:

- 1) The mobile phone registers to Pushy.
- 2) Pushy provides a “registrationID” or “token” to the mobile phone.
- 3) The mobile phone sends the “registrationID” or “token” to the application server.
- 4) If the application server, which is connected to Pushy, detects new information, notifies Pushy about it.
- 5) Pushy gets the information and sends it to the mobile phone.

It is also important to refer that Pushy provides 3 different services: Free service with no charge, Pro service that charges \$0.005 for each active device every month and Enterprise service [46].

2.2.2.4. *Parse*

Although this service was initially considered to be used, it was rapidly excluded as an option since the founders announced that their hosted service will be retired for good from January 28th, 2017 [47]. This would mean that if one used this service to develop the notification system of the mobile application, soon this service would have to be migrated to another, taking one back to stage one.

2.2.2.5. *Amazon Simple Notification Service (Amazon SNS)*

Amazon SNS [48] is a service that works as a broker between the application server and the Push Notification service. Therefore, it requires one of the following services [49] to work properly:

- 1) Amazon Device Messaging (ADM) [50] which is supported by Kindle.
- 2) Apple Push Notification Service (APNS) [51] which is supported by iOS and Mac OS X.

- 3) Baidu Cloud Push (Baidu) [52] which is supported by Web, Windows, Android, iOS and Windows Phone.
- 4) Google Cloud Messaging (GCM) [35] which is supported by Android, iOS and Chrome.
- 5) Microsoft Push Notification Service (MPNS) [53] which is supported by Windows Phone.
- 6) Windows Push Notification Service (WNS) [54] which is supported by Microsoft Windows and Microsoft Windows Mobile.

Amazon SNS service supports real-time events [55] which brings a tremendous advantage to mobile applications that are based on a real-time performance, such as applications that work as a messaging system. Moreover, it is extremely trustworthy and expandable, i.e. many events can occur and will always reach its destination. In fact, the main advantage of using Amazon SNS is that facilitates the integration of more than one platform. For example: Android mobile applications developed that use Amazon SNS should use GCM and iOS mobile applications can use APNS or GCM as well. If the developer rather uses APNS instead of GCM, the integration of the platform will be easier.

Moreover, Amazon SNS offers a message delivery analytics system through their console [56] where the devices registered can be also administrated. This service can be said to be inexpensive, since the provided cost for using it is free up to 1 million requests (per month). Thereafter, the cost will be \$0.50 per 1 million requests [57].

So, how will this communication actually work? For this, the communication among them was divided in the following steps (Figure 4), taking into consideration GCM as the service to be used:

- 1) The mobile phone registers to GCM.
- 2) GCM provides a “registrationID” to the mobile phone.
- 3) The mobile phone sends that “registrationID” to the application server. The application server registers that “registrationID”, which they call “token”, to a “PlatformApplicationArn” as a “PlatformEndpoint” from Amazon SNS.

Example of a PlatformApplicationArn:

```
arn:aws:sns:us-west-2:111122223333:app/GCM/gcmpushapp
```

Example of a PlatformEndpoint:

```
PlatformApplicationArn + “/5e3e9847-3183-3f18-a7e8-671c3a57d4b3”
```

Note: There will be a PlatformApplicationArn for each application in each platform (GCM, APNS, Baidu, etc).

- 4) If the application server, which is connected to Amazon SNS, detects new information, notifies Amazon SNS about it.
- 5) Amazon SNS gets the information and sends it to GCM, which is the Push Notification Service in use.
- 6) GCM sends the information to the endpoint.

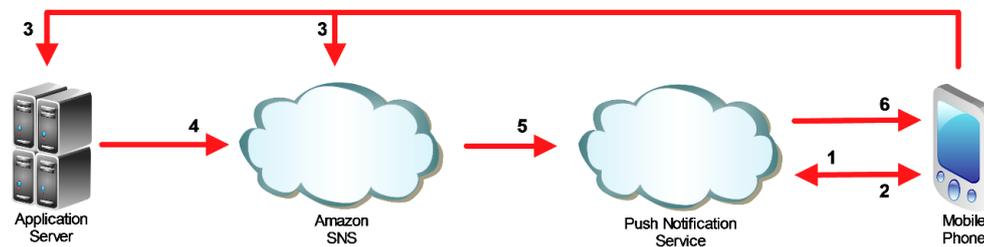


Figure 4 - Amazon SNS, Push Notification Service

2.2.2.6. Summary

To simplify the context, all the services explained are going to be summarized (see section [2.2.2.](#)) by characteristics and relevancy (see Table 2).

Table 2 - Push Notification Services

Service	Deprecated	Simplicity	QoS	Supported platforms	Cost	Advantage to centroproduto
GCM	No	Very simple	No	Android and iOS	N/A	-
MQTT	No	Simple	Yes	Android, iOS, Windows Phone and Kindle	N/A	-
Pushy	No	Very simple	Yes	Android	Has 3 services: Free service; Pro service: \$0.005 per active device; Enterprise service	-
Parse	Yes	N/A	N/A	N/A	N/A	N/A
Amazon SNS	No	Very Simple	No	Android, iOS, Windows Phone and Kindle	Free up to 1 million requests (per month) and \$0.5 per million requests thereafter.	centroproduto already uses Amazon Web Services

These characteristics refer to the service, whether the service is deprecated or not, the simplicity of integration of the service to the application. Also, it refers to which the supported platforms of each service are, whether it operates using Quality of Service or not and the cost of the service. The extra field refers to additional information about the utility of the service to *centroproduto*.

2.2.3. Image Loaders for Android

Considering that mobile applications regularly use images and that the images used by *centroproduto*'s system which are saved in their database are accessible by a Uniform Resource Locator (URL), it was necessary to find a way to upload those images to the application that will result in good quality images. Today, there are several ways to do so. One will now list some of them, as well as, their pros and cons.

2.2.3.1. *BitmapFactory*

Initially, one thought on uploading the images using *BitmapFactory* [58], which will convert the image from a specific URL to a *Bitmap*, but this brings some problems. Most of these problems are regarding to the mobile application's speed.

To start, one tried to upload several photos, one by one, to a listview. This caused such a delay that the list would take a while to load. Once this happened, one thought that perhaps the best way to upload the photos was by uploading and saving each one of them to a *bitmap* before displaying the list. However, this was not the best solution because this would cause that whoever that would use the application would wait for a long time until all the images were uploaded and saved, which in a case where the application has to upload at least 100 images would result in an extremely slow performance. Moreover, if one considers *centroproduto*'s images, it is important to state that to use them they have to be treated in order to fit the image view where they should be content which would result in a set low quality images. For these reasons, one had to consider doing this in some other way or in this case a using an API that will treat the image and upload it almost instantly.

2.2.3.2. *Picasso*

Even though *Picasso* [59] is considered by many as the best existent image loader, the truth is that one must use an image loader in accordance to the necessity of the user and the application. *Picasso* is very easy to use and saves images in the *ARGB_8888* *bitmap* format which according to Android Developers to each pixel there are 4 bytes available, and the RGB channels are stored with 8 bits of

precision [60]. Most importantly, it is very small and uses a small cache but it has huge disadvantages such as not saving thumbnails and it is very slow. Another disadvantage is that it would need an extension to be able to transform and crop images. However, there are extensions to this API, i.e. other libraries that can do the job. One of them is Picasso Transformations [61], that provide a series of different possible functions to transform and crop images. Some of them are:

- Crop Circle Transformation
- Crop Square Transformation
- Rounded Corners Transformation
- Color Filter Transformation
- Gray Scale Transformation

2.2.3.3. *Glide*

Glide [62] is very similar to Picasso. In fact, it offers the same functionalities. What differentiates the most Glide from Picasso is that Glide offers more functionalities, among them, the so called, transformations and the possibility to crop images.

Such similarities do not mean that both Picasso and Glide share the exact same characteristics. The truth is that Glide is very easy to use and it consumes less memory than other APIs. Also, it allows saving thumbnails, loading GIFs and caching in external storage. However, loading GIFs consumes a huge amount of memory. The main disadvantages of Glide are that it uses RGB_565 as bitmap format which according to Android Developers for a pixel there are only 2 bytes available and the RGB channels have 16 bits from which 5 bits are for red, 6 bits are for green and the last 5 bits are for blue [60].

2.2.3.4. *Fresco*

Fresco [63] is a library developed and released in 2015 by Facebook. However, it is more focused on memory level optimizations. To do so, Fresco uses the OS's ashmem which is another region of memory of the system that can be used for the image caching, i.e. it is focused on how the images are cached.

Also, this API offers a large list of functionalities such as cropping images around any point and not just the center, streaming progressive JPEG images and loading images from internet, local storage or resources. However, these advantages come with a huge price which is that this library is huge

and its cache occupies a large space. Moreover, it may freeze when large images are being loaded from a URL.

2.2.3.5. *Universal Image Loader (UIL)*

Probably the first image loader available to public use, as it was released in November 27th 2011 [64]. UIL offers the possibility to load images in both synchronous and asynchronous ways and is very easy to use. Also, it is considered to be highly customizable.

This API allows thumbnails and caching them and the original image in memory or disk and also allows transformations. However, it does not allow to specify the size for the treated image and it can consume a lot of memory.

In spite of being the most popular image loader that exists now-a-days, UIL is somehow compact if one considers having an application where treating an image is a must, as well as, having to upload many photos.

2.2.3.6. *Summary*

To simplify the context, one is going to summarize all the APIs explained in this section by specifying their characteristics such as complexity, cache and thumbnails (see Table 3). Note that to choose one of these, it is necessary to take into account the necessities of centroproduto and its users. To understand what these necessities are one must first explain the system requirements of this work (see section 3).

Table 3 - Image Loaders

Feature	BitmapFactory	Picasso	Glide	Fresco	UIL
Complexity	Very easy	Very easy	Very easy	Complex	Very easy
Format	ARGB_8888	ARGB_8888	RGB_565	Progressive JPEG	ARGB_8888
Cache	Yes	Yes	Yes	Yes	Yes
Size	Very small	Very small	Small	Huge	Small
Thumbnails	No	No	Yes	Yes	Yes
Speed	Slow	Fast	Fast	Very fast	Fast
Transformations	Customized	No – Use extensions	Yes	Yes	Yes
Memory consumption	Yes	No	Yes	Yes	Yes
Use of GIF	No	No	Yes	Yes	No
Cropping	Customized	No – Use extensions	Yes	Yes	No

3. System Requirements

3.1. Business context

centroproduto is a business-driven platform that works as a network of companies. This platform offers services to all the companies worldwide that pretend to buy/sell products and/or services in a dynamic and easy way. Also, it comes with the great advantage of allowing those companies to expand their business to other countries.

To take advantage of centroproduto's web portal the user must register an account on the system. Afterwards, he/she will have the opportunity to register his/her company or companies. Once the user has a company registered he/she can create a catalog offering its services and/or products and every time another person, or in this case business, is interest in any of them they will have the chance to contact that user and ask for more information. In the portal, the user also has the possibility to add other business to its network which will create a trustworthy relationship among them.

This platform has been online for two years now and it has been improving since then. For example, now it allows the users to communicate in an easier way by providing a chat-based messaging system. It offers a mobile web portal as well, and its looking for the possibility to improve even more the way the companies do business by developing a mobile application to ease the communication among them. To do so, it was necessary to define, together with centroproduto's team, the requirements of the application that one will expose in form of use cases (Figure 5).

3.2. Supported scenarios

Along with this diagram (Figure 5), one will be explaining the expected flow of the mobile application.

Note that in the use case diagram, there are use cases represented in a blue background and others with a yellow background. The use cases in yellow background are actions that the user can start in the mobile device but he/she will be redirected to centroproduto's web portal to complete the flow. On the contrary, use cases in blue represent actions that the user can perform in the mobile application.

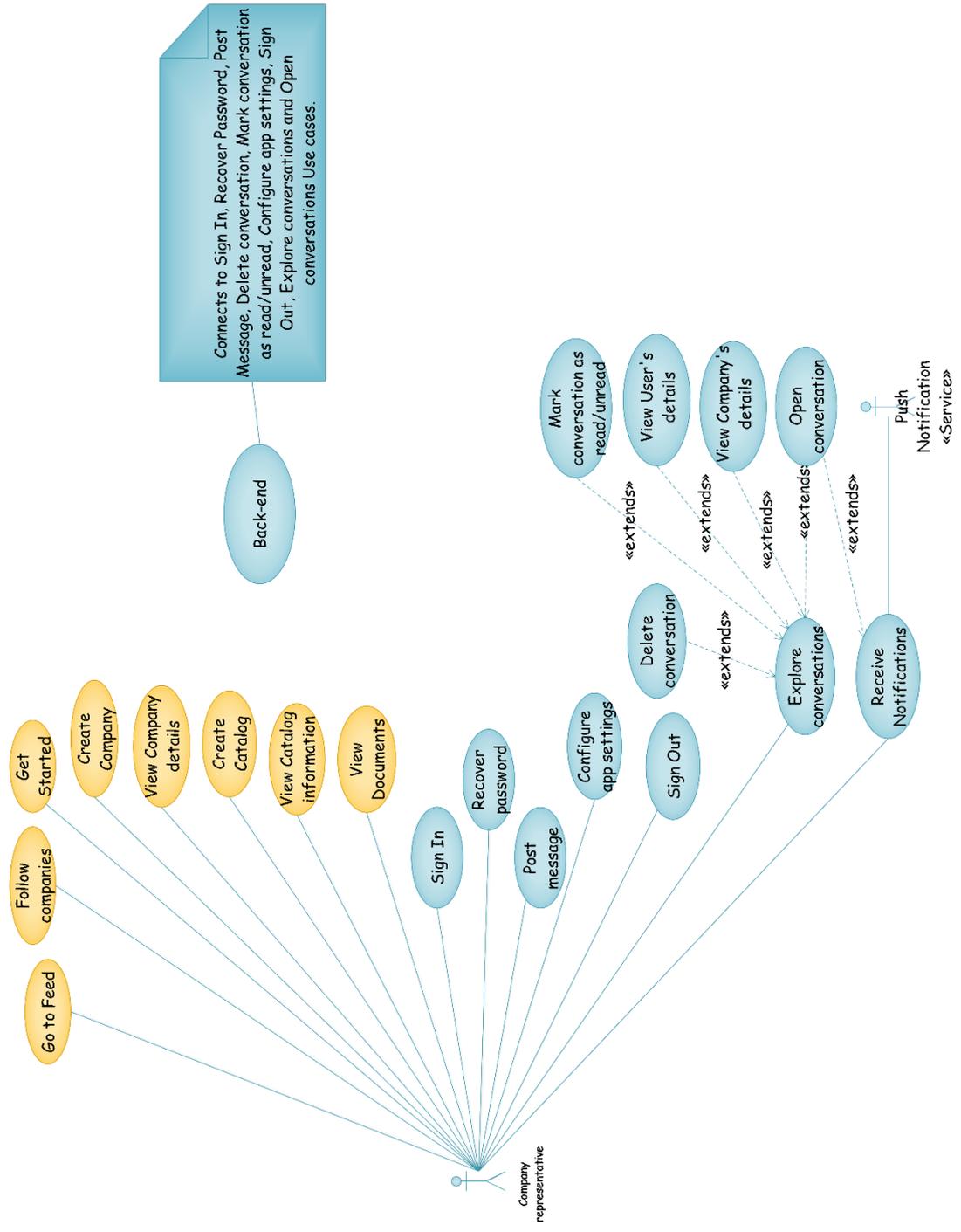


Figure 5 - Use case diagram

These use cases refer more specifically to:

- Sign in: The user must be able to Sign in to centroproduto's system.
- Recover password: The user must the possibility to recover his/her password through the application.
- Post message: Refers to the possibility of sending messages from the application.
- Configure app settings: the user must be able to realize some actions related to configurations of the application and the system. These configurations are: changing the logged company, changing the language of the application, enabling/disabling the reception of notifications and enabling/disabling the sound reproduced when a notification is received.
- Sign out: The user must be able to Sign out of centroproduto's system.
- Explore conversations: This use case extends important actions that the user must be able to perform. These actions include allowing the user to delete conversations, mark conversations as read/unread, view the user's and company's details such as names, photos and logo of the company. Also it includes the possibility of opening new and old conversations.
- Receive notifications: It refers that the application must receive notifications about new information in the system, in this case messages.

3.2.1. User Authentication

Let's start by talking about the authentication process. To follow how this process work, see Figure 6 which is an activity diagram that represents the interaction of the user with the system during the login and recovery of password process.

First, the user must open the application and then the application will have to detect whether the user has already signed in or not. Now depending on whether the user is logged or not, one of two things can happen:

- 1) If the user is not logged, he/she is redirected to the Sign in view.
- 2) If the user is logged in, he/she is redirected to the List of conversations' view.

Supposing that the 1st point is true, the user will be asked to insert his/her information. To do so, he/she will have to ask himself/herself "Do I remember my password?". If not, he/she will have to

provide his/her email to the application and then click on a recovery button. Afterwards, if the email is accepted, the user will receive in his/her inbox an email with a recovery link.

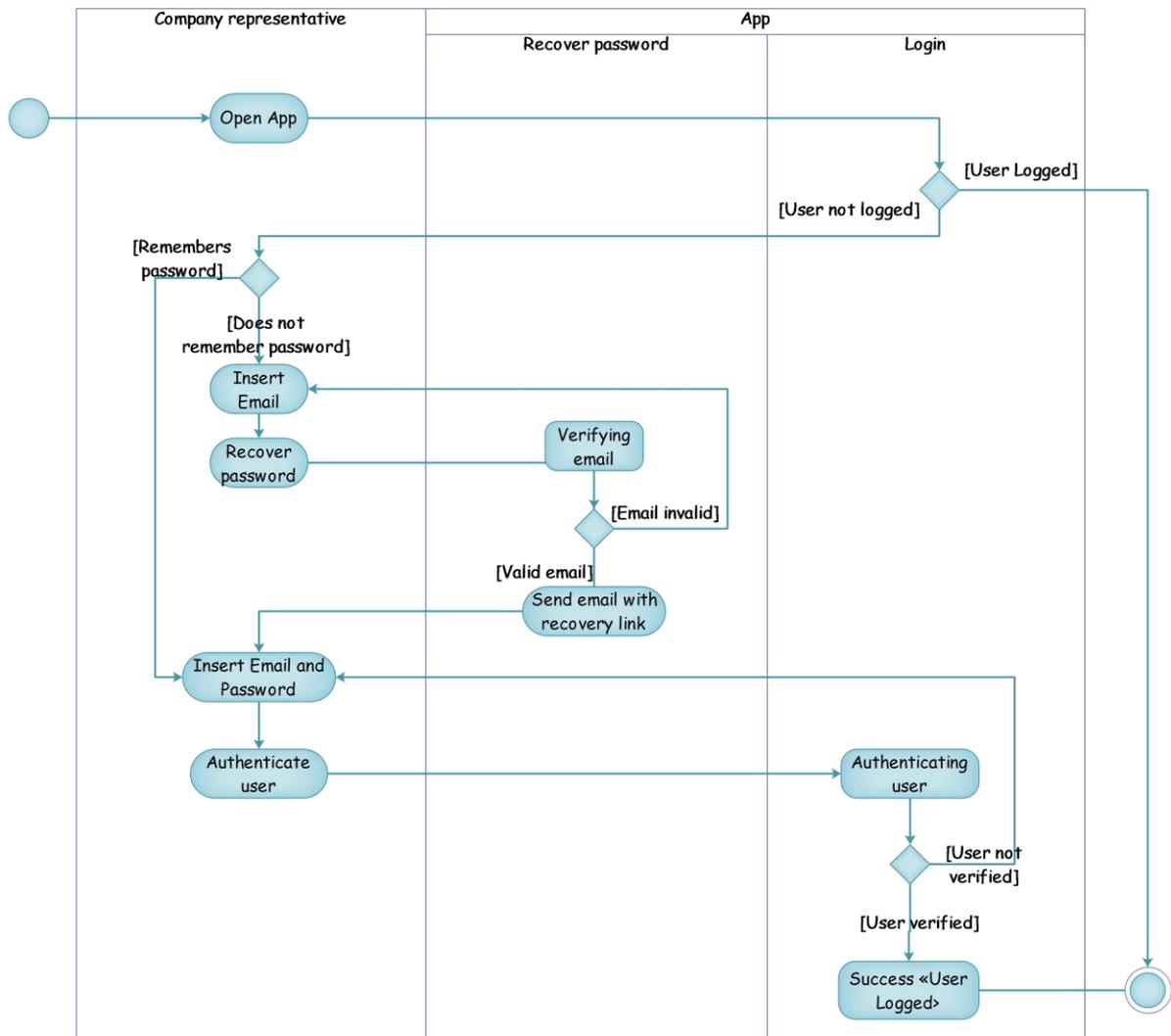


Figure 6 - Activity diagram - Sign In

On the contrary, if the email is not accepted he/she will be asked to insert a valid email. Now, supposing that he/she does remember her password. He/she will have to insert his/her email and password in the application and then click on Sign in to get authenticated. Later, if he/she was verified, he/she would have logged in successfully and if not, she will return user to stage one.

Note that, the authentication of any user to the system is handled by the company's Back-end. The email and password provided by the user are sent to the company's back-end through a REST service and then the back-end will send information acknowledging about the authentication, i.e. whether it was successful or not.

3.2.2. Messaging and Notification related Use Cases

Regarding to messaging and notification, it is important to mention that “Receive Notifications”, “Explore conversations” and “Open Message” use cases somehow relate to each other. This because the reception notifications and the exploration of conversations will trigger the opening of conversations by the user. For better understanding see Figure 7.

3.2.2.1. *Display of notification’s list*

This Use Case represents the view of the conversations list. First of all, and as its name refers, it contains conversations that hold the messages sent and received. Each one of them is characterized by having the name of the user who sent the message, as well as, his/her profile photo, the message’s subject and the date of the last message received (or sent if that is the case).

The conversation/notification list itself will be obtained by asking the Back-End about the past messages of the user in the system. From then on, to receive new messages, one of the Push Notification Services that were previously explained will be used (see section [2.2.2.](#)).

To obtain such list, it is mandatory that the user has already logged in his/her account. Afterwards, one of two things can happen (Figure 7):

- 1) The conversations list is empty.
- 2) The conversations list has active conversations.

In the first case (1), nothing will be done. If the second case (2) is true, the list of conversations will be displayed. In both cases, the application will wait until a new message is received. Every time a new message arrives, it will be displayed the full list of conversations.

More actions can be performed, starting from the displayed list. These are: open a menu, send message and open a conversation (see section [3.2.2.3.](#)).

In case that the user wants to create a new conversation, he/she should be redirected to the website to do it.

3.2.2.2. *Reception of Notifications*

As previously said, the client must be allowed to receive notifications. To understand how the user can receive notifications, let’s start by focusing on the following example: A company representative registered in centroproduto wants to do some business and sends a message to

another company. If the company representative of this other company has centroproduto's mobile application, he/she will receive a mobile notification with an indication that he/she has new messages to read. Note that these notifications arrive asynchronously to a mobile application.

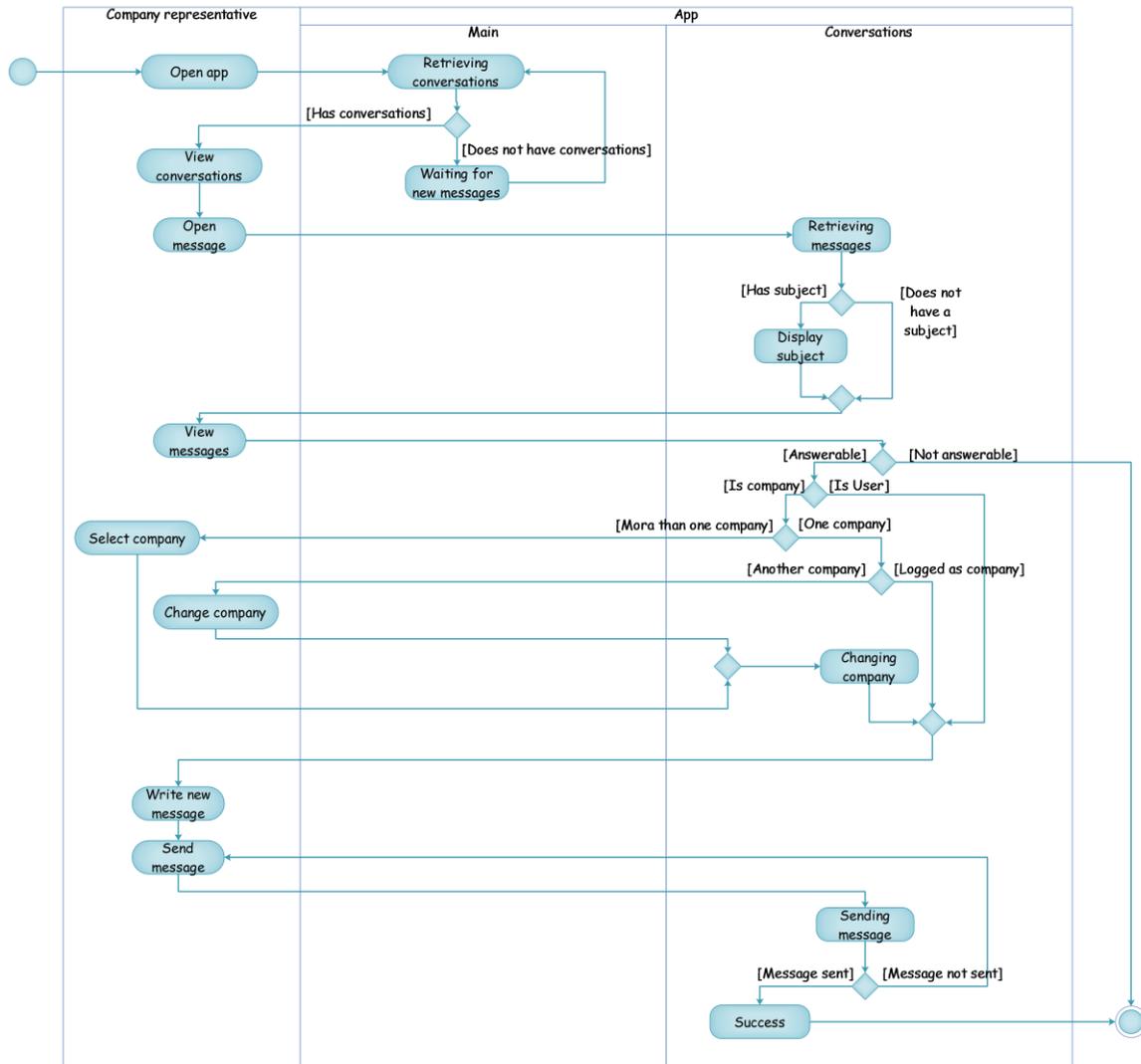


Figure 7 - List conversations

3.2.2.3. Opening conversations

First of all, when the user opens a message, the Back-End is asked to retrieve all the past messages of that specific conversation. Within the conversation, the user will be able to read all the messages and even reply, depending of the type of message received. One of those types of messages are “no reply” messages sent from centroproduto's system to its users which does not allow the user to reply. For better understanding, see Figure 7 and the list below containing the conditions that have to be considered for it to work properly.

1) In answerable conversations:

- a. If the user is participating in a conversation as user, he/she will be able to answer the message without any problem.
- b. If the user is participating in a conversation with more than one company, he/she will be able to choose with which one of them he/she wants to use to reply to a message.
- c. If the user is participating in a conversation as both user and company:
 - i. If the user initiated the conversation as user, including himself/herself as company, he/she will only be able to answer the conversation as user.
 - ii. If the user initiated the conversation as company, including himself/herself as user, he/she will only be able to answer the conversation as the company he used to start the conversation, being necessary for him/her to be logged as the company participating in the conversation.
- d. If the user is logged as the company involved in the conversation, he/she will be able to answer to the message immediately. Otherwise, he/she will have to change the company he/she is logged in with to the one involved in the conversation.

2) In conversations that the user cannot reply to the messages:

Considering that the message type is “no reply” when the user opens the conversation all the messages are listed and then he/she will be able to read the new message. Moreover, this type of messages will always have a subject which will be displayed as well.

Note that, for any of these cases, the user can only send a message whenever the message is not empty.

3.2.2.4. *Configuration of settings*

Being this project’s aim developing mobile application that works as a messaging system. The user has to be allowed to configure the settings of the application. To understand better all of these actions, see the list below.

- “Activate Company”: Imagine that the client has 2 companies, Company1 and Company2. He/she should be able to switch from Company1 to Company2.
- “Enable Notifications”: The client must be able to enable/disable notifications at any time.
- “Enable Sounds”: Works similarly to the previous Use Case, the difference is that only the sound will be affected.

- “Change the application’s language”: The user should be able to change the language of the application at any time, according to the languages supported by centroproduto’s system.

3.3. Cross-cutting requirements

There are some requirements that cannot be directly linked to the system or the previous scenarios but that are of extreme importance to the application in development. These requirements are contained in Table 4.

Table 4 - Cross-cutting requirements

Data Integrity	<ul style="list-style-type: none"> • All the data in the website and the mobile application must be synchronized. This means that, if User A sends a message to User B from the website, the message should appear in the User’s A mobile phone as a message sent. • The user’s information should be validated. • The information retrieved from the Back-End (BE) must be saved as long as the user is logged.
Usability	<ul style="list-style-type: none"> • Hold usability tests with company representatives, where they can try and perform the most critical actions of the system, like opening conversations and receiving notifications.
Efficiency	<ul style="list-style-type: none"> • The application must be as light as possible, even when the user is logged, in order not to consume much memory.
Response	<ul style="list-style-type: none"> • The application must relay in real-time events to work properly, i.e. it will be receiving new messages from other users. This, will relay in a Push Notification Service to work properly (see section 2.2.2.).
Portability	<ul style="list-style-type: none"> • The application must allow the user to use it at any place.
Compatibility	<ul style="list-style-type: none"> • The application must be compatible with Android OS 4.0 to 6.0.

3.4. Planned application outlined

The application will have a menu which will be accessible through the conversation’s list. It is important because it contains important actions that a user can execute in the system, i.e. in centroproduto’s system.

The options that will be shown in the menu will be depending on the following characteristics (Figure 8):

- a) Does the user have a registered company in the system?
- b) Does the user have a catalog for the company?

Also, it is important to highlight that the former question (b) depends on the result of the first one (a).

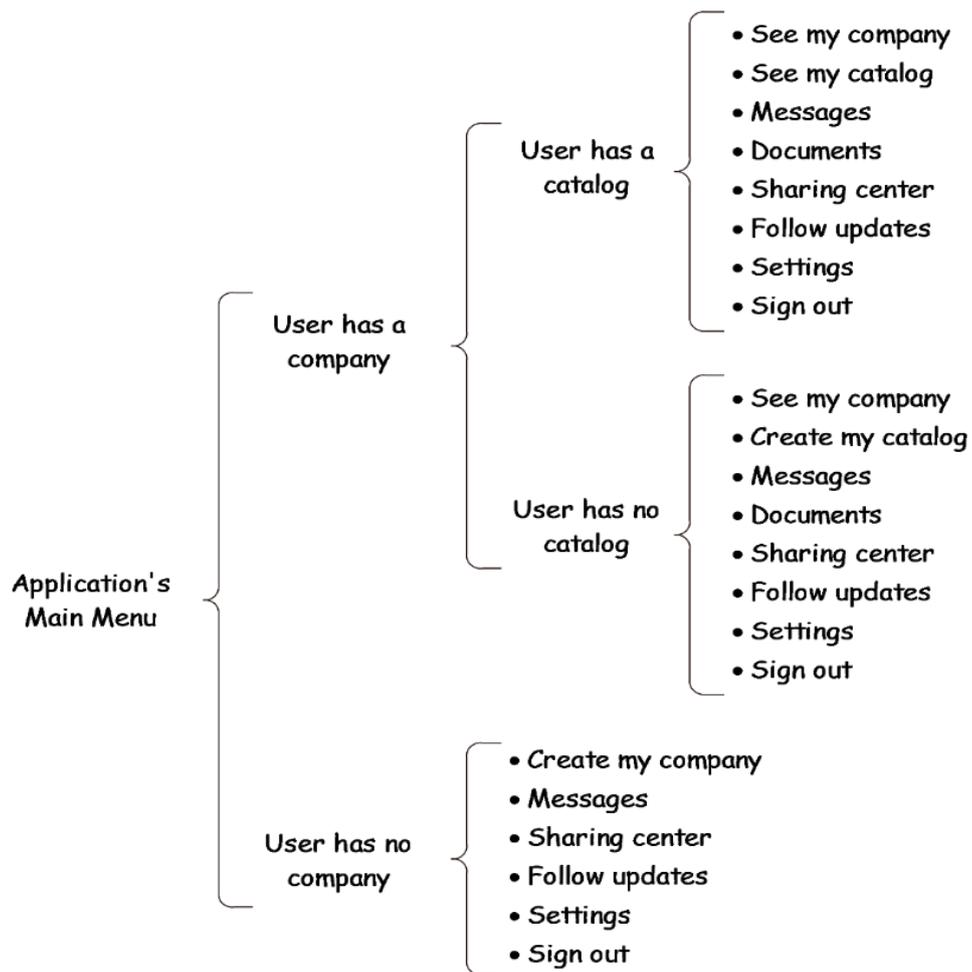


Figure 8 - Application's menu

3.5. Summary

To sum up, the requirements are going to be divided in Table 5 by importance, where “highly important” means that the requirement must be implemented, “fairly important” means that the requirements should be implemented and “less important” that should be implemented if possible.

Table 5 - Requirements importance

Importance	Requirements
Highly Important	<ul style="list-style-type: none"> • Login • Explore conversations • Receive notifications • Open notifications • Post message • Configure application's settings • Sign out
Fairly Important	<ul style="list-style-type: none"> • Recover password • Mark conversation read/unread • Delete conversation • View user's details • View company's details
Less Important	<ul style="list-style-type: none"> • Register (Get Started) • Create new message • Create company • View company's details • Create catalog • View catalog's information • View Documents • Follow companies • Go to feed

4. System Architecture and implementation

4.1. System Architecture

The system architecture incorporates the components involved in the implementation of the application. Starting from using a connection to the back-end of centroproduto through a REST service, until the usage of a push notification service that will allow the communication between company representatives. The notification service that will be used is Amazon SNS which demands the use of another service in order to properly work. This other service will be Google Cloud Messaging since the application is targeted for Android OS. Having said that, the proposed system architecture is represented in Figure 9.

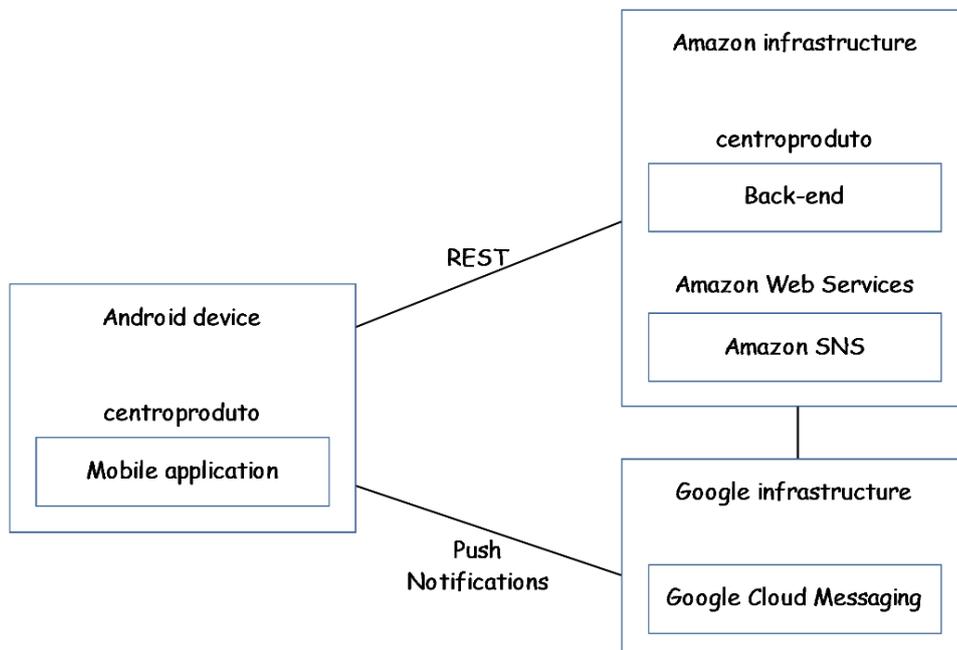


Figure 9 - Overall System Architecture

Native Application Vs Web-based Application

The development of native applications, unlike web-based applications, brings advantages in terms of capability, performance and feature usage [65]. Moreover, they provide best mobile experience to the users and can be interactive, eye-catching, simple and user-friendly.

Web-based applications, on the contrary, are based in the integration of web views and/or browser. These allow that the application uses the web portal to perform actions of the system, by being uploaded.

Although this application could be implemented by using web views, a native application aside from providing a faster performance is necessary to be able to implement the Chat Head feature. It also allows multi-touching, the use of animations and using other features of the smartphone such as the camera and GPS.

Connection to Back-end

There are few reasons why the connection to the Back-end has to be established. First of all, this connection is established through a REST API service in which the information is sent in JavaScript Object Notation (JSON) format [66] which is a data-interchange format that simplifies the way how information is read and written.

To start, the connection in which the information will be sent (or requested) will be opened using java's class "URLConnection" [67] to a specific URL provided by the company, i.e. by centroproduto. Note that there will be a new connection for each URL provided. Each connection is provided with a method request (POST, GET, PUT, HEAD, OPTIONS or DELETE) and depending on that request, information will be sent and/or received through it.

As asking the back-end for information every time the application executed would be not only time consuming but it would be power consuming as well, one had to find a way to keep the conversations activated, and the best solution is to integrate a push notification service (see section [4.1.3.](#)).

Amazon SNS Service

After studying the Push Notification Services explained before (see section [2.2.2.](#)), one decided, together with centroproduto's team, that the service that brings most advantages to centroproduto is Amazon SNS, due to the following reasons:

- 1) centroproduto already uses Amazon Web Services which would simplify the integration of this service into the system.
- 2) Amazon offers integration with many platforms [49], such as: Android through GCM, Apple through APNS, Windows Phone through MPNS and Kindle through ADM. Any of these platforms will be responsible by sending push notifications.

The most important advantage of Amazon providing this kind of integration is that after developing a mobile application for any of those platforms, if centroproduto's developer team decides to

develop a new application for any other platform, it will not be necessary an extensive change of many aspects of the back-end's code. The code would remain the almost same, since the changes that would be done would be an adaptation, in order to support both platforms.

Since, Amazon needs the help of another Service to send Push notifications and being this application is developed for Android OS, the service that needs to be used is Google Cloud Messaging (GCM).

Using Amazon SNS will simplify the application by fulfilling the following aspects of it:

- 1) When a user receives a new message this service will be used to push the message to the mobile application. This way the user will be always updated on what is going on with his/her account.
- 2) When the user sends a message through the website the message should appear automatically in the mobile application. To do so, this service will be send a push message with the content of that message. This will allow both the website and the mobile application to be consistent.
- 3) When the user marks a message as read or unread the application must be notified in order to mark the same conversation as read or unread if that is the case. This, as the point above, will make that the website's and the mobile application's information is consistent.
- 4) The same happens when the user deletes a conversation at the website. The conversation has to disappear from the mobile application, make the content consistent.

4.2. Android Implementation

4.2.1. Targeted versions

The main objective of this application is to target most users as possible, the application now supports version 16 to 23, i.e., Android 4.1 (Jelly Bean) to Android 6.0, which according to Android Developers are the Platform Versions with highest percentage of distribution [68].

The default configuration of the file build.gradle which is a configuration file of the project where all the dependencies, project type and targeted versions are defined, was defined accordingly. An exert of the configuration file can be seen in Table 6.

Table 6 - Exert of the Gradle configuration

```
defaultConfig {  
    minSdkVersion 16           //Minimum targeted version  
    targetSdkVersion 23       //Targeted version  
    versionCode 1             //Application's version code  
    versionName "1.0"         //Application's version name  
}
```

4.2.2. Layouts

The application is going to be used in many different mobile phones and tablets. Today, there are many different screen sizes and densities, and by different platform versions, two layouts were created for each view. One for mobile phones and one for tablets.

The views not only depend on these layouts; they also depend on the density of the screen where the mobile application is being executed. For example: the images displayed on the notifications list also depend on the density of the screen:

- 1) The default profile photos depend on different paddings. In screens where the density is higher the padding will have to be higher in order that the images look similarly.
- 2) The logos that are displayed depend on different heights according to the density of the screen. Also, the default logos depend on a different padding for each density.
- 3) The Chat Head size in each screen depend on the density of the screen.

Android has a class that return all the metrics of the display, including density which is the logical density of the screen and the density in dots p/inch [69]. These metrics can be retrieved from the mobile phone by using the lines of code in Table 7.

Table 7 - Exert of Density metrics code

```
DisplayMetrics displayMetrics = this.getResources().getDisplayMetrics();  
int density = displayMetrics.density;  
int densityDpi = displayMetrics.densityDpi;
```

Afterwards, the result density will be compared among the possible densities, until a suitable one is found (see Table 8). As it is already known, these possible densities are:

Table 8 - Density Metrics - Value

Density	Density Dpi	
0.75 – ldpi	120	DENSITY_LOW
1.00 – mdpi (default/baseline)	160	DENSITY_MEDIUM
1.50 – hdpi	240	DENSITY_HIGH
2.00 – xhdpi	320	DENSITY_XHIGH
3.00 – xxhdpi	480	DENSITY_XXHIGH
4.00 – xxxhdpi	640	DENSITY_XXXHIGH

What this table says is that for every screen where the density is 0.75, the density Dpi will be 120dpi and the same happens for the other levels of density.

4.2.3. Amazon SNS

To use this service, one had to follow the steps provided by Amazon [70], which are the following:

- 1) Complete the procedure to be able to use GCM (register the app with GCM, get a registration ID (sender ID) and a Server API key) [71].
 - a. After getting the sender ID, one had to integrate it in the code, specifically in the strings.xml file, in order to use it later for the registration of the device (see Table 9).

Table 9 - strings.xml – Example: Sender ID

```
<string name="sender_id">1234567891011</string>
```

- 2) Once one had done that, one followed an example provided by Amazon [70]. Such example states that one must create a new class which extends the class Service.
 - a. This Service will be responsible for registering the device to GCM and Amazon SNS when the user logs into his/her account in centroproduto’s app. In order to register the device, it was necessary to instantiate a class of GCM to later get the token/registration ID of the device (see Table 10). Please note that, the sender ID that was previously saved in the strings.xml file is being called in order to get a new Token for the device.
 - b. Apart from registering, the token will be later sent to the BE in order to register it on Amazon SNS.

Table 10 - Device Registration to GCM

```
/*Get an instance of InstanceID*/
InstanceID id = InstanceID.getInstance(getApplicationContext());
/*Through the InstanceID request a token which will be used as
identifier of the device. To request the token, the sender id must be
sent as argument*/
String token = id.getToken(getString(R.string.sender_id),
GoogleCloudMessaging.INSTANCE_ID_SCOPE, null);
```

- c. This service will be in charge of posting a Notification on the mobile phone to acknowledge the user about a new message, and/or to send the information straight to the application.
- 3) Although the notifications received are treated by this Service, the application will have to user a Broadcast receiver that will stay awake and listening for new notifications. Afterwards, when a notification arrives, this broadcast receiver will have to redirect the message to the service.

Apart from the code developed on the mobile application's side, there was code developed on the back-end's side. This code, will be in charge register each token received to a "PlatformApplicationArn" as a "PlatformEndpoint". Afterwards, every time new information is detected, this code will be in charge of publishing the information, i.e. of sending it. For example: imagine that Mary sent a message to Joseph, once the message is sent, Amazon SNS will have to get that message and sent it to Joseph by publishing the message. The code developed on the back-end's side was developed using Boto 3 Docs [72].

4.2.4. Chat Head Service

As it was previously mentioned, the Chat Head has to be part of the application. To implement it, it was necessary to create a new class which extends Service. This class will be responsible for:

- 1) Popping-up the Chat Head (see section [5.1.13.](#)) on top of the screen when a new message is received and the application is closed.
- 2) Allowing the user to click on it and open the application. This will be triggered by an onClick event.
- 3) Allowing the user to long click on it and move the Chat Head around or even close it.

To actually create this Chat Head, the views have to be manually insert into a Window. First, a Relative Layout is inserted. This layout will contain the image view of the Chat Head. In the case that the user wants to close the application (3), another Relative Layout will show up with a circular form and an “x” inside of it.

The Drag and Drop Event and the Touch Event are both very similar which means that any of them can be used. However, one opted to use the Drag and Drop Event because unlike the touch event, this will cause that the user has to wait for a bit to actually be able to move the Chat Head around the screen and close it.

The idea of this application is to engage the user as much as possible. So, as long as he/she has the Chat Head open he will not forget that he/she has new messages still to read.

If the user wants to move the Chat Head (3), after a long click the onLongClick event will be triggered and he/she will be able to move it around. To do so, a ShadowBuilder is created in order to show the ChatHead moving around the screen. The ShadowBuilder is an Android’s class that creates a copy of an image to be displayed by the system during the drag and drop operation [73]. See an example of code in Table 11.

Moreover, the button to close the Chat Head will instantly appear after a long click.

Table 11 - ShadowBuilder Code

```
/*Create an empty ClipData where the image to be dragged is going to be
copied to*/
ClipData clipData = ClipData.newPlainText("", "");
/*Create a shadow builder using the view that needs to be dragged*/
View.DragShadowBuilder shadowBuilder = new View.DragShadowBuilder(v);
/*Start dragging the image from one place to other*/
v.startDrag(clipData, shadowBuilder, item, 0);
```

4.2.5. Activities

The content of all the lists of this application were obtained through a HttpURLConnection to the Back-end which uses a REST service. Considering that the user is logged in, the information than can be obtained is:

- Provides the information of the user and the list of companies of the user.

- Provides the list of conversations of the logged user.
- Provides the list of messages of a specific conversation.
- Provides the list of users that participate in a specific conversation.

For better understanding see Figure 10, where the classes containing all the information of these lists is described.

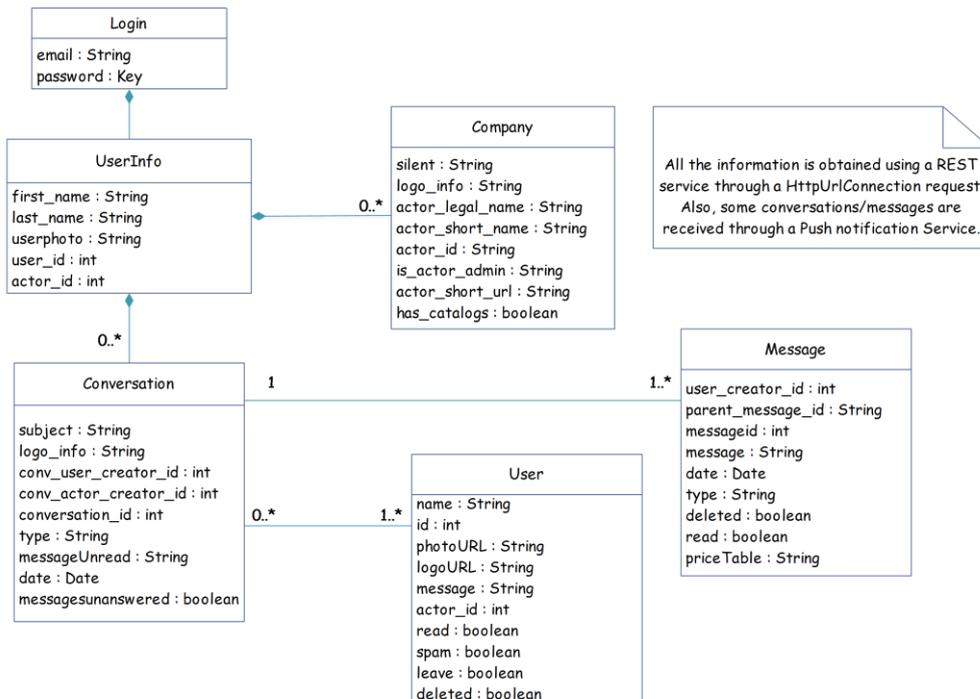


Figure 10 - Class diagram

In this diagram, it is possible to see that the user logged has access to his/her own user information which includes having access to his/her registered companies and conversations. Each conversation of the user, aside from having the id of the conversation, the type of conversation and so, also has a list of messages associated to it and the list of users who participate in the conversation.

Each one of those lists, will be displayed in the application using different row layouts. The list of conversations has now 5 different layouts while the list of messages has 2. The list of users has only one layout. Also, there is a list that was not mentioned which is the list contained in the drawer menu. To inflate the list, it will be necessary one layout as well.

Since to display this information in the application it is necessary to use an adapter. One adapter was created to inflate the given information in the correct fields of the mentioned layouts. For example: at each row a new conversation should be displayed showing different users and subjects.

Table 12 - Example ViewHolder Class

```
/*Create a static class containing the elements that have to be
inflated to a view*/
static class ViewHolder {
    public ImageView icon;
    public TextView item;
}
```

Table 13 - Example Row Inflation - getView Function

```
/*Create empty view*/
View view = null;
/*If existing view is null*/
if (convertView == null) {
    /*Inflate the view using LayoutInflater using R.layout.menu*/
    LayoutInflater inflater = (LayoutInflater)
context.getSystemService( Context.LAYOUT_INFLATER_SERVICE );
    view = inflater.inflate(R.layout.menu, null);
    /*Instantiate the class previously mentioned*/
    final ViewHolder viewHolder = new ViewHolder();
    /*Initialize the components of the class according to the elements in
the view*/
    viewHolder.icon = (ImageView) view.findViewById(R.id.icon);
    viewHolder.item = (TextView) view.findViewById(R.id.item);

    /*Copy the content of the instantiated class to the empty view*/
    view.setTag(viewHolder);
} else {
    /*Copy the existing view to the empty view*/
    view = convertView;
}

/*Set the information that must be displayed by the view in each component.
ImageView will contain list.get(position).getIcon() and TextView will
contain list.get(position).getItem().*/
ViewHolder holder = (ViewHolder) view.getTag();
holder.icon.setImageResource(list.get(position).getIcon());
holder.item.setText(list.get(position).getItem());

/*Return the new view*/
return view;
```

To be able to upload that information to the row, a new ViewHolder [74] has to be created and inflated. As example see Table 12 and Table 13. The content of Table 13 that if the view does not exist, it will first be inflated and then the data will be uploaded to it. Otherwise, the existing view will be updated.

4.2.6. Library for image optimization and access

The Image Loader API that was chosen was Picasso, because of few things:

- 1) Because of the light weight of the API.
- 2) Because of its simplicity.
- 3) Because of the resulting image quality.
- 4) And, because it uses less cache memory to cache all the images, which will make the application lighter.

Since Picasso does not support many kinds of transformations, one opted for using the extension that was previously mentioned (see section [2.2.3.2.](#)) to resize the images and add rounded corners to the images in the application.

The integration of this API was very easy thanks to the specifications in their website [59]. First, one downloaded the jar file in the website and imported it to the project. As previously mentioned, an extension [61] to Picasso was used as well. This extension allows to create images with rounded corners (see Table 14).

Table 14 - Picasso and crop transformations

```
/*Create a new crop transformation. The width of the new image will
be 150 and the height is 120. The crop gets the right part of the
image*/

CropTransformation transformation = new CropTransformation(150, 120,
CropTransformation.GravityHorizontal.RIGHT,
CropTransformation.GravityVertical.CENTER);

/*To apply a rounded corners transformation it is necessary to apply
a level of radius and indicate the corners where that radius should
be applied*/

RoundedCornersTransformation roundedCorners = new
```

```
RoundedCornersTransformation(7, 0,  
RoundedCornersTransformation.CornerType.LEFT);  
  
/*Load the image using Picasso, which result width should be 300 and  
height should be 120 to which both transformations should be applied  
(CropTransformation and RoundedCornersTransformation).*/  
Picasso.with(context)  
    .load(list.get(0).getImage())  
    .noPlaceholder()  
    .resize(300, 120)  
    .transform(transformation)  
    .transform(roundedCorners)  
    .centerCrop()  
    .into(holder.image);
```

What this means is that the image to upload will be cut in half of its original size starting from Center-Right (Crop Transformation) and then the rounded corners will be applied to the left corners of the image (Rounded Corners Transformation). After that, the image will be uploaded into holder.image.

5. User Interactions Design

The design of the user interface plays a crucial role in mobile applications [75]. In this project the user experience design was an evolutionary work, with the joint participation of centroproduto's design team. In this chapter, one is going to describe the mobile application's design, such as prototype and libraries used. Moreover, one should state that this application will be only supported by Android OS, reason why the prototype will be only represented for such operating system.

5.1. Android prototype

Thanks to the Design team at centroproduto it is possible to present a prototype for this application. To create this prototype, the team studied what the Wireframes for Android applications are [76]. Furthermore, they had to consider the website's look in order to achieve a consensus among the application and the website, so they both look alike. The main idea was to design and develop a mobile application that would look the most similar to the website. The application has to look like a chat-based application that allows the user to know which conversations are unread, which are read and answered or not. Moreover, the application has to notify the users about new information in the best way possible.

5.1.1. Colors and Icons

According to what is used to design and implement centroproduto's website, the following list of colors that must be used to design the mobile application, as well as, to develop it.

● Gray – #333333	● Yellow – #F3DB8F
● Gray – #999999	● Yellow – #FFCD2
● Gray – #CCCCCC	● Yellow – #FAF1DA
● Gray – #EEEEEE	● Red – #C16347
● Gray – #F3F3F3	● Red – #FE6C3E
● Yellow – #8A6821	● Green – #7C9E3F
● Yellow – #FEC53E	○ White – #FFFFFF

Moreover, there is a list of icons that must be used which were taken from a free Icon library called FontAwesome [77]. Each icon was downloaded in 5 different sizes and colors depending on what

their purpose is on the application. The idea of downloading the icons in different sizes comes from having to support different screen sizes [33].

5.1.2. Launcher Icon and Splash Screen

The launcher icon of the application plays an important role because it will be used to access the application. This icon not only will be used for that matter as it will also be used to represent the application in the main screen (Figure 11), or at the applications list of a mobile phone (Figure 12).

Aside from the Launcher icon, today, developers use a method called Splash which introduces not only brand or product of a company but the start of an application. In this case, centroproduto's application will have a Splash screen represented by centroproduto's logo (Figure 13) and it will last 1.5 seconds to disappear.



Figure 11 - Launcher button - Main Screen

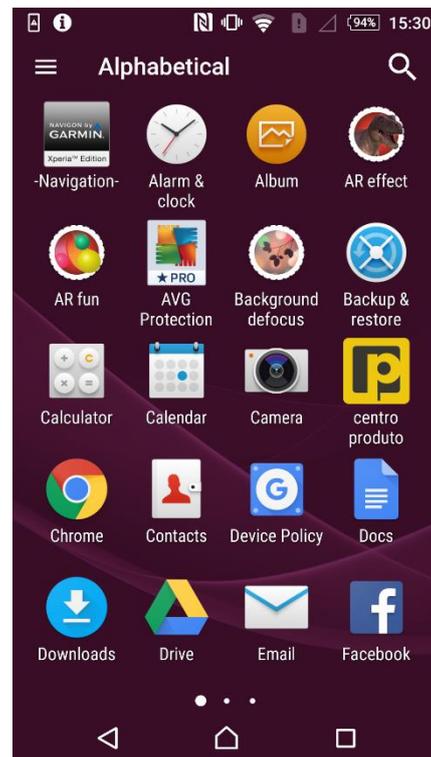


Figure 12 - Launcher button - List of Applications

5.1.3. Authentication

This screen allows a potential client to sign in. It has a dark gray background (see section 5.1.1.), with a couple of fields where the user can insert his/her email and password. It shows the logo of centroproduto and options to recover the password and register.

To sign in, the user inserts his/her email and password and clicks on the "Sign in" button (Figure 17). However, there are a few situations to be considered. These situations are:

- a) What would happen if the client clicks on the button without inserting his/her email and password?
- b) What would happen if the client deletes his/her email and/or password from the fields?
- c) What would happen if the email or password is incorrect?
- d) What would happen if the inserted email does not exist?
- e) What would happen if the email and password are correct?

To answer the first question (a), if the client does not insert an email and password, then it will not be possible for him/her to sign in. Moreover, a message saying "Field required" is going to be shown under each field, i.e. email and password, respectively; and the stroke around each of these will turn red. See an example of this message in Figure 14. After considering the situation above, it is important to point out that that same message will appear every time the client deletes the text in the email and/or password field.

Now, to answer question (c), imagine that one actually inserts all the information requested, i.e. the email and the password, but either the email or the password are incorrect. The mobile application will show the message "Verify the email and password" to let the client know that one of them is incorrect (Figure 15).

This same example could be used to answer the question (d), what would happen if the inserted email does not exist? The difference is that the email does not exist or has not been registered yet and a result message will be shown asking to "Type an email which is already registered on centroproduto".

Last but not least, if both email and password are correct (e), a progress bar will be shown as feedback. The progress bar will be represented by the short logo of centroproduto (Figure 18) and will notify the user that information is being uploaded to the application and that he has successfully logged into the system. Afterwards, he/she will be able to see all its previous messages, reply and receive new ones (Figure 19).

If by any reason the client forgot its password before login in, he/she will be able to click on "Did you forget your password?" and be redirected to a recover view. There he/she will be asked for the registered email. When the client inserts the email in the recovery field and clicks on "Recover

Password”, he/she will receive a feedback from the application to let him/her know whether the email is valid and to let the user know that an email was successfully sent to his/her inbox (Figure 16).

If instead the user is not registered to centroproduto, he/she will have the opportunity to register to centroproduto by clicking on “Get Started”. Afterwards, the client will be automatically redirected to centroproduto’s registration website so he/she can create its company’s account.

5.1.4. Conversation List

The conversation list is the main and for sure the most important screen in this mobile application. In this screen the client will be able to see, who sent him/her messages, as well as information coming from centroproduto (Figure 19). These messages will be presented in form of a list. In the list the messages will be organized by (see section [5.1.1](#)):

- a) Messages to read: This type of messages can be distinguished by a yellow background.
- b) Messages read unanswered: These messages instead, will be differentiated from the others by their white background.
- c) Messages read and answered: Can be recognized by their light gray background.

All the messages in the list, will have one of the following 5 formats:

- 1) Conversation with 1 user (Figure 20): This conversation will be characterized by having:
 - The logo of the user’s company
 - The profile photo of the user
 - The name of the user
 - The name of the user’s company or “(user)” if he/she is sending the message as user.
 - The subject of the message/conversation or “(no subject)” if it does not exist.
 - The date of the last message sent or received.
- 2) Conversation with 2 users (Figure 21): A conversation with two users will be represented a little bit different. The conversation row will show a merge of both the logos and both photos of the users involved in the conversation. In such case, the characteristics presented will be the following:

- The logo of each user's company: As there are 2 possible images and the space available is the same as if there was only one photo, each image will be divided in 2 and merged.
 - The profile photo of each user: The profile photos will merge the same way the logos did.
 - The name of the users: The names will be separated by commas.
Example: Sofia Góis, Ana Góis
 - The name of each user's company or "(user)" to represent when the user is not sending messages related to the company: It follows the order of the list of users.
Example: If both are in the conversation as users: (user), (user); or if one of them is in the conversation as user: (user), Company1; or if both are sending messages related to their companies: Company1, Company2.
 - The subject of the message/conversation or "(no subject)" if it does not exist.
 - The date of the last message sent or received.
- 3) Conversation with 3 users (Figure 22): As this is a conversation with 3 users, 3 images will be shown for both logo and profile photo. In that case, the characteristics presented will be the following:
- The logo of each user's company: There are 3 possible logos. One of them will be partly shown and both the others will be shown in small sizes.
 - The profile photo of each user: The same happens with the profile photo as the logo, 3 images will be shown.
 - The name of the users: The names will be separated by commas.
 - The name of each user's company or "(user)" as it was previously explained.
 - The subject of the message/conversation or "(no subject)" if it does not exist.
 - The date of the last message sent or received.
- 4) Conversation with 4 users (Figure 23): The same logic follows here. When there are 4 users in a conversation, then 4 logos and profile photos should be shown in this row. In such case, the characteristics are:
- The logo of each user's company: Now, there are 4 possible logos which will be shown in small sizes. Note that all the images will have the same size.
 - The profile photo of each user: The same happens with the profile photo, 4 images will be shown, all with the same small sizes.



Figure 13 - Splash Screen

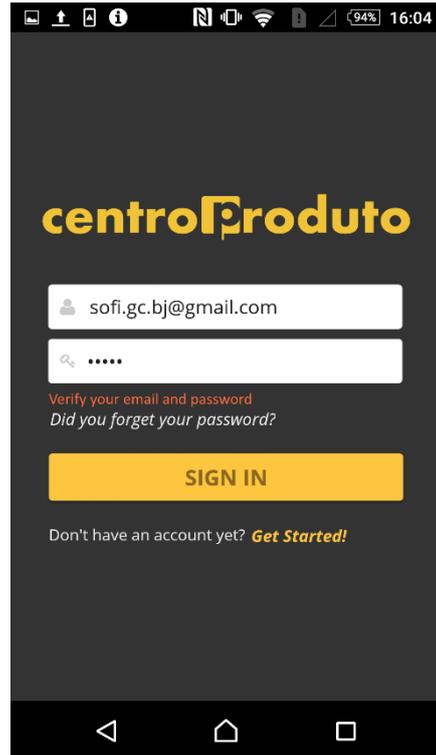


Figure 15 - Sign In - Verify email and password



Figure 14 - Sign In - Required Field (Password)

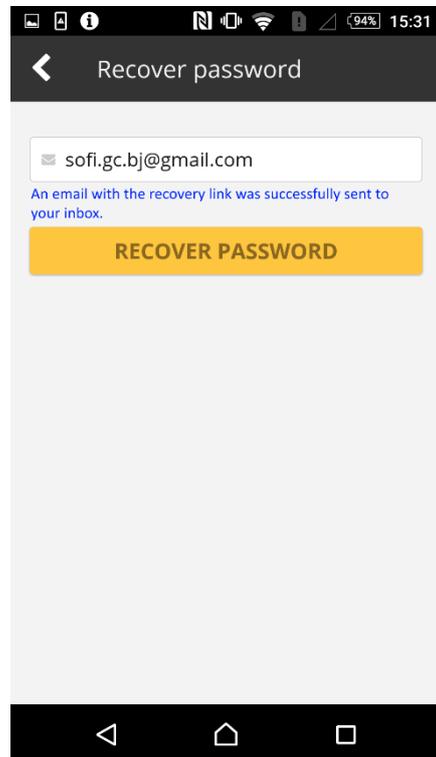


Figure 16 - Recover Password – Success



Figure 17 - Sign in



Figure 18 - Loading Bar

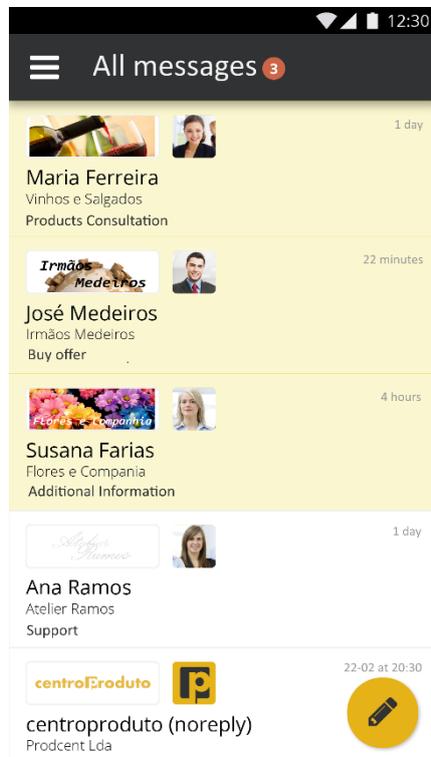


Figure 19 - List of Conversations Screen

- The name of the users: The names will be separated by commas.
 - The name of each user's company or "(user)" as it was previously explained.
 - The subject of the message/conversation: It can either show the subject if it exists or a message saying "(no subject)".
 - The date of the last message sent or received.
- 5) Conversation with more than 4 users (Figure 24): This case is most similar to the previous one, the difference is that the conversation counts with more than 4 users. In this case, the characteristics are:
- The logos of each user's company: As there is not enough space in the layout to display all logos, i.e., each user's logo. The space available for logos was divided in 4 but only 3 spaces will be filled with the correspondent images. The remaining space will display how many more users are implicated in the conversation.
 - The profile photo of each user: The same happens with the profile photo, only 3 images will show and the remaining space will indicate how many more users are participating in the conversation as well.
 - The name of the users: The names will be separated by commas.
 - The name of each user's company or "(user)" as it was previously explained.
 - The subject of the message/conversation: It can either show the subject if it exists or a message saying "(no subject)".
 - The date of the last message sent or received.

Also, the user can click on a "pencil" button which redirects him/her to the website in order to create new messages or in the same screen exists a Menu button which will be explained hereinafter.

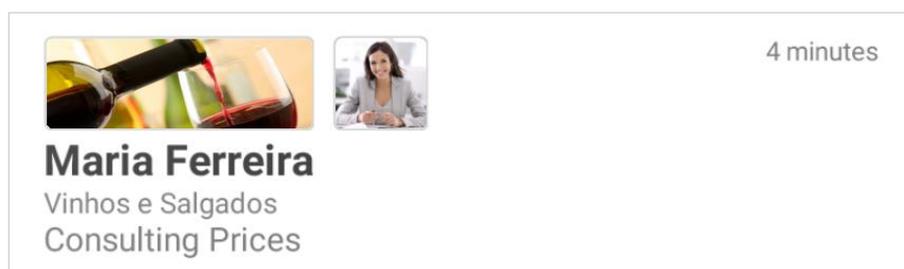


Figure 20 - Conversation - One user



Figure 21 - Conversation - Two users



Figure 22 - Conversation - Three users



Figure 23 - Conversation - Four users

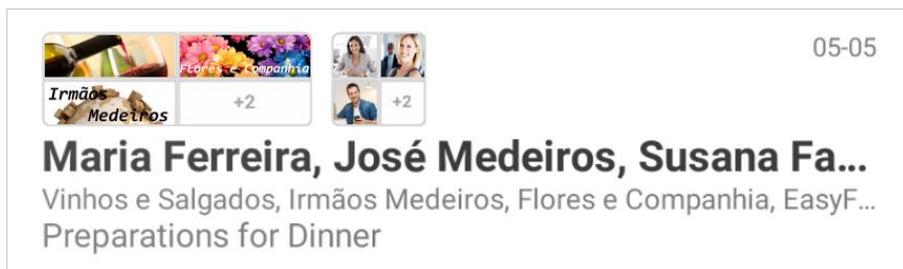


Figure 24 - Conversation - More than Four users

Also, the list of conversation must allow the user to perform a refresh action every time he/she wants. To do so, the user will have to swipe down the list and the refresh will be executed. During the refresh period, a circular loading bar will appear on top of the list (Figure 25).

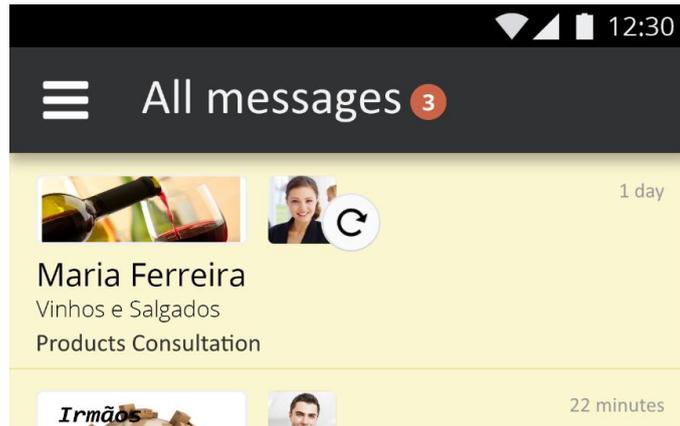


Figure 25 - Refresh

5.1.5. Drawer menu

The drawer menu is a menu that interacts with the conversation list screen in the following way:

To start, this menu opens by clicking on the Menu button or by sliding a finger from the left side of the screen, i.e. the messages list to the right side of the screen. To close it, the client should perform the same action in the inverse way (from right to left) or click on the right side of the screen where the list is shown in background (Figure 26).

The drawer menu shows, can display 4 different contents:

- a) When the client has not registered any company.
 Header: Contains the profile photo of the client and displays its name.
 Menu: Shows 6 items, which are: Create my Company, Messages, Sharing Center, Follow updates, Settings and Sign Out.
- b) When the client signs in and has a company and the company does not have a catalog.
 Header: Contains the logo of the user's company and his/her the profile photo. The name of the user and the company's name is displayed.
 Menu: Shows 8 items, which are: See my company, Create my catalog, Messages, Documents, Sharing Center, Follow updates, Settings and Sign Out.
- c) When the client has a company and the company has a catalog.
 Header: Contains the logo of the user's company and his/her the profile photo. The name of the user and the company's name is displayed.
 Menu: Shows 8 items, which are: See my company, See my catalog, Messages, Documents, Sharing Center, Follow updates, Settings and Sign Out.
- d) When the client has more than one company.

Header: The client will sign in as the default company (if any) or he/she will sign in using the last company as he/she was logged in. So the header will contain, the logo of the active company and the profile photo of the client. Moreover, the name of the client and his/her company will be displayed.

Menu: The menu will show 8 items from which one will depend on whether the company has a catalog or not. These are: See my company, Create my catalog or See my catalog, Messages, Documents, Sharing Center, Follow updates, Settings and Sign Out.

Every time the user changes companies, the logo of the company and the name displayed will change as well.

Whenever the client did not upload one (or both) of the images – user’s photo and / or logotype photo – to centroproduto a default image will be displayed instead (Figure 27). In any case, if the user clicks on any option that is not “Settings” or “Sign out”, he/she will be redirected to the browser, while if he/she clicks any of those 2 options the following can happen:

- a) Settings: The client is redirected to a new screen which will be presented in the following section (see section [5.1.6.](#)).
- b) Sign Out: This item simply performs the action of logging out of the system. When the client clicks on this item, the application returns to the Sign in screen until he/she insert its information again and gets authenticated.

5.1.6. Configuration of Settings

As it was referred previously (see [3.2.2.4.](#)), the client has to be able to realize 4 different actions (Figure 28):

- a) Change companies: The client has a dropdown with a list that contains the companies he/she registered on centroproduto (if any). If the user does not have a company, a message to create a company will be shown. Otherwise, if he/she has a company registered, he/she will be automatically logged with that company by default. If instead he/she has more than one company, one of 2 things can happen:
 - a. He/she will sign in as the default company he/she chose.
 - b. He/she will sign in as the last company he/she was logged in with. Note that this case will only work if the user did not chose a default company.

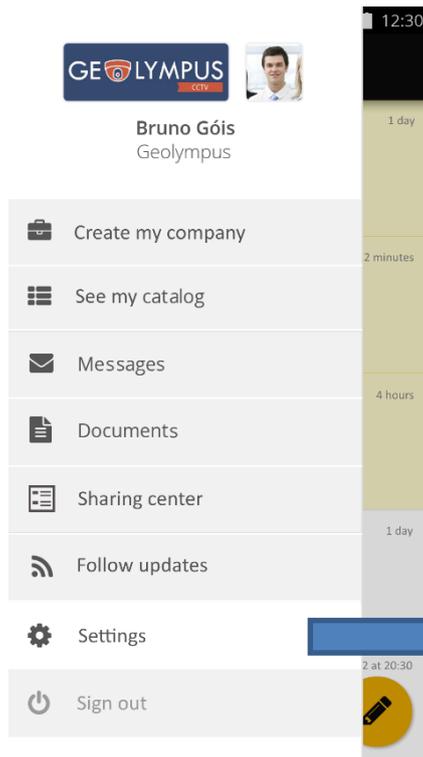


Figure 26 - Menu

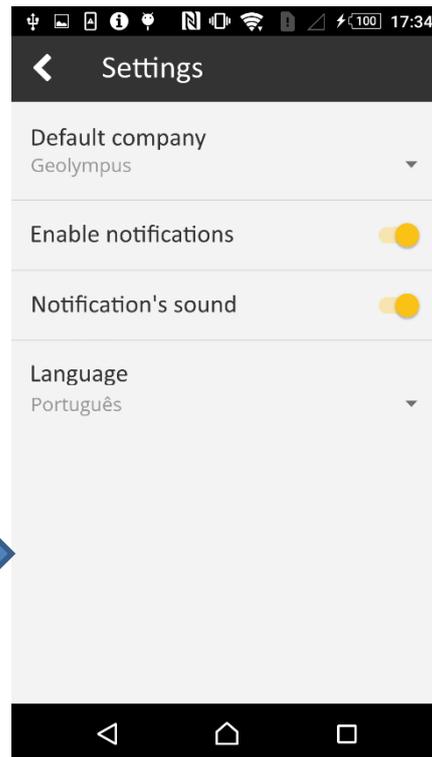


Figure 28 - Settings



Figure 27 - Default logo (Left) and default photo (Right)

- b) Enable/Disable Notifications: Even when the client is logged in the application, he/she will be able to decide if he/she wants to receive notifications or not. This includes Android notifications and the Chat Head button popping up on the screen when a notification is received.
- c) Enable/Disable Sounds: The client can decide if he/she want to hear any kind of sound when a notification is received.
- d) Change the app's language: Currently, centroproduto supports 7 different languages (English, Portuguese, Spanish, French, Italian, Romanian and Russian) which are going to be supported by the mobile application as well. In such case, the user will be allowed to can change the language of the mobile application by selecting any of those languages item of the languages list. Note that at first the application uses the mobile's language to select the language of the application. If the application does not support the language of the mobile, the application will be installed using the default language, which is English.

5.1.7. Chat Head

Previously, it was mentioned that the Chat Head as Facebook named it [78], is a notification method that will be used by this application when receiving notifications.

This notification methodology works in the following way: if the client receives a notification, the Chat Head will pop-up on the front of the current screen. This way, the user will be interrupted and notified about new messages in the system. The Chat Head is characterized by its appearance. It looks like a round circle with the short version of centroproduto's logo. And when it opens, it will show the total number of notifications received (Figure 29). There are two possible interactions that the client can perform with it:

- 1) By clicking on the Chat Head, the user can open the notification list, i.e. the main view of the mobile application.
- 2) A notification will appear at the same time the Chat Heads pop-up but instead of popping-up on the front of the screen, it will be listed at the notifications section of the mobile system. This notification will show the short logo of centroproduto, the name of the application and a message saying that there are new messages.

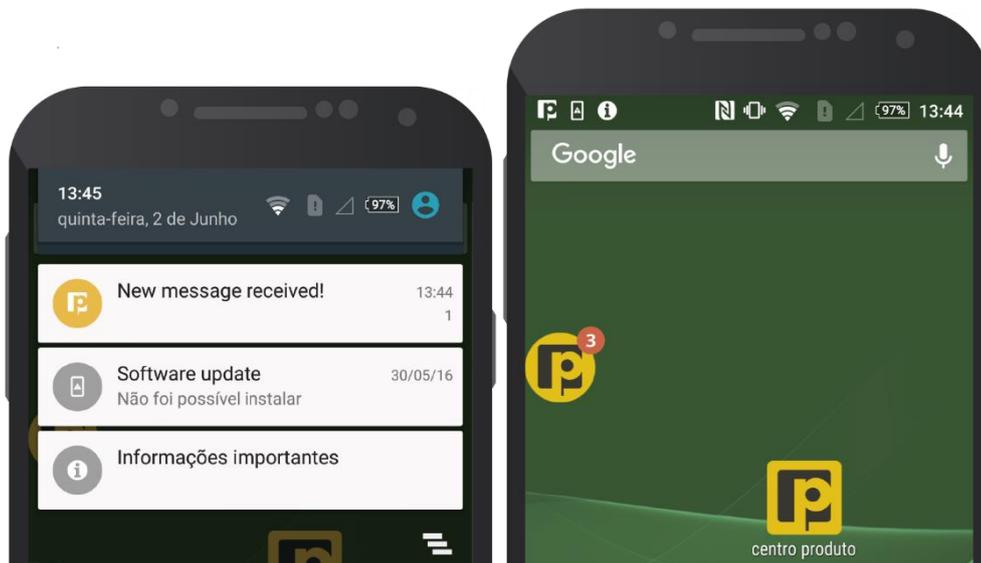


Figure 29 - Notification (Left) and Chat Head (Right)

5.1.8. Conversation

The conversation is what most people call a chat, it shows all the messages received and also sent, the user (or users) involved in the conversation, the subject if any, and whether the user (or users)

are sending messages as a user (or as users) or as company. It is important to refer that depending on the type of the message received, the client will be able or not to reply the message. As example, it can be mentioned the informative messages sent from centroproduto to its clients which cannot be replied (Figure 30).

As mentioned above, the conversation may or not have a subject. If it does, it will be presented under the main toolbar and can be differentiated by its gray background. If the subject occupies a larger space than the one available in the screen, then the subject will be shown partially (Figure 30) and a button will be shown that will allow the user to read the whole subject (Figure 31).

With respect to the list of messages contained in the conversation. They will be ordered depending on the type of conversation, i.e. if the conversation is informative the messages will start appearing at the top but eventually they will be presented from bottom to top, starting with the latest message (Figure 30). Otherwise, the conversation will be presented from bottom to top, starting with the most recent message (sent or received) on the bottom and in either case the messages will be ordered by date/time (Figure 32). Moreover, the messages in a conversation will have 2 different formats, depending on whether a message was sent or received.

- 1) Message received (Figure 33): Each message will show the profile photo of the user that sent the message rounded by a gray stroke, the message sent by that same user on a text view which uses a 9-patched image as background. The image has a white background and is surrounded by a gray stroke as well (see section [5.1.1.](#)). Under the message, the date/time of it will be shown.
- 2) Message sent (Figure 34): This type of message instead, does not show a profile photo. Also the 9-patched image used as background of the message uses a different set of colors. The background color is a light yellow surrounded by a darker yellow stroke (see section [5.1.1.](#)). As the message received, the message sent also shows the date/time.

The 9-patch, which was mentioned above, it is a tool that allows developers to create images that can change sizes automatically to fit contents and/or spaces in a view [79]. These images have to be saved in a special format which is “*.9.png”, and the parts that are allowed to be rescaled have to be selected. These parts are only selected horizontally and vertically, since the images can only expand in both those ways.

Other case to consider is replying to messages. To allow the user to reply, the application must consider a few things:

- 1) Whether the user is participating in a conversation as user: he/she will be able to reply to the message without trouble.
- 2) Whether the user is participating in a conversation with more than one company: the user will be able to reply at any time, but he/she will be able to select whether he/she wants to reply as one company or another.
- 3) Whether the user is participating in a conversation as user and as company: the user will be able to reply at any time, but he/she will only answer the message as the one (user or company) with which the conversation was created. If he/she created the conversation as company and is not logged with that company, he/she will be obliged to change the default company to reply.



Figure 30 - Informative Message – Short Subject



Figure 31 - Subject Extended

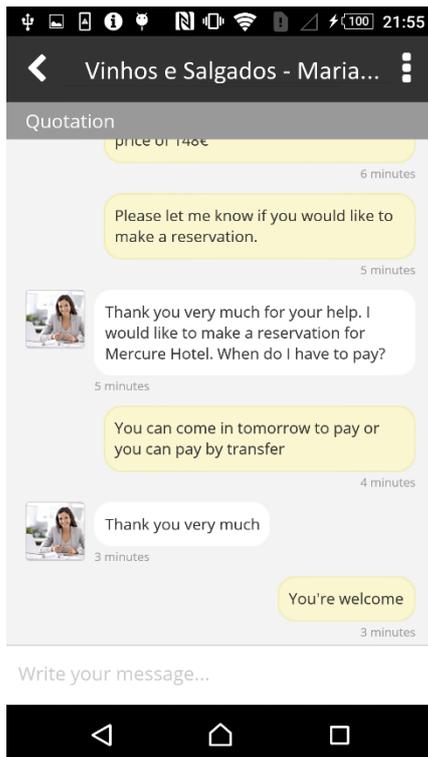


Figure 32 - Message List

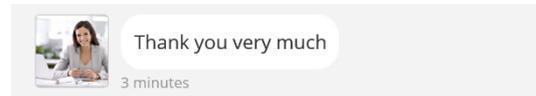


Figure 33 - Message Received

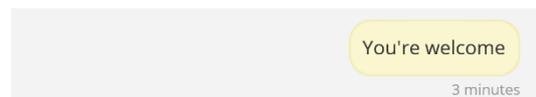


Figure 34 - Message Sent

- 4) Whether the user is not logged as the company involved in the conversation: the space for writing the message will be gone, and a new layout asking the user to change companies will appear instead (Figure 35). On the contrary, if the client is already logged as the company he/she will be able to answer by clicking where the message “Write your message here...” is displayed (Figure 32).

Aside from all these possibilities in a conversation, the user will be able to select more options in the conversation. By clicking on the menu button the user will see a dropdown list containing three options (Figure 36):

- a) See User’s List: If he/she chooses to visualize the list, all the users that participate in the conversation will be listed in a new screen.
- b) Mark as Unread: The client will be redirected to the notifications list, i.e. to the main screen where all the conversations are listed, but the conversation that was marked as unread will be highlighted in a yellow color.
- c) Remove Message: The client will be redirected to the notification list but the conversation would have disappeared from the list.

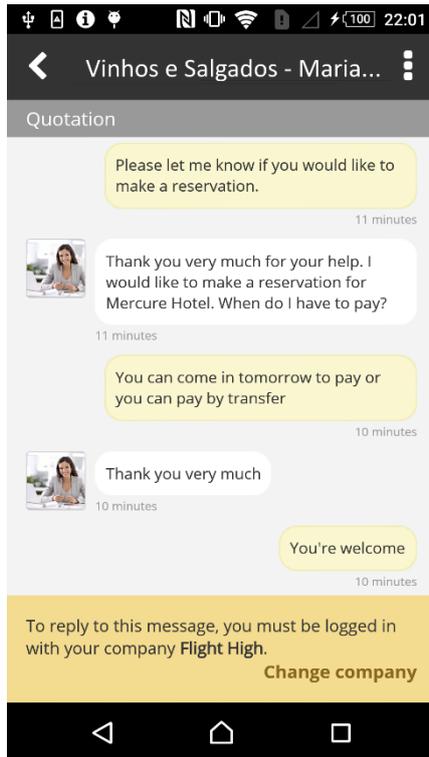


Figure 35 - Change Company

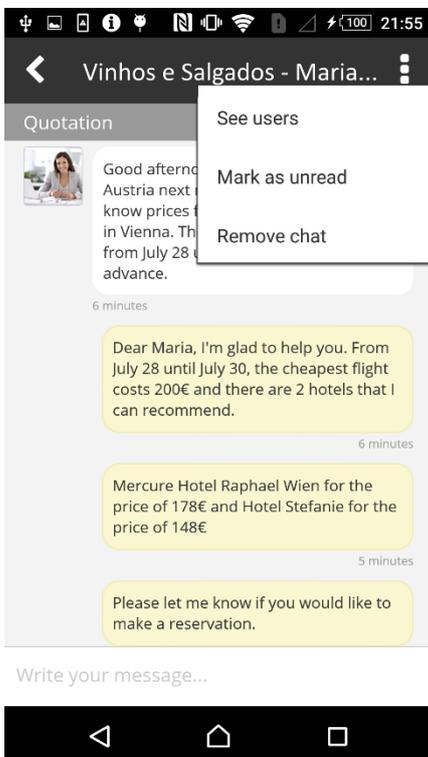


Figure 36 - Menu - Option



Figure 37 - User's list

5.1.9. Feedback on internet connection

A good application depends on internet connection to function properly should always provide feedback about connectivity to the users. In such case, this application will provide this type of information for 5 different cases. The first one is when the user is going to Sign in, the second is when the user is trying to visualize the list of conversations followed by the user trying to visualize the conversation, change companies and/or sends a message through a conversation. Moreover, this content will also appear when the user goes to the settings and when he/she tries to visualize the list of users of a conversation. In any of these cases, the application will have to detect when the connection is lost or established in order to show or hide that feedback message.

In the case of Sign in process if the connection is lost, i.e. when the user clicks to Sign in and there is no internet connection, the application shows a message to verify the internet connection, disables the login button and stops the process (Figure 38). However, if the connection is established, the application will be able to detect it and hide the message.



Figure 38 - Sign In - Connection Lost

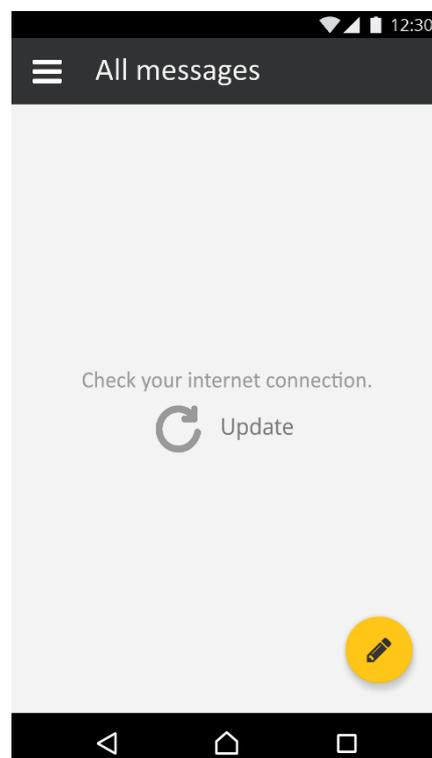


Figure 39 - List of Conversations - No connection

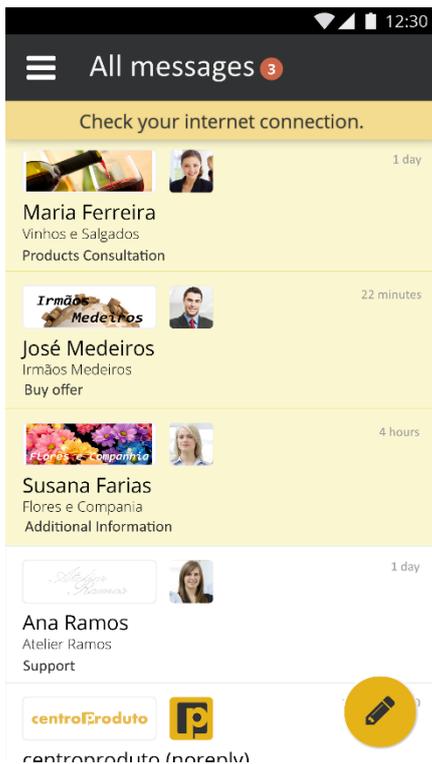


Figure 40 - List of Conversations - Connection Lost

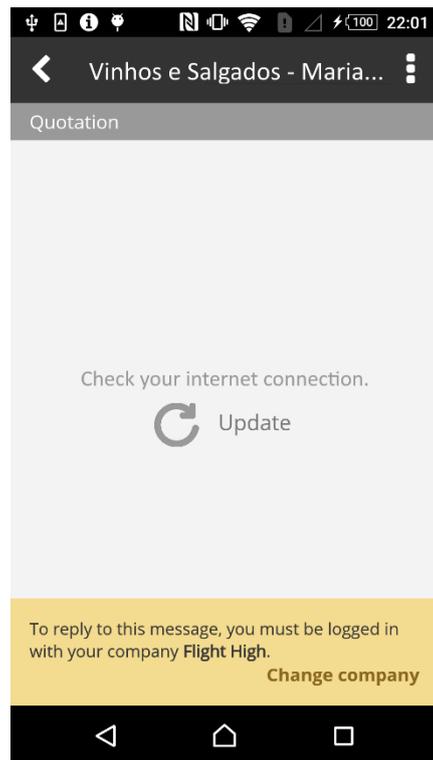


Figure 41 - Conversation - No connection

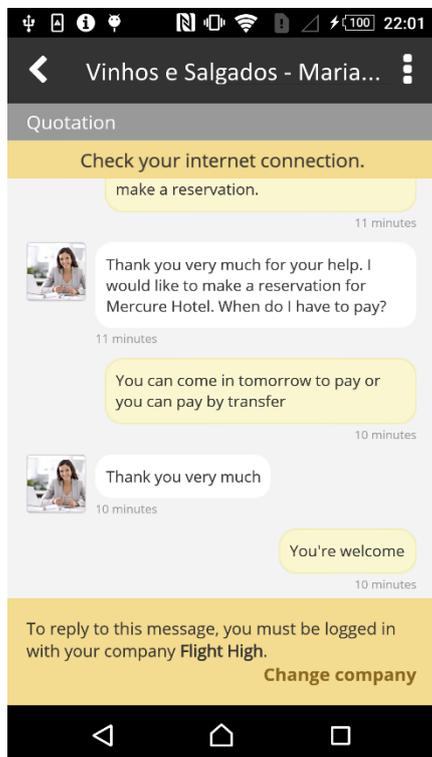


Figure 42 - Conversation - Connection lost

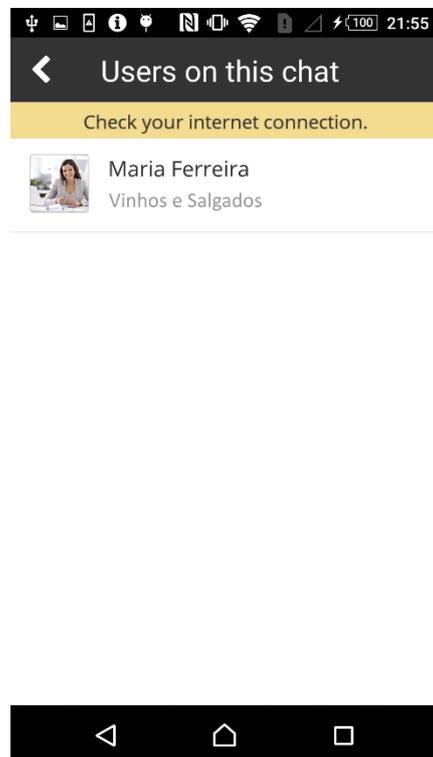


Figure 43 - List of users - Connection lost

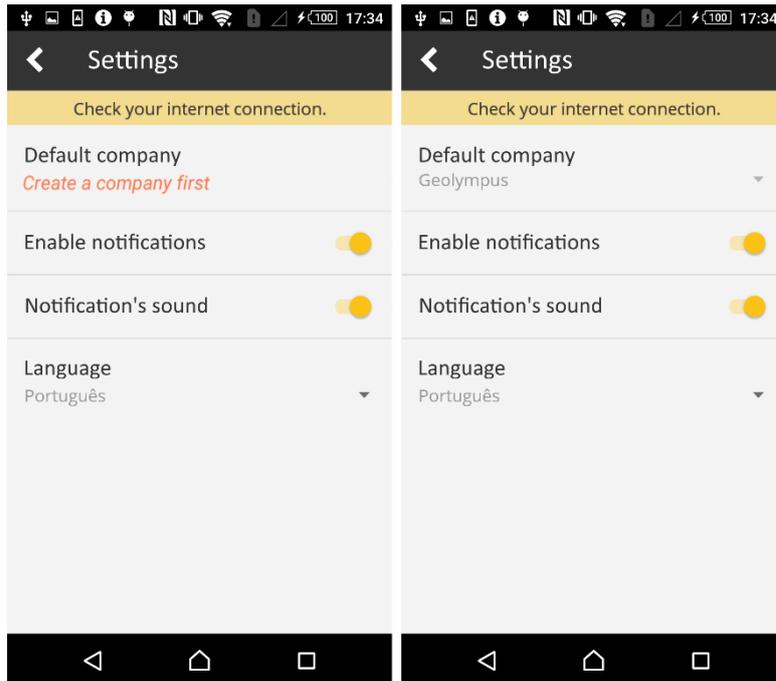


Figure 44 - Settings - Connection lost - Left: No company, Right: List of companies

In the case of the list of conversations, two things can happen:

- 1) If the user opens the application and there is no internet connection, a message will show telling him/her to verify the internet connection and then to try refreshing the list (Figure 39).
- 2) If the user is already in the application and the connection is lost, a message will appear on top of the list of conversations, asking him/her to check the connection (Figure 40).

Considering the second point is true, if the tries to open a conversation, a message will appear saying there is no internet connection and then to try refreshing the list (Figure 41). If instead, the user is already in the conversation, i.e. the conversation is already opened and the connection is lost, a message will appear on top of the list of messages (Figure 42). Afterwards, if considered that there is no internet connection yet and the user pretends to visualize the list of users of a conversation a message to check the internet connection will appear (Figure 43), but this message will not interfere with visualizing the list.

Aside from the cases presented above, there is another case in which the loss of the internet connection can interfere. This case is presented when the user goes to settings and there is no internet connection. When this happens a message to check the internet connection will be visible. Moreover, whether the user has or not a company (Figure 44) registered, a certain opacity will be

applied to the Default company field, which means that he/she cannot change the company until there is internet connection

5.2. Collaboration with the Design team

The design team of centroproduto has the duty to design the identity of the company which implies that any design referring to the web portal or the new application of the company has to go through them. The design team aside from designing the application's look was responsible by providing all the necessary images, such as the logo of centroproduto and image files that are not included in the library previously mentioned in section [5.1.1](#), such as the 9-Patch files for the messages' background, and the code of the colors that are used in the entire prototype and the application. Also, this team was responsible of approving the look of each views of the application after being developed.

6. Validation

6.1. Unit testing with Espresso

To verify the correct performance of the application, it was necessary had to write some unit tests. To do so, Espresso [80] which is a testing framework provided by Android that also runs with AndroidJUnitRunner was used. Espresso synchronizes the actions to be performed by a function in order to run in perfect timing.

The implemented test functions are related to all the views presented in the previous section (see section [5.1.](#)) and were divided in different functions according to what is supposed to happen. See an example of these tests in Table 15. This example starts by starting the view where the login is going to be executed. Afterwards, it tests that the login process is executed as it is supposed (inserting email and password and clicking Sign in). Then, it tests that a loading bar is shown while the application is executing the logging process. Once the user is logged, the exert of code verifies that the list of conversations of the user is displayed on the screen. These tests run in any Android device where the API is supported by the mobile device. During the implementation of these tests 4 different devices where used:

- Android 5.1.1 – Sony Xperia Z3
- Android 5.0.2 – Samsung Galaxy Tab4
- Android 4.1.2 – Samsung Galaxy Note 2
- Android 4.3 – Huawei Ascend P7 mini

Table 15 - Exert of Espresso test

```
@Test
public void loginAndDisplayListConvTest() {

    /*Start the new activity*/
    mActivityRule.launchActivity(new Intent());

    /*Detect the view with R.id.emailfield id and type the email*/
    onView(allOf(withId(R.id.emailfield))).perform(typeText("sofi.gc.bj@gmail.com"));

    /*Detect the view with R.id.passwordfield id and type the password. Afterwards,
close the keyboard*/
    onView(allOf(withId(R.id.passwordfield))).perform(typeText("Testtest1"),
        closeSoftKeyboard());

    /*Detect the view with R.id.buttonlogin id and perform a click*/
```

```
onView(withId(R.id.buttonlogin)).perform(click());

/*Verify that the intent changes to show the progress bar*/
intended(hasComponent(new ComponentName(getTargetContext(), Loader.class)));

/*Verify a list of conversations is displayed*/
onData(instanceOf(ListConversations.class))
    .inAdapterView(withId(R.id.listnotifications))
    .atPosition(0)
    .check(matches(isDisplayed()));
}
```

After executing this test 10 times for each device the resulting average is shown in Figure 45.

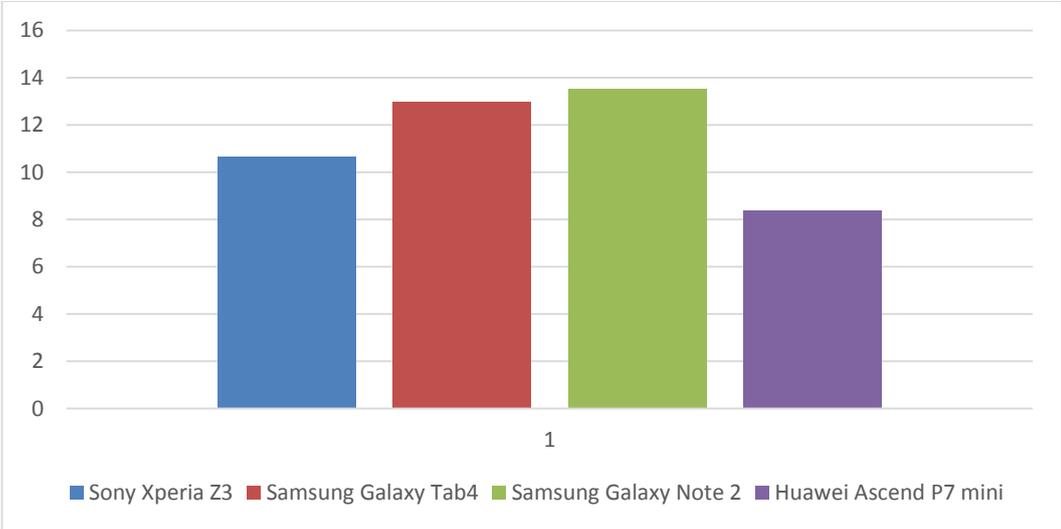


Figure 45 - Test results – Unit test

6.2. centroproduto’s Tests

centroproduto has staff dedicated to quality assurance of its software products. In the context of this project, it was possible to have the collaboration of this team to test the new mobile application workflows. The idea is to test as many times as possible the different scenarios to find out whether the application is executing as is supposed or not. Some of the results of these tests were:

- Menu button was not clickable.
- Application was allowing users to send URLs.
- Could not refresh the list of conversations.
- The unread messages counter was showing “0” instead of the number of messages to read.

Please note that these problems have already been addressed.

6.3. Pilot Users

To test the developed application in terms of the supported workflows and general ease of use, a questionnaire was created [81]. This questionnaire is targeted to people that had used centroproduto’s system before but did not know about the application. To answer this questionnaire, the user has to follow some steps through the application. The results obtained from the questionnaire are the following:

- Six people from age 17 and up answered the questionnaire.
- Six different mobile devices were used:
 - Alcatel Pop C7 7041x – Android 4.2
 - Sony Xperia M4 Aqua – Android 5.0
 - Sony Xperia Z3 – Android 5.1.1
 - Huawei Ascend G6 – Android 4.3
 - Samsung Galaxy Core Prime – Android 4.4.4
 - Huawei Ascend P7 Mini – Android – 4.3
- All of them were asked the level of difficulty to answer the questionnaire. The results are shown in Figure 46.
- The users were asked “Do you think the interface is simple and intuitive?”. All of them replied that it is, having a 100% of positive answers.
- The users were asked whether they think the application is useful or not and 100% of the users replied it is very useful.

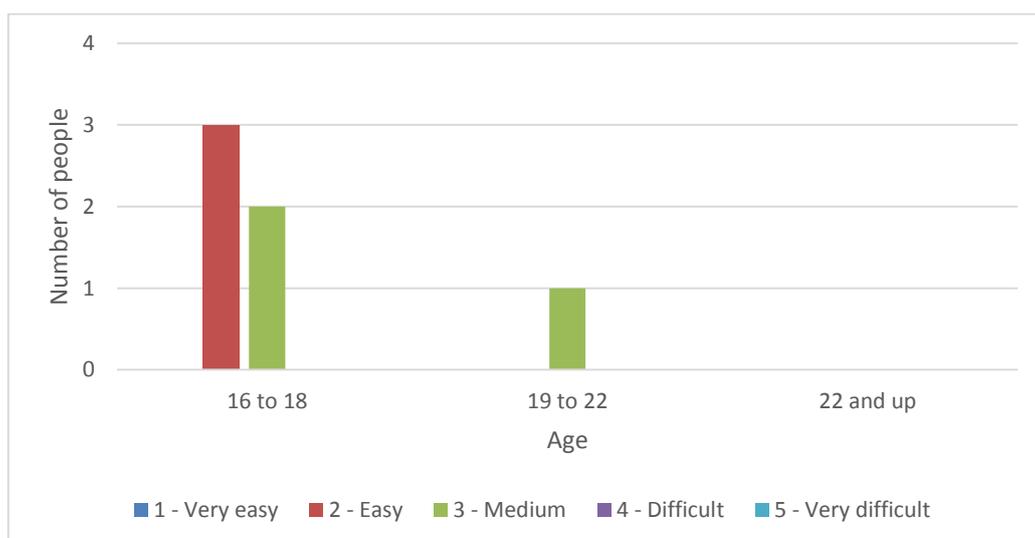


Figure 46 - Test results - Level of difficulties

- At the end they left some comments about the application where 3 of them expressed their discontent about being sent many times to the browser. And other 2, expressed the application is very easy to use and very efficient.

The reason why the users participating in this pilot are indeed young users is because the application deployment is not yet finalized and the involvement of end-users, representatives of business, was found to be too early. However, these young users are familiar with the system since they were collaborators at centroproduto.

7. Conclusion and Future Work

7.1. Achievements

centroproduto operates a Portal to facilitate online match-making of business partners, in which companies can announce their products and services, or look for possible suppliers to their businesses. Product and service catalogs can be promoted in the Portal, but the platform does not handle trading transactions, unlike web stores, acting specially as a digital forum for pursuing business contacts.

Before this work, centroproduto did not have any specific mobile application in its portfolio. The Portal provides, however, a responsive interface that adapts to the characteristics of client screens and can be browsed in mobile devices, but not the native and familiar user experience provided by mobile applications.

The main outcome of this work is a mobile application for the Android platform that provides an alternative channel for user interactions. The application is fully integrated with the existing back-end end, using a REST API that was enhanced with the requirements from this new scenario.

The application was intended to be used as a complementary channel by centroproduto and, for several use cases, it allows the users to start the interaction from the app, being redirected to the web portal for further/more complex actions.

The key feature in the mobile app is related to the presentation of notifications, alerting the users to new messages from ongoing/new business contacts. The Android notification system was central to its implementation, integrated with Amazon's push notifications service. The integration of a cloud push messaging infrastructure proved to be scalable and reliable.

The new application is being used by a pilot group of users with centroproduto, as beta testers, and in the phase of porting it to the market and making it available to all users.

The resulting product allows users to send messages, follow existing threads and get notification on new messages, without the need to open the browser or the email, with a coherent continuity with the existing portal, which were the essential centroproduto's requirements.

7.2. Lessons Learned

Mobile applications in the present day are very important. In companies, mobile applications help increasing the number of users engaged to the system. Moreover, they are simple, interactive and

can be used wherever the user is and whenever he/she wants. To develop an application in a company, first it is necessary to be able to work with a team.

Team-work can be considered the most important aspect of working in a company. It is crucial to move forward with a project. Each member of the team should be the most communicative as possible and should have the best relationship with other co-workers. Developing an application while interacting with a team can be helpful in many aspects, they can give advice and help.

Another important aspect is time management. Managing the time of a project's development is somehow tricky. It depends on the people involved in developing the project, the number of functionalities to develop and the limited time of development. To develop a project, it is necessary to link each one of these by scheduling how long the implementation of each functionality should last in order to meet the deadline.

In terms of development, it is very advantageous developing an application and using different components for the correct functioning of it. In fact, Android provides mechanisms to develop a simple, responsive and interactive mobile application. With Android it is possible to create list views, to use the phone's camera and to make an application play sounds. Aside from those mechanisms, there are other mechanisms that are not directly attach to Android but that work perfectly together. Amazon SNS and Picasso are two of them.

7.3. Future Work

For future work, it will be necessary to stabilize the application, i.e. solve the possible errors found before and after releasing the application. Afterwards, to adapt the application it will be necessary to implement other features in the application such as mark as read/unread and delete conversations from the list of conversations. This should be implemented by using a swipe mechanism that shows a button to mark as read unread on the left side of an item of the conversation list and a button to delete the conversation at the right side as well. Also, since centroproduto's application requires that the browser is opened many times, it could be considered a possibility to include in the mobile application a web view to open the website without sending the user out of the application.

Moreover, the idea of centroproduto is to provide an application to all their registered company representatives which means that it is necessary to develop an application like the one introduced in this dissertation that works for iPhone OS.

References

- [1] P. W. Shahar Markovitch, "Accelerating the digitalization of business processes," McKinsey & Company, May 2014. [Online]. Available: <http://www.mckinsey.com/business-functions/business-technology/our-insights/accelerating-the-digitization-of-business-processes>. [Accessed 11 July 2016].
- [2] centroproduto, "centroproduto," Prodcent Lda, [Online]. Available: <http://www.centroproduto.com>. [Accessed 06 October 2015].
- [3] centroproduto, "centroproduto mobile," centroproduto, [Online]. Available: <http://m.centroproduto.com>. [Accessed 06 October 2015].
- [4] M. Rosoff, "Facebook is officially a mobile-first company," 05 November 2015. [Online]. Available: <http://www.businessinsider.com/facebook-mobile-only-users-most-common-2015-11>. [Accessed 22 July 2016].
- [5] Jerry Dischler, Vice President, Product Management, AdWords, "Inside AdWords - Building for the next moment," Google, Inc., 05 May 2015. [Online]. Available: <https://adwords.googleblog.com/2015/05/building-for-next-moment.html>. [Accessed 22 July 2016].
- [6] "Facebook," Facebook, Inc., [Online]. Available: www.facebook.com. [Accessed 21 October 2015].
- [7] "LinkedIn," LinkedIn Corp, [Online]. Available: <http://www.linkedin.com>. [Accessed 21 October 2015].
- [8] "Alibaba.com," Alibaba Group, [Online]. Available: <http://www.alibaba.com/>. [Accessed 21 October 2015].
- [9] "Number of monthly active facebook users worldwide," Statista, [Online]. Available: <http://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/>. [Accessed 02 June 2016].
- [10] "Google Play," Google, Inc., [Online]. Available: <https://play.google.com/>. [Accessed 27 May 2016].
- [11] "Messenger," Facebook, Inc., [Online]. Available: <https://www.messenger.com/>. [Accessed 05 April 2016].
- [12] "Search - Chat Heads," Google, Inc., [Online]. Available: <https://play.google.com/store/search?q=chat+heads>. [Accessed 11 July 2016].

- [13] J. Greene, "Microsoft to Acquire LinkedIn for \$26.2 Billion," Wall Street Journal, 14 June 2016. [Online]. Available: <http://www.wsj.com/articles/microsoft-to-acquire-linkedin-in-deal-valued-at-26-2-billion-1465821523>. [Accessed 11 July 2016].
- [14] "Quarterly Numbers of LinkedIn Members," Statista, [Online]. Available: <http://www.statista.com/statistics/274050/quarterly-numbers-of-linkedin-members/>. [Accessed 02 June 2016].
- [15] "Google Play - LinkedIn," Google, Inc., [Online]. Available: <https://play.google.com/store/apps/dev?id=6860682062931868151>. [Accessed 22 July 2016].
- [16] A. L. Vikas SN, "Decoding Alibaba - The world largest e-commerce company," Economic Times, 09 October 2015. [Online]. Available: <http://tech.economictimes.indiatimes.com/news/internet/decoding-alibaba-the-worlds-largest-e-commerce-company/49290466>. [Accessed 02 June 2016].
- [17] "About Alibaba," Alibaba Group, [Online]. Available: <http://www.alibabagroup.com/en/about/overview>. [Accessed 02 June 2016].
- [18] A. Group, "Taobao," Alibaba, [Online]. Available: <http://www.taobao.com>. [Accessed 02 June 2016].
- [19] A. Group, "Tmall," Alibaba, [Online]. Available: <http://www.tmall.com/>. [Accessed 02 June 2016].
- [20] A. Group, "AliExpress," Alibaba, [Online]. Available: <http://www.aliexpress.com/>. [Accessed 02 June 2016].
- [21] "Alibaba cumulative active online buyers," Statista, [Online]. Available: <http://www.statista.com/statistics/226927/alibaba-cumulative-active-online-buyers-taobao-tmall/>. [Accessed 02 June 2016].
- [22] "About AliExpress," Alibaba, [Online]. Available: <http://activities.aliexpress.com/adcms/help-aliexpress-com/about.php?spm=2114.11010108.0.105.NJZ015>. [Accessed 02 June 2016].
- [23] "Search - Alibaba," Alibaba Group, [Online]. Available: https://play.google.com/store/apps/details?id=com.alibaba.icbu.app.seller&hl=pt_PT. [Accessed 02 June 2016].
- [24] A. Rodriguez, "developerWorks - RESTful Web services: The basics," International Business Machines Corporation, 09 February 2015. [Online]. Available: <https://www.ibm.com/developerworks/library/ws-restful/>. [Accessed 24 July 2016].

- [25] "Android release date," Google, Inc., [Online]. Available: https://www.google.pt/search?q=about+android+release+date&ie=utf-8&oe=utf-8&client=firefox-b-ab&gfe_rd=cr&ei=LHxQV_e_EOar8wfgwJTQDw#safe=off&q=android+release+date. [Accessed 02 June 2016].
- [26] "Android Open Source Project," Google, Inc., [Online]. Available: <https://source.android.com/>. [Accessed 02 June 2016].
- [27] J. Vincent, "The Verge," Vox Media Inc., 29 September 2015. [Online]. Available: <http://www.theverge.com/2015/9/29/9409071/google-android-stats-users-downloads-sales>. [Accessed 02 June 2016].
- [28] "Number of available applications in the Google Play store," Statista Inc., [Online]. Available: <http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/>. [Accessed 12 November 2015].
- [29] "Android Developers - Get started with publishing," Google Inc., [Online]. Available: <https://developer.android.com/distribute/googleplay/start.html>. [Accessed 02 June 2016].
- [30] "Android Developer - Developer Console," Google Inc., [Online]. Available: <https://developer.android.com/distribute/googleplay/developer-console.html>. [Accessed 02 June 2016].
- [31] "Android Developer - Uses SDK element," Google, Inc., [Online]. Available: <https://developer.android.com/guide/topics/manifest/uses-sdk-element.html>. [Accessed 04 June 2016].
- [32] "Android Developer," Google, Inc., [Online]. Available: <https://developer.android.com/>. [Accessed 14 March 2016].
- [33] "Android Developer - Screens support," Google Inc., [Online]. Available: https://developer.android.com/guide/practices/screens_support.html. [Accessed 02 June 2016].
- [34] "Android Push Notifications," Parse, Inc., [Online]. Available: <https://parse.com/tutorials/android-push-notifications>. [Accessed 09 July 2016].
- [35] "Google Cloud Messaging (GCM)," Google Inc., [Online]. Available: <https://developers.google.com/cloud-messaging/>. [Accessed 04 June 2016].
- [36] "Google," Google Inc., [Online]. Available: <https://www.google.com/>. [Accessed 06 October 2015].

- [37] "Google Cloud Messaging - HTTP Connection Server Reference," Google Inc., [Online]. Available: <https://developers.google.com/cloud-messaging/http-server-ref>. [Accessed 02 June 2016].
- [38] G. Developers, "Google Cloud Messaging - Architectural overview," Google Inc., [Online]. Available: <https://developers.google.com/cloud-messaging/gcm#arch>. [Accessed 02 June 2016].
- [39] "Google Cloud Messaging - Keeping the Registration State in Sync," Google Inc., [Online]. Available: <https://developers.google.com/cloud-messaging/registration#keeping-the-registration-state-in-sync>. [Accessed 04 June 2016].
- [40] "MQTT," [Online]. Available: <http://mqtt.org/>. [Accessed 09 July 2016].
- [41] L. Zhang, "Facebook," Facebook Inc, 12 August 2011. [Online]. Available: <https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920>. [Accessed 02 June 2016].
- [42] "IBM Knowledge Center - Qualities of service provided by an MQTT client," IBM Corporation, 23 April 2016. [Online]. Available: https://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q029090_.htm. [Accessed 02 June 2016].
- [43] "HiveMQ - MQTT essentials Part 6 - MQTT Quality of Service levels," dc-square GmbH, [Online]. Available: <http://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels>. [Accessed 09 July 2016].
- [44] E. Nava, "Pushy a new alternative to Google Cloud Messaging," Elad Nava, 10 January 2015. [Online]. Available: <https://eladnava.com/pushy-a-new-alternative-to-google-cloud-messaging/>. [Accessed 02 June 2016].
- [45] "Pushy - Support," Pushy, [Online]. Available: <https://pushy.me/support>. [Accessed 02 June 2016].
- [46] P. Developers, "Pushy - Pricing," Pushy, [Online]. Available: <https://pushy.me/pricing>. [Accessed 02 June 2016].
- [47] "Parse - Migration," Parse, [Online]. Available: <https://parse.com/migration>. [Accessed 02 June 2016].
- [48] "Amazon Web Services - Amazon Simple Notification Service," Amazon.com Inc., [Online]. Available: <https://aws.amazon.com/sns/>. [Accessed 29 May 2016].

- [49] "Amazon Web Services - SNS Mobile Push," Amazon.com Inc., [Online]. Available: <http://docs.aws.amazon.com/sns/latest/dg/SNSMobilePush.html>. [Accessed 02 June 2016].
- [50] "Amazon Web Services - Mobile Push ADM," Amazon.com, Inc., [Online]. Available: <http://docs.aws.amazon.com/sns/latest/dg/mobile-push-adm.html>. [Accessed 09 July 2016].
- [51] "iOS Developer Library - Apple Push Service," Apple, Inc., [Online]. Available: <https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html>. [Accessed 09 July 2016].
- [52] A. Developers, "Amazon Web Services - Getting Started with Baidu Cloud Push," Amazon.com, Inc., [Online]. Available: <http://docs.aws.amazon.com/sns/latest/dg/mobile-push-baidu.html>. [Accessed 09 July 2016].
- [53] A. Developers, "Amazon Web Services - Getting Started with MPNS," Amazon.com, Inc., [Online]. Available: <http://docs.aws.amazon.com/sns/latest/dg/mobile-push-mpns.html>. [Accessed 09 July 2016].
- [54] "Developers resources - Tiles and notifications windows push notification services WNS Overview," Microsoft Corporation, [Online]. Available: <https://msdn.microsoft.com/en-us/windows/uwp/controls-and-patterns/tiles-and-notifications-windows-push-notification-services--wns--overview>. [Accessed 09 July 2016].
- [55] "Amazon Web Services - SNS FAQs," Amazon.com Inc., [Online]. Available: https://aws.amazon.com/sns/faqs/?nc1=h_ls. [Accessed 09 July 2016].
- [56] "Amazon Web Services - AWS Management Console," Amazon.com Inc., [Online]. Available: <https://aws.amazon.com/console/>. [Accessed 02 June 2016].
- [57] "Amazon Web Services - Pricing," Amazon.com Inc., [Online]. Available: <https://aws.amazon.com/sns/pricing/>. [Accessed 02 June 2016].
- [58] "Android Developers - Loading Large Bitmaps Efficiently," Google Inc., [Online]. Available: <https://developer.android.com/training/displaying-bitmaps/load-bitmap.html>. [Accessed 09 July 2016].
- [59] "GitHub - Picasso," Square Open Source, [Online]. Available: <http://square.github.io/picasso/>. [Accessed 04 June 2016].

- [60] "Android Developers - Bitmap.Config," Google Inc., [Online]. Available: <http://developer.android.com/reference/android/graphics/Bitmap.Config.html>. [Accessed 03 June 2016].
- [61] D. Furiya, "Github - Picasso Transformations," Wasabeef, [Online]. Available: <https://github.com/wasabeef/picasso-transformations>. [Accessed 03 June 2016].
- [62] S. Judd, "GitHub - Glide," Sam Judd, [Online]. Available: <https://github.com/bumptech/glide>. [Accessed 03 June 2016].
- [63] T. Nicholas, "Facebook Code - Introducing Fresco," Facebook Inc., [Online]. Available: <https://code.facebook.com/posts/366199913563917/introducing-fresco-a-new-image-library-for-android/>. [Accessed 03 June 2016].
- [64] S. Tarasevich, "GitHub - Android Universal Image Loader (UIL)," Sergey Tarasevich, 27 November 2011. [Online]. Available: <https://github.com/nostra13/Android-Universal-Image-Loader/commits/master?page=30>. [Accessed 03 June 2016].
- [65] s. Developers, "Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options," Salesforce.com, inc., June 2016. [Online]. Available: https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options. [Accessed 11 July 2016].
- [66] D. Crockford, "JSON," [Online]. Available: <http://www.json.org/>. [Accessed 04 June 2016].
- [67] A. Developers, "Android Developers - HttpURLConnection," Google Inc., [Online]. Available: <https://developer.android.com/reference/java/net/HttpURLConnection.html>. [Accessed 30 June 2016].
- [68] "Android Developers - Dashboards," Google Inc., [Online]. Available: <https://developer.android.com/about/dashboards/index.html>. [Accessed 02 June 2016].
- [69] "Android Developers - Display Metrics," Google Inc., [Online]. Available: <https://developer.android.com/reference/android/util/DisplayMetrics.html>. [Accessed 29 May 2016].
- [70] "Amazon Web Services - Getting Started with Google Cloud Messaging for Android," Amazon.com Inc., [Online]. Available: <http://docs.aws.amazon.com/sns/latest/dg/mobile-push-gcm.html>. [Accessed 04 June 2016].
- [71] "Google Developers Console - Projects," Google Inc., [Online]. Available: <https://console.developers.google.com/iam-admin/projects>. [Accessed 04 June 2016].

- [72] "Amazon Developers - Boto 3 Docs," Amazon.com, Inc., [Online]. Available: <http://boto3.readthedocs.io/en/latest/reference/services/sns.html>. [Accessed 04 June 2016].
- [73] A. Developers, "Android Developers - View.ShadowBuilder," Google Inc., [Online]. Available: <https://developer.android.com/reference/android/view/View.DragShadowBuilder.html>. [Accessed 2016].
- [74] "Android Developers - RecyclerView.ViewHolder," Google Inc., [Online]. Available: <https://developer.android.com/reference/android/support/v7/widget/RecyclerView.ViewHolder.html>. [Accessed 04 June 2016].
- [75] D. C. Division, "Interaction Design Basics," U.S. Department of Health and Services - Digital Communications Division, [Online]. Available: <https://www.usability.gov/what-and-why/interaction-design.html>. [Accessed 11 July 2016].
- [76] "Google - Layout Principles," Google Inc., [Online]. Available: <https://www.google.com/design/spec/layout/principles.html#principles-how-paper-works>. [Accessed 03 June 2016].
- [77] D. Gandy, "IcoMoon," Roonas, [Online]. Available: <https://icomoon.io/app/#/select/library>. [Accessed 03 June 2016].
- [78] "Facebook - Help Center," Facebook Inc., [Online]. Available: <https://www.facebook.com/help/android-app/101495056700254?rdrhc>. [Accessed 06 June 2016].
- [79] "Android Developers - Draw 9-patch," Google Inc., [Online]. Available: <https://developer.android.com/studio/write/draw9patch.html>. [Accessed 29 May 2016].
- [80] "Android Developers - Testing UI for a Single App with Espresso," Google Inc., [Online]. Available: <https://developer.android.com/training/testing/ui-testing/espresso-testing.html>. [Accessed 02 June 2016].
- [81] A. S. Góis, "Google Forms," Google, Inc., 06 June 2016. [Online]. Available: <http://goo.gl/forms/mnWXthOwQ4Hx7LKV2>. [Accessed 06 June 2016].
- [82] JUnit, "JUnit," [Online]. Available: <http://junit.org/>. [Accessed 2016].
- [83] U. P. & T. Office, "US Patent & Trademark Office," [Online]. Available: <http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&p=1&u=%2Fnetacgi%2FPTO%2Fsearch->

bool.html&r=0&f=S&l=50&TERM1=Facebook&FIELD1=AANM&co1=AND&TERM2=&FIELD2=&d=PG01. [Accessed 2015 - 2016].

- [84] A. Developers, "Android," Android, [Online]. Available: <https://www.android.com/history/#/marshmallow>. [Accessed 2015 - 2016].
- [85] A. Developers, "Android Developers," Google Inc., [Online]. Available: <http://developer.android.com/reference/android/graphics/Bitmap.Config.html>. [Accessed 2016].
- [86] Oracle, "Oracle Docs," Oracle Corporation, [Online]. Available: <http://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html>. [Accessed 2016].
- [87] K. Fakhroutdinov, "Unified Modeling Language (UML)," OMG UML, [Online]. Available: <http://www.uml-diagrams.org/>. [Accessed 2015].
- [88] I. J. G. B. James Rumbaugh, The Unified Modeling Language, Reference Manual, Second Edition, Addison-Wesley, 2005.