TIAGO FILIPE MODESTO FORNELOS
Integrated Master in Computer Science and Engineering

MULTIMODAL INTERACTION TECHNIQUES FOR CHATBOTS

COMPUTER SCIENCE

NOVA University Lisbon
November, 2021

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
COMPUTER SCIENCE

MULTIMODAL INTERACTION TECHNIQUES FOR CHATBOTS

TIAGO FILIPE MODESTO FORNELOS

Integrated Master in Computer Science and Engineering

Adviser: Nuno Manuel Robalo Correia
Full Professor, NOVA University of Lisbon

Co-adviser: João Miguel da Costa Magalhães
Associate Professor, NOVA University of Lisbon

COMPUTER SCIENCE

NOVA University Lisbon
November, 2021

Multimodal Interaction Techniques for Chatbots

Acknowledgements

Throughout the writing of this dissertation I had a great deal of support and assistance. Firstly, I would like to thank my supervisor and co-supervisor, Professor Nuno Correia and Professor João Magalhães, for their dedicated support and guidance, always willing to assist in any way they could throughout this process.

I'm also very thankful for being part of the Faculdade de Ciências e Tecnologias community, where I felt welcome since the first day I started my graduation, giving me the formation needed to be prepared for the future that awaits me.

I couldn't forget to thank my family for being there for me and supporting me during the compilation of this dissertation.

And last, but someone I could never forget to mention, my girlfriend Inês Leão. For you, a special thanks for all the love and support, for all the advices and for being there when I most needed it.

Abstract

Chatbots have become popular and are being used in the most diverse areas. The most common use for chatbot are the ones integrated in our smartphones, like Google Assistant and Siri. This kind of technology evolved significantly in the past few years and now the users have the feeling to be talking to another human being. In a near future, most of the interaction between large organisations and their users will be mediated by AI agents.

A market that benefited a lot from this arising technology are the e-commerce websites. Sometimes, it can be a struggle to find a product in a web site due to bad interface designs. When integrated in these websites, the clients have immediate access to a seller that is able to help them find the product they want or clarify some doubts that they might have.

Despite the great evolution in the responsiveness of these systems, when talking about the User Interface (UI) of the systems, there are no major changes. Normally, chatbots are represented using the common chat box in the bottom right corner of the screen.

In the scope of iFetch project, that the main focus is revolutionise the chatbot assistance in online sales, it is proposed for this dissertation the development of a user interface for a chatbot with multimodal interaction, making the user experience more interactive and engaging. This interface is a representation of a virtual room where the user can see the products that he/she wants to buy. The is also a component of image processing to be able to use the 2D images for the generation of their 3D representation.

Keywords: Chatbot, Multimodal Interaction, User Interface, E-commerce, 3D Environments

Resumo

Atualmente, a utilização de chatbots alargou-se para as mais diversas áreas. O exemplo mais comum são os chatbots integrados nos smartphones como o Google assistant ou a Siri. Esta tecnologia tem evoluído significativamente nos últimos anos, dando a sensação ao utilizador que está a falar com outro ser humano. Num futuro próximo, a maioria das interações entre os clientes e as empresas será feito através de agentes AI.

Uma área que tem beneficiado muito com a introdução desta tecnologia, são os websites de vendas online. Por vezes, poderá ser complicado um utilizador encontrar o produto que pretende no site da loja, devido ao mau design da interface. Assim, os chatbot permitem que os seus clientes tenham acesso rápido a um vendedor a qualquer momento, que os ajudará a encontrar o artigo pretendido e tirar possíveis dúvidas que tenham.

Apesar de ser notória uma grande evolução na capacidade de resposta destes sistema, no que toca a interface digital dos chatbots, não se tem notado grande inovação. A maioria dos chatbots que encontramos hoje em dia, são a típica caixa de texto que se encontra no canto inferior do ecrã.

Assim, dentro do contexto do projecto iFetch, que pretende desenvolver um chatbot para a assistência no mundo das vendas online, surge este trabalho cujo objetivo passa por desenvolver uma interface inovadora multimodal que permita tornar a experiência dos utilizadores do website mais interativa. Esta interface consistirá numa loja virtual onde o utilizador poderá visualizar os produtos que pretende comprar. Existe também uma componente de processamento de imagem para que seja possível a visualização das imagens 2D em manequins 3D.

Palavras-chave: Chatbot, Interação Multimodal, Interface Utilizador, Vendas Online, Ambiente 3D

Contents

List of Figures **x**

List of Tables **xiii**

Acronyms **xiv**

1 Introduction **1**

1.1 Motivation 1

1.2 Context and Description 2

1.3 Main Objective 2

1.4 Contributions 3

1.5 Document Structure 3

2 Related Work **5**

2.1 Chatbots in the Current Days 5

2.1.1 Usability Heuristics of Chatbots 5

2.1.2 The 6 Norman Design Principles 6

2.1.3 Chatbots User Experience 7

2.1.4 Chatbots Interface 8

2.1.5 Chatbots in Online Sales 10

2.1.6 Chatbot Implementation Techniques 10

2.2 3D Modelling from 2D Images 11

2.2.1 Pix2Surf 11

2.2.2 Style.me 12

2.3 Convolutional Neural Networks 13

2.3.1 How do CNNs Work? 13

2.4 Background Removal 16

2.4.1 Semi-Automated Tool 16

2.4.2 Morphological Operations 16

2.4.3 Using CNN 18

2.5 Clothes Retrieval in Online Shopping 19
2.6 Virtual Shopping Environments 20
2.7 Discussion 22

3 Application Design **23**
3.1 Application Concept 23
3.2 Application Requirements 26
3.3 Technologies 27

4 Implementation **29**
4.1 Interface Features 29
4.2 Environment Setup - Web Server 36
4.2.1 Endpoints 36
4.2.2 JSON Structure 37
4.3 Clothing Models 39
4.3.1 Pix2Surf Setup Requirements 40
4.3.2 Re-scale Front Image 41
4.3.3 Generating Back Image 41
4.4 Background Removal 48
4.4.1 GrabCut 48
4.4.2 Morphological Operations 48
4.4.3 Convolutional Neural Networks 48
4.4.4 Methods Comparison 49
4.5 System Integration 49
4.5.1 Backend of Chatbot 49
4.5.2 Images Dataset 52
4.6 Android Application 53

5 Evaluation and Results **54**
5.1 System Performance Tests 54
5.1.1 Number of Draw Calls 55
5.1.2 Memory Usage 57
5.2 Background Removal Performance Tests 58
5.3 Preliminary User Test 58
5.3.1 Participants and Evaluation Methods 59
5.4 Final User Tests 59
5.4.1 Participants and Evaluation Methods 60
5.4.2 Results 60

6 Conclusions and Future Work **62**
6.1 Conclusion 62
6.2 Future Work 63

Bibliography 64

A Questionnaire Results of Preliminary Tests 68

B Questionnaire Results of Final Tests 73

List of Figures

2.1 The chatbot interface, taken from [27] 8
2.2 Use of buttons in chatbot interface, adapted from [35] 9
2.3 Chatbot design examples, adapted from [26] 9
2.4 Style.me examples 12
2.5 How human brain can missclassify images, adapted from [11] 13
2.6 Feature map generation, adapted from [11] 14
2.7 Feature map generation, adapted from [11] 14
2.8 Application of Rectified Linear Unit (ReLU) to Feature map, adapted from [11] 15
2.9 Convolutional Neural Network (CNN) structure, adapted from [11] 16
2.10 GrabCut example, adapted from [31] 17
2.11 Morphological Operations examples, adapted from [7] 18
2.12 Tiramisu Network structure, adapted from [18] 19
2.13 Springfield app image search screens 20
2.14 Amazon virtual double body 20
2.15 Example of Marxent 21
2.16 Example of Amazon solution 21
2.17 Example of Amazon solution 22

3.1 Store planning 23
3.2 User spawn in store 24
3.3 Store attendant and chat 24
3.4 Store section 25
3.5 Make your Outfit 25

4.1 Loading Screen 29
4.2 Use of buttons and recommended items in chatbox 30
4.3 Confirm item message 31
4.4 Store section - different features 32
4.5 Store attendant 32
4.6 NavMesh overview 33

4.7 Recommended Wall Section 34
4.8 Recommended mannequins 34
4.9 Combine outfits in mannequin 35
4.10 Clothing Details 35
4.11 User cart 36
4.12 Sensibility Settings 36
4.13 Clothing meshes and mannequin models, adapted from [21] 40
4.14 Comparison between Pix2Surf image and Farfetch image 40
4.15 Clothing bounding box 41
4.16 Comparison between Pix2Surf image and Farfetch image after scale 42
4.17 Plain clothing with image 42
4.18 Comparison between Pix2Surf image and Farfetch image after scale 43
4.19 Clothing Images and Histograms 44
4.20 Images histogram analysis 45
4.21 CNN structure, adapted from [11] 46
4.22 Front and back images after all processing 47
4.23 short 50
4.24 Android Application 53

5.1 Unity Profiler 55
5.2 Draw calls tests 56
5.3 Section item's shelf 57

A.1 Age 68
A.2 Gender 68
A.3 Do you prefer to make your purchases online or in a physical store? 69
A.4 How often do you use online sales services? 69
A.5 In these online services, have you ever used a Chatbot to clarify doubts about a product you were looking for? 69
A.6 If so, was it helpful? 70
A.7 I would use an interactive Chatbot interface with a virtual environment like the one shown, regarding examples that only use textbox chatting 70
A.8 It is useful presenting outfits proposals to the user. 70
A.9 Do you think it is preferable to use 2D images or use 3D mannequins? . . . 71
A.10 The shop assistant who follows us and interacts with us makes the experience more realistic. 71
A.11 It is advantageous to have a section dedicated to creating outfits, both with 2D images and using the 3D mannequin. 71
A.12 Having shelves and screens where you can view certain store items gives a more interactive feel to Chatbot, although this functionality exists on the store's own website. 72

B.1 Age . . . 73
B.2 Gender . . . 73
B.3 Do you prefer to make your purchases online or in a physical store? . . . 74
B.4 How often do you use online sales services? . . . 74
B.5 In these online services, have you ever used a Chatbot to clarify doubts about a product you were looking for? . . . 74
B.6 If so, was it helpful? . . . 75
B.7 I would use an interactive Chatbot interface with a virtual environment like the one shown, comparing to a chatbot that only use textbox chatting . . . 75
B.8 Do you think it is preferable to use 2D images or use 3D mannequins? . . . 75
B.9 Reading characters on the screen . . . 76
B.10 Organization of information . . . 76
B.11 Sequence of screens . . . 76
B.12 Use of terms throughout system . . . 77
B.13 Position of messages on screen . . . 77
B.14 Prompts for input . . . 77
B.15 Computer informs about its progress . . . 78
B.16 Error messages . . . 78
B.17 Learning to operate the system . . . 78
B.18 Exploring new features by trial and error . . . 79
B.19 Performing tasks is straightforward . . . 79
B.20 System speed . . . 79
B.21 System reliability . . . 80
B.22 System tends to be quiet/noisy . . . 80
B.23 Correcting your mistakes . . . 80
B.24 Designed for all levels of users . . . 81
B.25 I think that I would like to use this frequently . . . 81
B.26 I found the system unnecessarily complex . . . 81
B.27 I thought the system was easy to use . . . 82
B.28 I think that i would need the support of technical person to be able to use this system . . . 82
B.29 I found the various functions in this system were well integrated . . . 82
B.30 I thought there was too much inconsistency in this system . . . 83
B.31 I would imagine that most people would learn to use the system quickly . 83
B.32 I found the system very awkward to use . . . 83
B.33 I felt very confident using the system . . . 84
B.34 I needed to learn a lot of thing before I could get going with this system . . 84

List of Tables

4.1 Dataset image's composition 46
4.2 CNN experiment results 47

5.1 Background removal performance results 58

Acronyms

A.L.I.C.E.	Artificial Linguistic Internet Computer Entity i, 1
AI	Artificial Intelligence i, 2
AIML	Artificial Intelligence Markup Language i, 1
AR	Augmented Reality i, 20
BOW	Bag of Words i
CNN	Convolutional Neural Network i, x, xi, 12, 13, 14, 15, 16, 18, 39, 45, 46, 47, 49, 62
CNNs	Convolutional Neural Networks i, 13, 48, 49, 50
FCN	Fully Connected Network i, 18
FFM	Five Factor Model i
GMM	Gaussian Mixture Model i, 16
HTML	HyperText Markup Language i, 45
LDP	Label Distribution Protocol i
LSTM	Long Short-Term Memory i, 10
NLG	Natural Language Generation i
NLP	Natural Language Processing i
QUIS	Questionnaire for User Interface Satisfaction i, 60
ReLU	Rectified Linear Unit i, x, 14, 15, 47
RNNs	Recurrent Neural Networks i, 13

ROI	Region of Interest i
SUS	System Usability Scale i, 60, 61
SVM	Support Vector Machine i
UI	User Interface i, v, 2, 3, 8
URL	Uniform Resource Locator i, 45
VR	Virtual Reality i, 20, 63
XML	Extensible Markup Language i, 45

1 Introduction

This chapter presents an introduction to this dissertation project. Firstly a motivation, then the context and description of the problem, main objectives and contributions and to finalise a description of the document structure.

1.1 Motivation

It may seem that chatbots are a technology from the current century, but the first ever developed was back in the 60s. It was called Eliza and was developed by an MIT professor, Joseph Weizenbaum. The way it worked was by passing the user input message into a computer that would parse it to a list of possible scripts representing the human responses. In this case, the scripts simulated the mind of a psychotherapist. The author intended to develop a basic caricature of human conversation.

This was a remarkable work for the time and captured users' attention. Specialists found it engaging and were concerned that chatbots conversations would become indistinguishable from a real user to user conversations [1].

Another remarkable work done in this area that is important to mention was Artificial Linguistic Internet Computer Entity (A.L.I.C.E.), developed by Richard Wallace. This chatbot was based on Artificial Intelligence Markup Language (AIML), used to configure conversation rules. A.L.I.C.E. was a free software community from the 1995-2000 that resulted in the creation of a wide variety of chatbots [32], which was designed to compete for the annual Turing-Test-Based Loebner Prize.

Currently, chatbots are becoming increasingly popular and examples of them can be found everywhere: from a medical consulting chatbot [14], to one that gives psychiatric advises [24], to a chatbot that present us with highlights from sports games [38], the well known Amazon *Alexa* [1] and *Siri*[2] and also from the areas that nowadays mostly benefit from this technology, the e-commerce websites [6, 19].

[1]https://developer.amazon.com/pt-BR/alexa
[2]https://www.apple.com/siri/

The increasing chatbot usage in the online sales industry has become so popular due to the fact that they make it possible to have a 24/7 shopping adviser to help us. Without the use of chatbots, it would be needed to have much more human resources to fulfil this need, leading to more expenses in the end of the month.

Users also agree that chatbots allow an instant response in a easy way of communication, that results in quicker sales and less time wasted navigating throw the website, which in some cases can be difficult due to bad interface design[5].

Apart from all the benefits that the chatbots bring, most chatbots in the market are usually that small text-box in the bottom right corner of the screen. A great investment in the backend of the chatbots allow one to develop chatbots that would simulate the user to user conversations making those conversations indistinguishable from real ones, but one thing that didn't evolve as much was the chatbot's UI.

1.2 Context and Description

This dissertation is done in the scope in the *iFetch* project. *iFetch* is a project led by FarFetch in partnership with Universidade NOVA de Lisboa, Instituto Superior Técnico and Carnegie Mellon University[13]. This project "put together a multidisciplinary team with strong competence in key areas such as multimedia, computer vision and language understanding both from universities and from a leading ICT company. iFetch will leverage FarFetch's unique and rich product knowledge base, massive (multimodal) training data and experience in deploying and maintaining large Artificial Intelligence (AI) systems ensuring a successful research program." The main focus of this project is developing an innovative chatbot that will "track the evolution of the user's shopping need throughout conversation" and "relate product categories and characteristics to the conversation which thus leads to a better user engagement".

It is intended for this dissertation to develop an innovative user interface for the project chatbot that will be using multimodal information in order to promote a more interactive design that will stand out from the existing interfaces that exists on the market. Also, the system should be able to receive input images and process them. These images should represent clothing parts that the user likes, and intends to buy so the chatbot suggests similar items from the website.

1.3 Main Objective

As result of this dissertation, it was developed a Chatbot UI: the main focus of this dissertation is developing an interface for the FarFetch chatbot that could be integrated in the website. It's expected that this interface brings more then just the common chatbox used in most of the chatbots integrated in websites. It will provide an immersive experience for the user to feel that he/she is in the store being attended by the salesperson.

The UI consists of a virtual room representing the physical shop. In this room, there will be some shop sections representing the brands/item categories, and clothing stands displaying some of the items.

There will be an attendant that will follow us through the store, so that the user can interact with him/her and ask questions using the text or image inputs. Through the conversations the attendant will suggest the user some products. Those items will be displaying in a section of the store where the user can combine them into different outfits. Each item has some information associated with it, including a 2D image.

With those 2D images, as a proof of concept, it was intended to generate 3D models and display them in a mannequin, so the user can plan an outfit and see if the top part goes well with the bottom part, something that the user would be able to do in the store by dressing the clothes. This involved image pre-processing to normalise the clothing images.

1.4 Contributions

As result from the work done in the scope of this dissertation, there are the following contributions:

- **Chatbot Interface**: Development of the chatbot interface. This includes having a functional interface that can be used to ask questions to the chatbot and interact with the store items.
- **Integration with backend**: Integration of the interface with the work done in the scope of the iFetch project regarding the backend of the chatbot. This involves having webserver to connect to the interface that will be used to exchange messages. This work is part of other colleagues dissertation that were developed as the same time as this project.
- **System Evaluation**: Evaluation of the developed system through user and performance tests, in terms of acceptance of proposal interface, reliability, usability and understand the complexity of the proposed interface.

1.5 Document Structure

This dissertation is structured with the following chapters:

Chapter 1: Introduction

This is the current chapter where it is provided a brief overview of what are the main objectives to be achieved, contextualisation and a motivation for the work.

Chapter 2: Related Work

This chapter presents the state of the art of the technologies involved. Firstly, it's presented an overview of the chatbot development in the present: what are the main areas of usage of chatbots, their usability and user experience. Then the focus is some aspects referring to chatbots UI.

In a second part of this chapter will be focused on analysing the usage of chatbots in online sales market. The last topic related to the UI that will be presented is related with the generation of 3D models using the clothing images.

In the end of this chapter, will be presented some of the techniques concerning clothes recognition and segmentation and data set to train the models.

Chapter 3: Application Design

Chapter 3 gives an overview of the system that was developed. It starts by explaining the application concept, then its requirements and technologies that were used.

Chapter 4: Implementation

Chapter 4 presents a detailed description of the implementation of the system application. The system is divided into several components and it is explain their implementation and decisions that were made.

Chapter 5: Evaluation and Results

In this chapter will be presented how the system was evaluated, the different testing phases, evaluation methods and the results and conclusions and improvements that resulted from it.

Chapter 6: Conclusion and Future Work

The sixth and last chapter of this dissertation will present the conclusions that resulted from this dissertation and also improvements that should be done in future work regarding it.

2 Related Work

This chapter presents the state of the art of the technologies involved in this dissertation. Firstly it is analysed the current use of chatbots: areas of usage, chatbots features and interfaces and user experience. Then this study will be focused on the use case of chatbots in online sales and how to generate 3d clothing models from 2D images. As a final section, will be analysed the evolution of the technologies that are used for clothes and patterns recognition and matching.

2.1 Chatbots in the Current Days

Chatbots have become a powerful tool for providing intelligence and natural communicative capabilities to the systems. They receive natural languages inputs and produce a natural language response, engaging in a conversation with the user [15]. This technology is used to automate the interaction between a company and costumers, leading the user to believe that he/she is having a conversation with a real human being [10].

As seen in section 1.1, chatbots are becoming popular nowadays, and examples can be found in the most different areas. In the following subsection will be analysed some relevant aspects about chatbots.

2.1.1 Usability Heuristics of Chatbots

According to ISO 9241–11, usability is "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use".

According to [36] the use of some heuristics can help us understand the usability of the product (in this case, the chatbot). The Jacob Nielson's heuristic principles[1], are the ones referred in this section. The ones that were considered the most relevant for this project are the following:

[1] https://www.nngroup.com/articles/ten-usability-heuristics/

- **Visibility of the System Status**: this heuristic says that the user should know about the system status. For instance, how many steps are left to complete some task, if some task will take some time to be completed. This is an important heuristic to consider when applied to online sales. An example would be some queries that might take a while to fulfil like the ones when the user inputs some image to the system. The system needs to process the image, get the items that match it and present it to the user. It's important that the user is warned about that in some way to understand what the program is doing.

- **User Control and Freedom**: this heuristic says that the user should know the risk or implications involved in some actions. As the subject is online shopping, this heuristic needs to be taken into consideration when adding some item to the shopping cart, for example. A confirmation from the user should be requested.

- **Recognition Rather than Recall**: this heuristic says that should be used different ways of communication rather than just text, suggesting the use of emojis because the user can detect their meaning immediately and are the quickest way to consume information while reading.

- **Flexibility and Efficiency of Use**: as the name says, this heuristic is concerned with the efficiency of the communication. In some cases selecting a button to fulfil some task would be faster and more intuitive. This can be applied when the user receives a list of items that the bot is suggesting, and the user want to express his/her interest in some of them, rather it be to confirm that he/she wants to buy it or, for instance, a button to ask more information about it.

- **Aesthetic and Minimal Design**: this heuristic says that the chatbot interface should be as simple as possible and designed in a way that the user should know about everything that he/she is able to do in the system. It's important to take this heuristic in consideration because the main goal is to develop an interface for the chatbot that is not the usual chat box. It needs to be ensured that the user understands how it works.

2.1.2 The 6 Norman Design Principles

Another way of guiding the development of an interface takes into consideration the 6 Norman design principles. Those principles were established by Donald Norman, one of the most notable researchers in the field of human-computer interaction and user-centered design [37]. He defends that interfaces should work in an intuitive and easy way.

According to Norman, the six principles that should guide a development of an interface are the following:

- **Visibility:** similar to what was described in the previous subsection, the user should be able to look at the interface and see all the available functionalities that the interface provides at that moment, and should only be visible the minimal relevant information.
- **Feedback:** this principle says that when performing an action, one should see the reflection of this action in the interface.
- **Affordance:** the interface should be intuitive and the user should be able to deduce how to achieve certain goal. The user should be able to know how to do do some task just by looking at the interface.
- **Mapping:** this principle says that one should know how an action will be reverted in the interface. For example, is is expected that when using a scroll the page should move the same way and with the same pace.
- **Constraints:** this principle says that some tasks should constraint the possible action of the user. For example, when texting a phone number the user should only be able to type number, or when selecting a size for a clothing, it should be one from a certain list of available sizes.
- **Consistency:** if two things look similar, they should reflect similar action in the interface. Users tend to manage things better if they recognize patterns.

2.1.3 Chatbots User Experience

When designing a chatbot, the main focus should be providing a better experience to the end user: the customers. An article concerning a better user experience in chatbots, presents some guiding tips when implementing a chatbot [28]. They present 19 tips for chatbot design and below will be presented the ones that are more related with this project.

When developing the chatbot, it must clearly specify what the user is capable of doing. Chatbots have their own limitations, and sometimes the user might be induced in error and expect something the chatbot is not capable of doing. The conversation should be clear and guided so the user doesn't have any doubts about what to do next.

Another point mentioned in the article concerns the length of the input message that the user is required to write: answers should be direct and as short as possible. They also say that the bot messages should be as clear as possible. The user should be able to understand the meaning of the message with clarity.

Another important thing to be aware of is long tasks that would make the user wait some time for response: the user should always be informed about the task will and if it will take more time then expected.

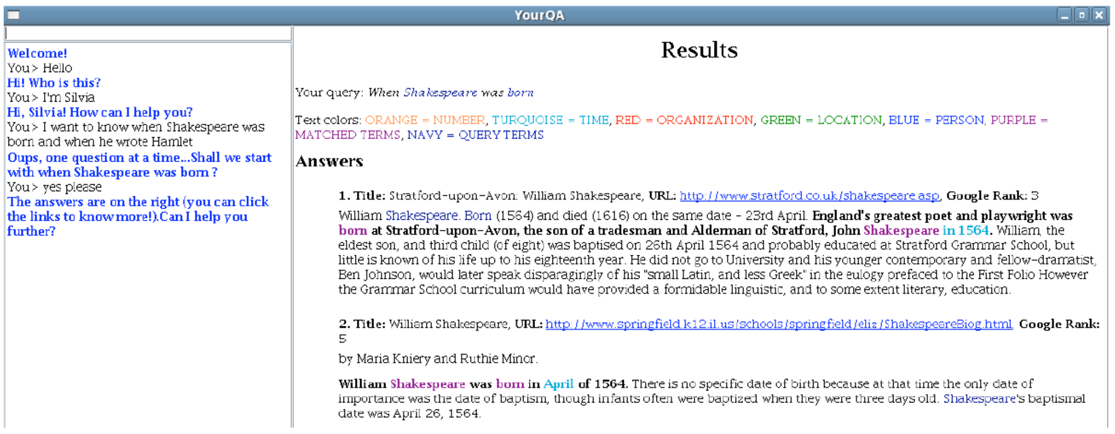
Now, concerning the user interface of the bot, they mention that it should be used other components like buttons or pre-defined list of choices rather then pure text, as input message, but it should be used those only in certain cases because it might trap the user and create the illusion of limited answers, and pushing us far away from the main goal of achieving a conversation as similar as possible to a user-to-user conversation.

One last point says that all interactions should be predictable. When using interactive messages with images and buttons, rather than text, the user might think that most of them are clickable and would trigger some action: the UI should be designed in a way that all buttons are distinguishable and their intent is predictable, avoiding the user to fall in unpredictable situations.

2.1.4 Chatbots Interface

When talking about a chatbot UI, the first thing that comes to mind is the usual text box in the corner of the screen. This type of interface is suitable for most of the cases because it's the same as a typical chat application, but in other cases, it would benefit from other ways of interaction.

The first example is a chatbot interface from a team from University of York published in 2007 [27]. They increased the window size and splitted it in two: on the left side there is the text box and on the right side a panel to present more information about what is being talked about. It could be links for web pages, a more detailed answer for the user question, etc. An example of this interface can be found in figure 2.15.



Welcome!
You > Hello
Hi! Who is this?
You > I'm Silvia
Hi, Silvia! How can I help you?
You > I want to know when Shakespeare was born and when he wrote Hamlet
Oups, one question at a time...Shall we start with when Shakespeare was born ?
You > yes please
The answers are on the right (you can click the links to know more!).Can I help you further?

Results

Your query: *When Shakespeare was born*

Text colors: ORANGE = NUMBER, TURQUOISE = TIME, RED = ORGANIZATION, GREEN = LOCATION, BLUE = PERSON, PURPLE = MATCHED TERMS, NAVY = QUERY TERMS

Answers

1. Title: Stratford-upon-Avon: William Shakespeare, **URL:** http://www.stratford.co.uk/shakespeare.asp, **Google Rank:** 3

William Shakespeare. Born (1564) and died (1616) on the same date - 23rd April. **England's greatest poet and playwright was born at Stratford-upon-Avon, the son of a tradesman and Alderman of Stratford, John Shakespeare in 1564.** William, the eldest son, and third child (of eight) was baptised on 26th April 1564 and probably educated at Stratford Grammar School, but little is known of his life up to his eighteenth year. He did not go to University and his younger contemporary and fellow-dramatist, Ben Johnson, would later speak disparagingly of his "small Latin, and less Greek" in the eulogy prefaced to the First Folio However the Grammar School curriculum would have provided a formidable linguistic, and to some extent literary, education.

2. Title: William Shakespeare, **URL:** http://www.springfield.k12.il.us/schools/springfield/eliz/ShakespeareBiog.html, **Google Rank:** 5

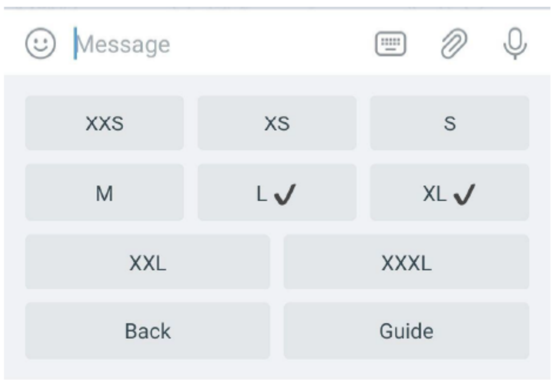
by Maria Kniery and Ruthie Minor.

William Shakespeare was born in April of 1564. There is no specific date of birth because at that time the only date of importance was the date of baptism, though infants often were baptized when they were three days old. Shakespeare's baptismal date was April 26, 1564.

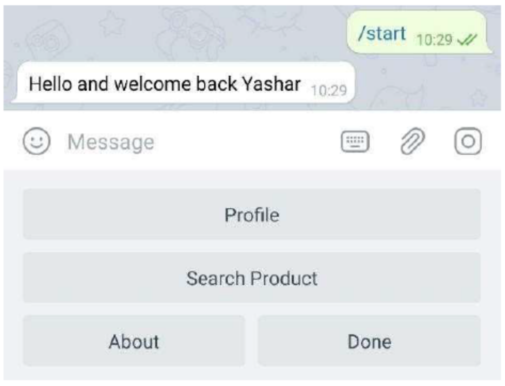
Figure 2.1: The chatbot interface, taken from [27]

Another related work was done by a team from the Department of Computer Science from Lakehead University [35]. They developed a wrapper for online shopping chatbots in mobile devices. The chatbot, instead of having only pure text messages, also uses buttons and cards.

They suggest that the buttons are helpful when selecting the size of a product: instead of requesting the user to insert the size, it is more convenient for the bot to display the buttons with the available sizes and the user only needs to select it. An example of this interface can be seen in figure 2.2.



(a) Buttons for size selection

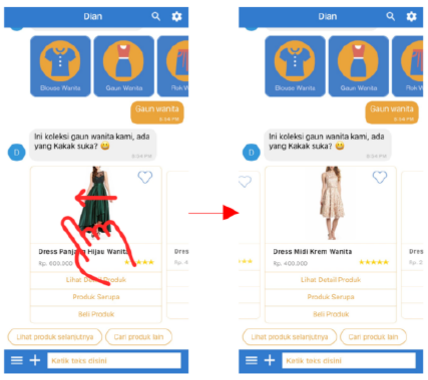


(b) Select action buttons

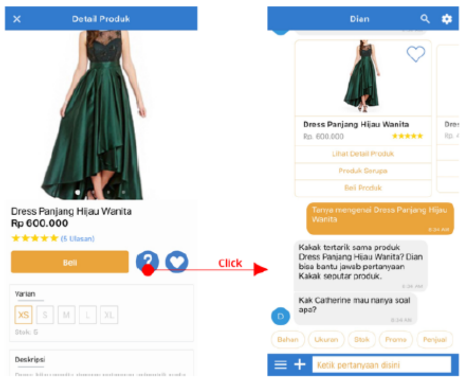
Figure 2.2: Use of buttons in chatbot interface, adapted from [35]

The cards' purpose is for displaying images/videos of the products the user requests mixed with some links and text.

Another work done in this area, from a group from the School of Electrical Engineering and Informatics, Institut Teknologi Bandung in Indonesia, developed yet another mobile chatbot interface for online fashion shopping [26]. This team followed the user-centered approach for chatbot development (later presented in section 2.1.6) and proposed some features like messages as a swiping list with different products presented by the bot, that can be seen in figure 2.3(a). Each item of this list contains things like the product image, information about it and the link for the web page or for a more detailed view like in figure 2.5(b).



(a) List swipe feature



(b) Detailed screen

Figure 2.3: Chatbot design examples, adapted from [26]

2.1.5 Chatbots in Online Sales

With the evolution of technology, smartphones and desktops have become part of our life. According to [23], there are around 3.5 billion smartphone users around the world and, by 2018, more than 80% of households from developed countries have access to personal computers [2].

With ease of internet access, the online sales have been thriving in the last few years. By 2019, the e-retail sales account for 14.1% of all retail sales worldwide with a contribution of $4.5 trillion. By the end of 2020, it's expected to reach 16.1% and 22% until 2023 [8, 9].

With this increase in demand on online sales, came the need for the stores to provide better online support. Conversational shopping channels such as WhatsApp are already being used by online e-commerce shops, but as the sales market grows, the need for better scaling solutions appears. One of the ways to solve this problem is by the use of AI agents. Some examples of this technology can be found in some well known brands like H&M[2], Domino's[3], Sephora [4] and Burbery[5].

2.1.6 Chatbot Implementation Techniques

Below is presented two different approaches for chatbot development. One of them is focused on data and backend of the chatbot. The second one is focused on the user and chatbot interface design.

Data-Centered design: a team from Microsoft Research Asia [6], presented a methodology for developing Data-Centered customer service chatbot that leverages large-scale and publicity available data, what they call SuperAgent. They use product information (PI), questions and answers (QA) and customer reviews (CR) as a source of information for the bot. When a user visits a product web page, SuperAgent crawls the HTML page and scrapes PI + QA + CR data.

They developed four different engines to process the input message from a user: one based on the product description (PI) using a DSSM model [17], another based on FAQ questions (QA) using a regression forest model [22], a third one based on customer reviews (CR) and a last one for chit chat conversations, mainly designed for responding to greeting messages using an attention-based Long Short-Term Memory (LSTM) seq2seq model trained with Twitter conversation data [3]. SuperAgent will choose the best answer based on the confidence of each model.

They made an usability analysis and concluded that SuperAgent has improved the user experience in online shopping.

[2] https://bots.kik.com/#/hm
[3] https://anyware.dominos.com/
[4] https://www.kik.com/casestudy/sephora/
[5] https://botlist.co/bots/burberry

User-Centered design: another example of chatbot implementation technique is the one presented in [26], where it is proposed an User-Centered approach. This development technique differs from the one presented above in the way that they believe that for a good chatbot design, firstly the focus should be on understanding the user's needs. They start by doing user research through questionnaires to gain insights from users. This research helps define the user needs, scope and defining a user model. The next step was defining the usability and user experience goals, and with this, derive features that the bot should fulfil.

The development phase is followed by a testing and evaluation phase. The users will have an exploration phase to understand the application an then a task completion phase. In the end its done some post-test interviews and questionnaires to understand if the prototype fulfil the usability goals defined for the chatbot.

2.2 3D Modelling from 2D Images

As the goal is to develop an interface for an online clothing sales chatbot that one of the features will be retrieving similar images to the ones provided by the user, it's needed to have a way of presenting those images to the user. When buying an outfit in a physical store, the costumer always has the opportunity to use the dresser to try it out and see if some pants go well with the jacket he want, something that is impossible via online selling.

Given this problem, there are two ideas to solve it: one was to use some mannequins that will be dressed up with the clothes that the user wants to buy. The second idea consists of using the user avatar to dress the clothes, so he could check how it looks in a mirror. That would be an easy job if every item in the store had 3D models. Since there are only 2D images, those need to be converted into a 3D model.

2.2.1 Pix2Surf

A team from Germany developed a method to automatically transfer textures of clothing images (front and back) to 3D garments worn, in real time, called Pix2Surf [21]. Their algorithm is implemented to handle input images of t-shirts, shorts and pants and they have the possibility of generating the 3D model with different poses. They made their code available so it can be used to train models with the datasets.

In their case, they crawled images from different online clothing stores to train their model. For the training process, they use pre-defined 3D meshes for the three types of clothing they are using, that can be created using Blender[6]. Each different pose that can be used, has its own meshes to fit the model. Then, they start by using GrabCut[7] to obtain

[6]https://www.blender.org/

[7]https://docs.opencv.org/3.4/d8/d83/tutorial_py_grabcut.html

the silhouettes of the crawled clothing images. There were scripts for this process, where they do silhouette matching, correspondence extraction and texture map.

Once the data has been obtained, they train a CNN for mapping and segmentation. They focus on the silhouette of the images only and not the texture. The models will receive the images as input and will map the clothing silhouette to the 3D meshes.

After training the models, it can be tested with the clothing images intended to dress the mannequin with, and it will generate an image for each clothing, this is the texture that needs to be applied to 3D meshes.

All the code for image processing, training and obtaining the 3D models is available in their repository, although with a restriction of non-commercial use license. As mentioned, they are doing this for only three types of clothing, but it can be extended for other types by generating the clothing meshes and training different models with the appropriate apparel images.

2.2.2 Style.me

Style.me[8] is a company that allows you to add a virtual fitting room to your clothing store website. They believe that "shopping should be an engaging, efficient, and enjoyable experience for consumers"and it should "maximize loyalty and revenue while minimizing the environmental impact". Their main goal is to "help our clients stay at the forefront of innovation".

Their technology allows the user to select the mannequin body shape according to its own: there are customizable parameters like the user height, weight and body shape. The user can also change the mannequin face and hair style to look as similar as possible to him/her. For women, the user is allowed to select the bra and hips size, waist and length of the legs.

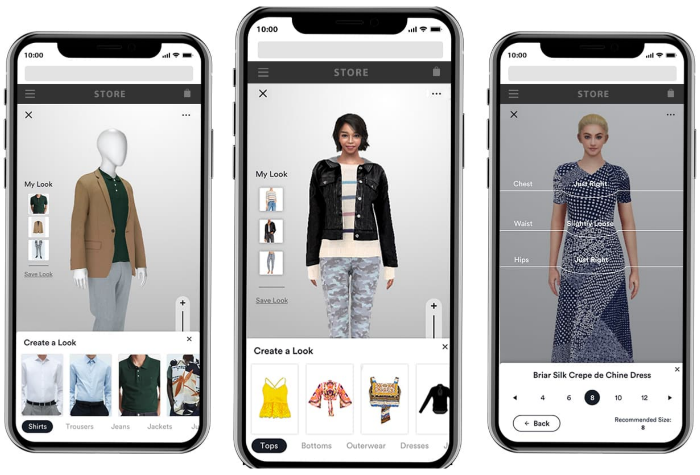


Figure 2.4: Style.me examples

[8]https://style.me/

2.3 Convolutional Neural Networks

In the current days, there is a strong resurging interest in neural-network-based learning [20]. This happens due to the fact that this technology has a superior performance in image and video understanding. ár There are two distinct types of neural networks: the Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). CNNs are used for image and video analysis for pattern recognition. The RNNs were designed to work with audio and speech waveforms. Here will be discussed the implementation of CNNs and how they work.

A CNN is an attempt to virtually simulate the human brain. To understand how CNNs work, it is important to start by understanding briefly how the human brain works when recognising visual objects. When the human eye looks at a certain object, the brain will try and extract some important features from it to be able to recognise which object is he/she looking at. But sometimes, by just taking a brief look, he/she might not get all the important information needed to well recognise what has been seen.Figure 2.5 is a good example of how the human brain can missclassify what is seeing due to the features that it is processing.

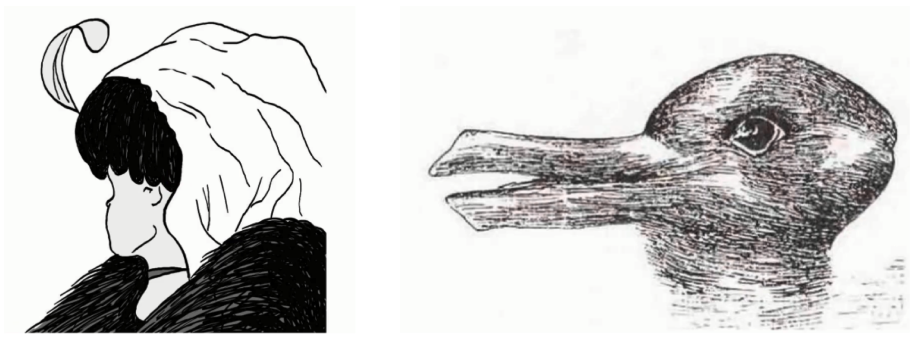

(a) Old lady/lady looking back (b) Detailed screen

Figure 2.5: How human brain can missclassify images, adapted from [11]

Those two images show that, depending on the angle and attention given to certain parts of the image, the human brain detects some features and predicts what it is seeing according to them. To accurately analyse an image, the human brain needs to process it in different ways so that he gets all the features. This is how the CNNs work: they look at the images in different ways to extract all the possible features and, given those features, will learn to categorise images and output its prediction.

2.3.1 How do CNNs Work?

In general, a CNN has the following steps: the user inputs an image, this image will be processed by the CNN and then the output label is generated. The process inside the CNN can be break down in the following steps:

- **Convolution:** the first step of a CNN is the extraction of features. Convolution is applying a filter to an image to reduce its dimension, retaining all the important aspects of it. For each filter applied to an image, a feature map is generated. In general, the convolution step uses several different filters and, for each image, generates several feature maps.

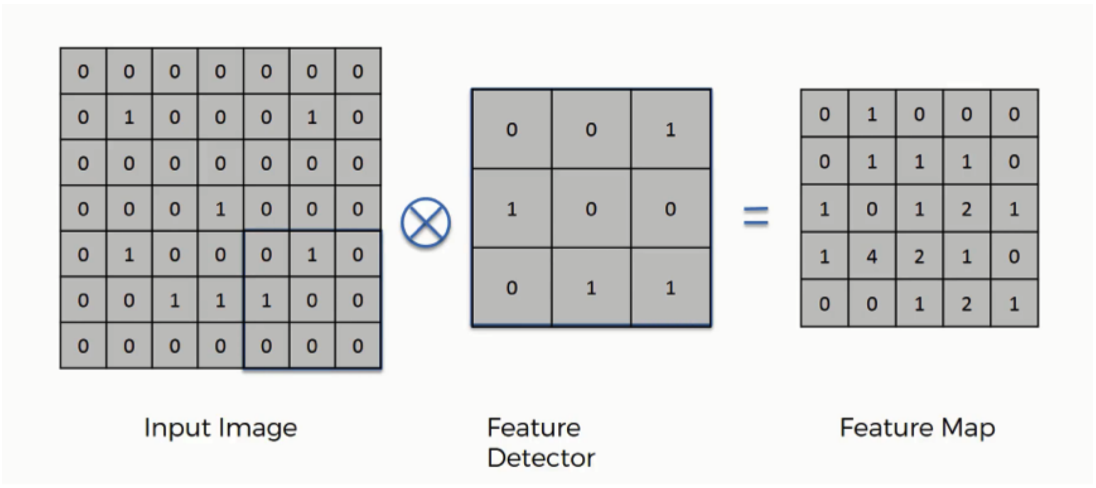


Figure 2.6: Feature map generation, adapted from [11]

- **ReLU layer:** ReLU is a process done in the convolution step.ReLU is a rectifier function and its purpose is to break the linearity in the images, and maintain only the abrupt changes in the images. This means that it will maintain the non-linear parts of the images (the ones that are good candidates for being a representative feature of the image). In figure 2.7, can be seen the ReLU rectifier function graph.

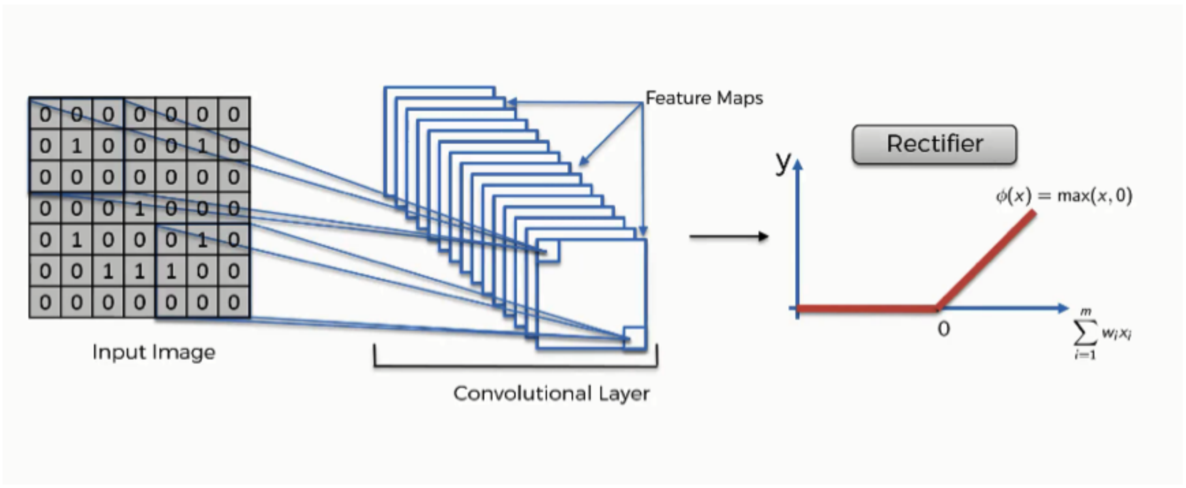
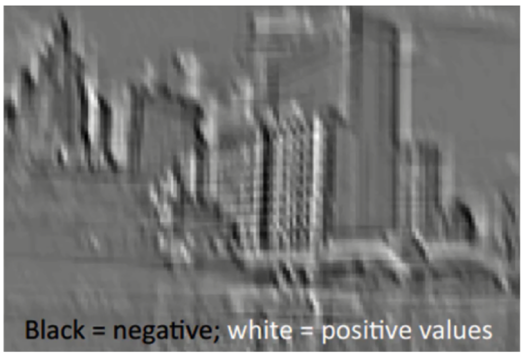
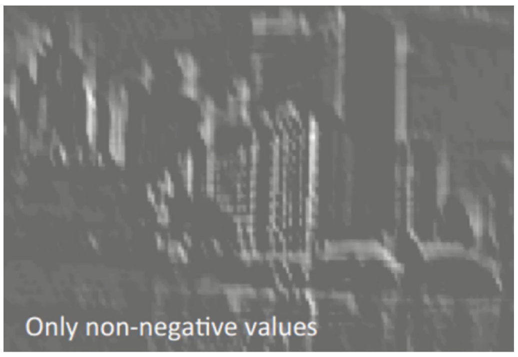


Figure 2.7: Feature map generation, adapted from [11]

The figure 2.8 shows an example after processing a feature map with ReLU. It can be seen that all the dark spots and transitions between black, grey and white disappear, maintaining the whiter parts of the image.



(a) Before ReLU Rectifier



(b) After ReLU Rectifier

Figure 2.8: Application of ReLU to Feature map, adapted from [11]

- **Feature Pooling:** the third step of the network is necessary for the network to detect an object in an image, even if it is rotated, scaled or with a different orientation: this is called spatial variance. Feature pooling help the network abstract from those characteristics and focus on the important features.

  There are several ways of pooling: Max pooling, mean poling or sum pooling. Considering the case of Max pooling, each feature map will be processed with an X by X image (let's consider 2x2), and maintain the Max value of the current four. This reduces even more the size of the feature map, but maintain the important parts.

  This process of selecting the real important parts of the feature map, will help the CNN to avoid overfitting to the training images.

- **Flattening:** The last step of the pre-processing of the input images is flattening the feature maps. This is done after the feature pooling, and the goal is to turn the 2D array into a flattened 1 dimension array.

- **Full Connection:** At this step, all the images have been pre processed and they are ready to go through the network. The network has the structure represented in figure 2.9.

The input layer will contain the flattened array that resulted from the pre-processing of the images. Then, it will go through the fully connected layer that will take the input data and combine the features into attributes to make the CNN capable of classifying the images. In case that the network miss classifies an image, there will be used a cross-entropy function to calculate an error. As the network is fully connected, it back propagates the error to update the neurons' weights.

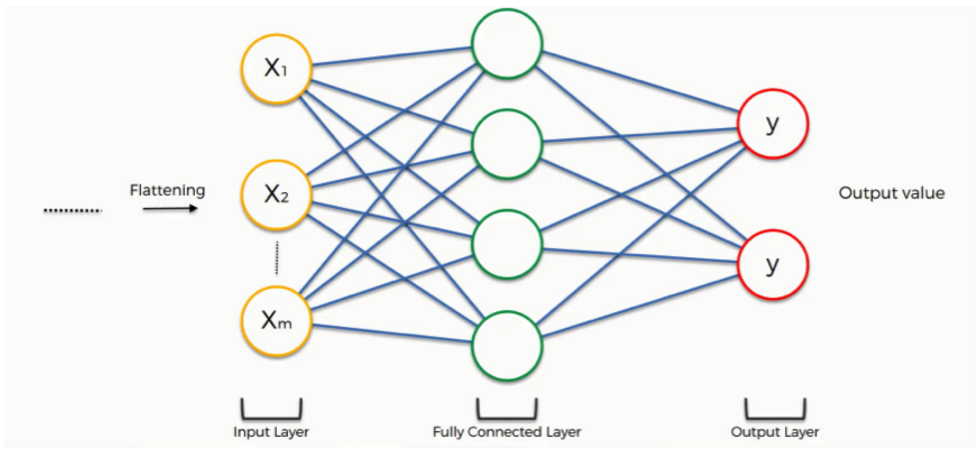


Figure 2.9: CNN structure, adapted from [11]

With this, each class starts noticing which neurons are the most important for them: for example, if the images of a certain class always get values close to one in a certain neuron, it will have this neuron in consideration to classify the images.

2.4 Background Removal

In the interface it is expected that the items are displayed to the user as 2D images. To improve the quality of the images, it would be better to remove the background of the images. This section presents three different background removal techniques.

2.4.1 Semi-Automated Tool

In [34] it is proposed an approach to this problem using GrabCut [29]. GrabCut is an algorithm based on graph cuts to separate the foreground from the background. Iteratively, it uses a Gaussian Mixture Model (GMM) to model the foreground and background. Then, based on the new pixel distribution, a graph is generated. The final step is a graph cut to exclude some of the pixel from the background.

The drawback of this algorithm is that it needs the object to be marked with a bounding box. In [34] they used a technique based on facial recognition, assuming that the body is below the face and using the face dimensions and human proportions to derive the position of the other parts of the body. This works well for full body pictures, but if the image doesn't have a face and the box has to be drawn manually.

In figure 2.10 can be seen an example of use of GrabCut.

2.4.2 Morphological Operations

Morphological Operations can also be used to extract the background of images with plain backgrounds. These techniques are used for images with a solid color background.

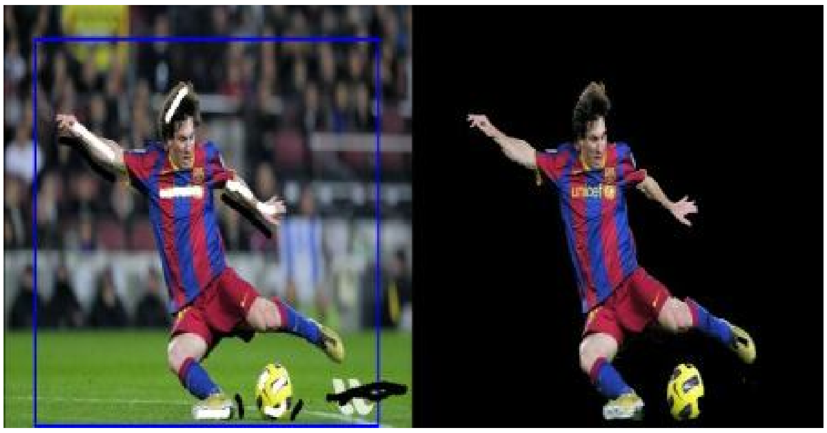


Figure 2.10: GrabCut example, adapted from [31]

2.4.2.1 Basic Operations

To explain how Morphological Operations can be applied to background removal, it is important to know how they work.The two most important operations are the **Erosion** and **Dilation**. Those are the basis of every Morphological operation.

- The effect of **Erosion** is to remove spurious pixels (such as noise) and to thin boundaries of objects on a dark background (that is, objects whose pixel values are greater than those of the background).

- The effect of **Dilation** is to fill up holes and to thicken boundaries of objects on a dark background (that is, objects whose pixel values are greater than those of the background).

Other then this two basic operations, there are two well know operations that combine this two: the **Opening** and **Closing**.

- The process of **Opening** has the effect of eliminating small and thin objects, breaking objects at thin points, and generally smoothing the boundaries of larger objects without significantly changing their area.

- The process of **Closing** has the effect of filling small and thin holes in objects, connecting nearby objects, and generally smoothing the boundaries of objects without significantly changing their area.

In figure 2.11 can be seen an example of the 4 methods explain above. In addiction, there is also an image referencing an image mask that will be explain in the next subsection.

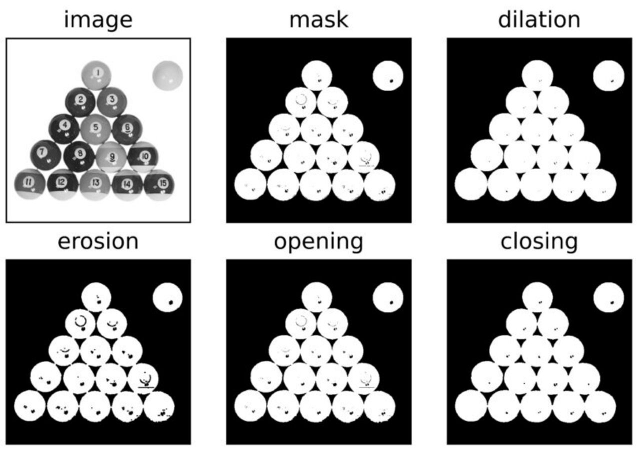


Figure 2.11: Morphological Operations examples, adapted from [7]

2.4.2.2 Applying the Techniques for Background Removal

So, the used method starts by loading the image as grey scale. Then,it applies a threshold to the image and changing all the values between 250 and 255 to 255 and all the other to 0.

Then, as the morphological operations assumes that the background of the images has a solid black color, and the clothing images have white backgrounds, the image is negated to turn the white into black and vice-versa.

Then, will be applied the Morphological Operations of **Opening** and **Closing** to the image in order to remove image noise.

The next step consists of applying a blur to the mask in order to smooth the edges. The last step consists of a linear stretch that is used to turn the mask only into values equal to 0 or 255. The pixels with value equal to 255 are the ones that the algorithm identifies as being part of the object.

2.4.3 Using CNN

Another approach [12] is using CNN to remove the background of images. They proposed DOG, a model that uses Tiramisu, a deep learning approach for background removal. They ran experiments using three different algorithms: Tiramisu, Unet and Fully Connected Network (FCN). The best results were achieved using Tiramisu because this algorithm can capture sharp edges in the image.

Tiramisu is a kind of convolution network where each layer is connected to every other layer called DenseNet [16]. This type of structure has some advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

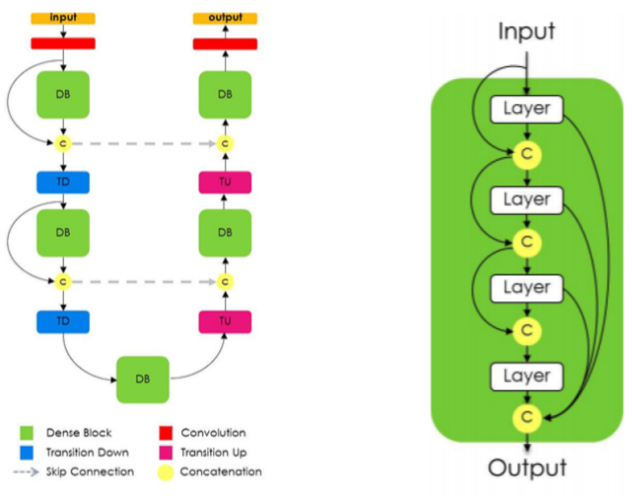


Figure 2.12: Tiramisu Network structure, adapted from [18]

When using models for object recognition, they concluded that some images that were initially missclassified, can be well classified into their types if pre-processed with the background removal model.

2.5 Clothes Retrieval in Online Shopping

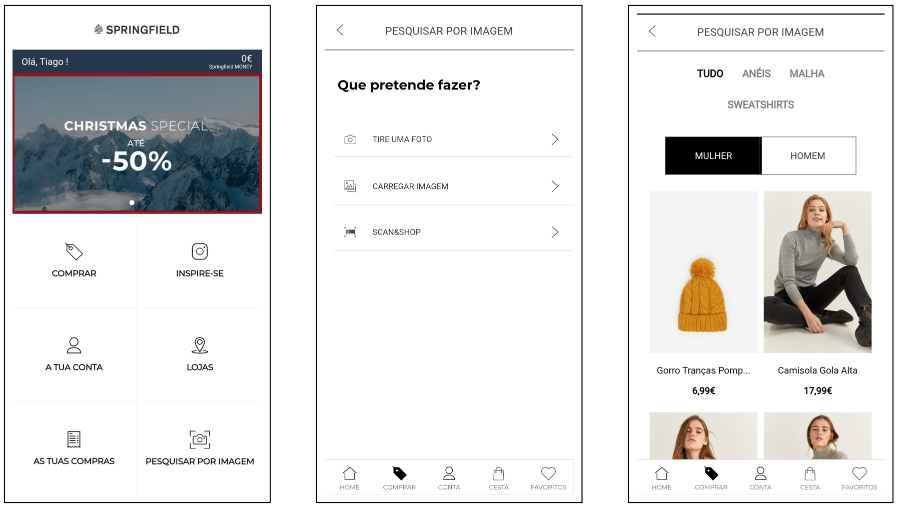
A good market example that is important to be mentioned because it was developed by a well known company and it fits exactly in what it is intended to be developed, is the Springfield app[9]. They introduced the functionality of image search that works exactly what is intended in this project. They give three options to the user: take a picture, select one from the device or scan the clothing shop tag. Then a list of similar clothes are suggested to the user. The only difference is that this functionality will be introduced in a chatbot, taking the most benefit from it.

Another example from a market stand point that was announced recently and is yet to be launched is the Amazon virtual double body. In this system, they are developing a clothing recommendation system where the user selects the type of clothing they want, the color, and this will be shown in a human model representation to show how they would look. Users can select the fabric, colour, length, fit, neckline, sleeve length and even customise their own labels.

There is not yet much information released about this project, but more details can be found in [4].

Still another investment in this area from amazon, is the amazon echo look [10]. This system uses an webcam that takes a photo of your daily outfit and evaluates it. Then,

[9] https://myspringfield.com/pt/pt/landing-app
[10] https://www.theverge.com/2018/2/7/16984218/



(a) Main Screen (b) Search Screen (c) Results Screen

Figure 2.13: Springfield app image search screens

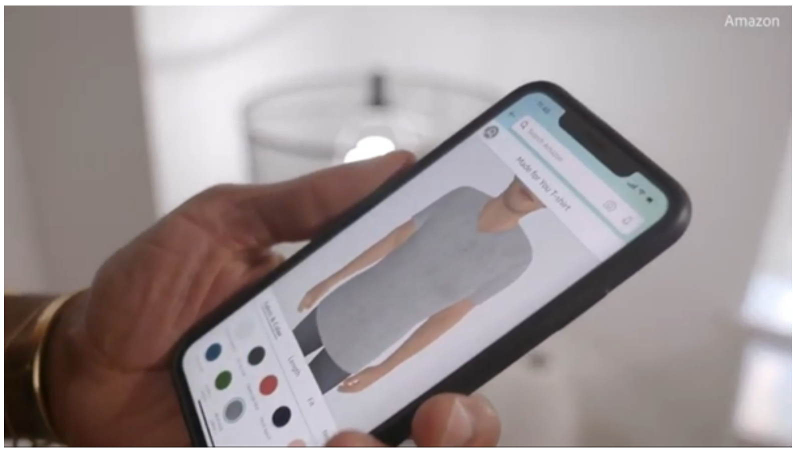


Figure 2.14: Amazon virtual double body

some similar items from the website will be recommended to the user.

2.6 Virtual Shopping Environments

In the past years there was a great investment in the online shopping. Nowadays, in the fashion industry there are website with features like Style.me (section 2.2.2) that allow having a different perspective of the clothing.

But when going a bit far from the clothing industry, can be find different solutions that give use to Augmented Reality (AR). One of the is called *Marxent*[11], and it is developed for rendering 3D objects. This system can either be used with a smartphone or with Virtual Reality (VR) headsets, and let the buyer see how the object they want to buy would look

[11]https://www.marxentlabs.com/

like in their homes. The user need to point the headset or the smartphone camera to some location in their hoes and a 3D render of the furniture will be displayed. They have solution for furniture, kitchen, office and others. An example can be seen in

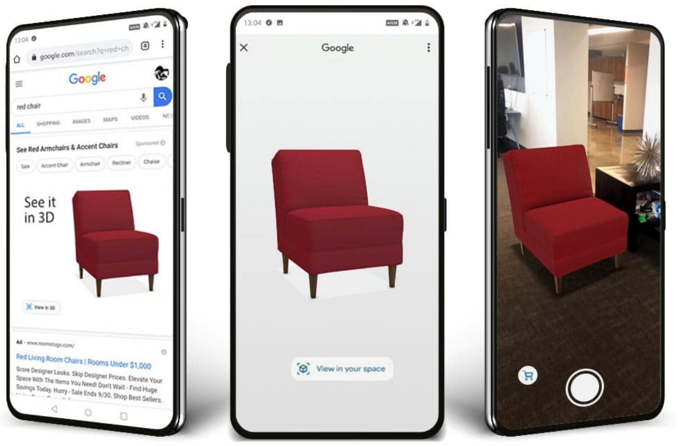


Figure 2.15: Example of Marxent

There are also some similar solutions from Amazon and IKEA[12]. An example from the amazon solution can be seen in figure 2.16.

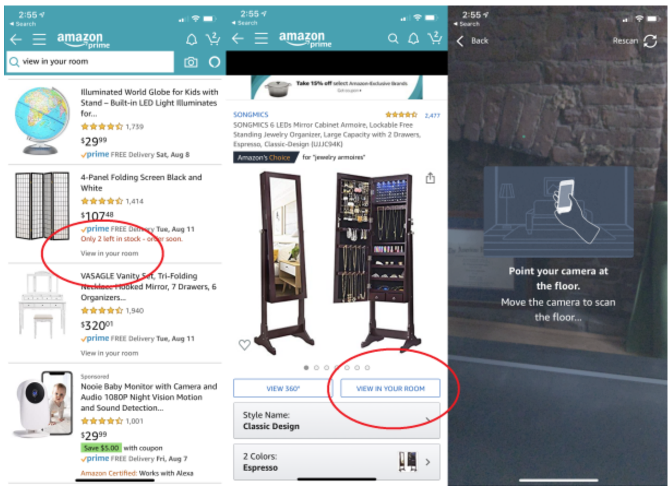

amazon prime
view in your room
VIEW 360°
VIEW IN YOUR ROOM
Point your camera at the floor.
Move the camera to scan the floor...

Figure 2.16: Example of Amazon solution

Amazon also developed a solution called Showroom[13] that allows to customise an entire room with the 3D rendered products. There are several different rooms and the user can change all the elements, from the color of the walls and the type of floor, to every single decoration object (figure 2.17).

[12]https://www.youtube.com/watch?v=r0ViFTEb8aQ&t=58s&ab_channel=IKEAUK
[13]https://www.amazon.com/showroom/switch

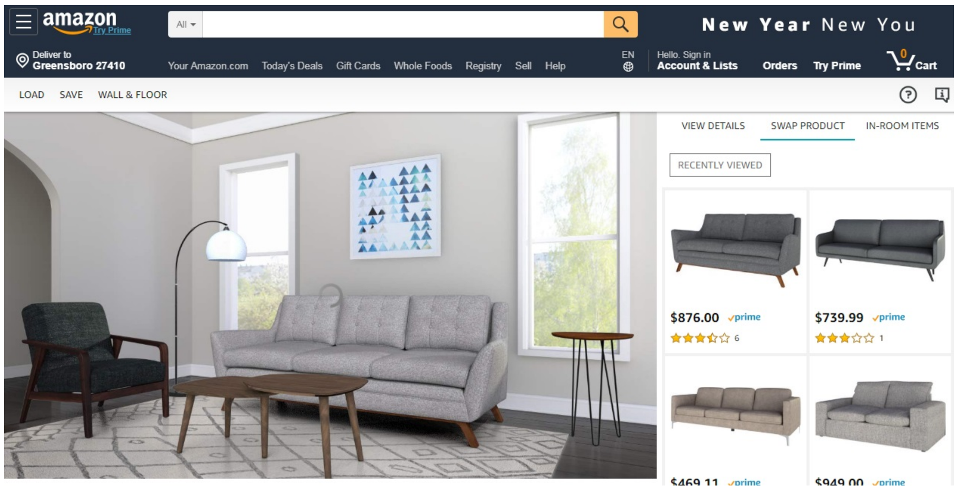


Figure 2.17: Example of Amazon solution

2.7 Discussion

To summarise this chapter, this section is a brief discussion about the work presented above.

When developing the interface, will be followed an user-centered approach. It will be taken in consideration the usability heuristics and user experience to fulfil the user need and promote a better experience. Some functionalities will be added to the chatbox like the different types of input messages presented in [35, 26].

Regarding the 3D models, the work done by [21] will be useful to generate the clothing 3D models to use as mannequins in the virtual store.

From the work presented in section 2.4, it is intended to experiment the different techniques and analyse which one is better in terms of performance and final result.

During the implementation of this project, there will an evaluation process using system and acceptance testing to get feedback about the usability of the interface.

3

Application Design

The following section starts by explaining the design of the application and decisions made regarding the system requirements. Will shortly describe the application concept and the development platforms that were used.

3.1 Application Concept

Chapter 1 presents the main objective is to develop an interface for a chatbot that differs and improves from the ones that are seen across the internet. To achieve this goal, it was intended to develop a *WebGL* user interface that would represent a virtual room to emulate the experience of buying products in physical stores. The store has the plan shown in figure 3.1. There are three sections: the spawn, followed by the store section with the products and the the "Make your Outfit"section.

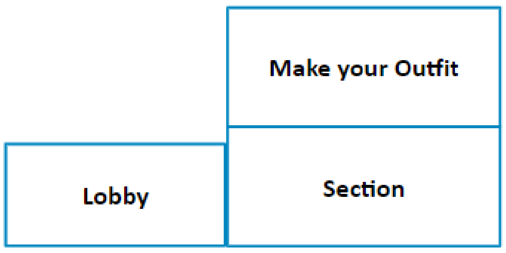


Figure 3.1: Store planning

The user spawns in the entrance of the store (figure 3.2) and has to use the mouse and keyboard to navigate through it. Sometimes there will be the need to use the mouse cursor and to do this the user can click on the right mouse button.

In the spawn/entrance section, there is an attendant that is the embodiment of the chatbot, that the user can talk to and ask questions using the chatbox in the bottom right

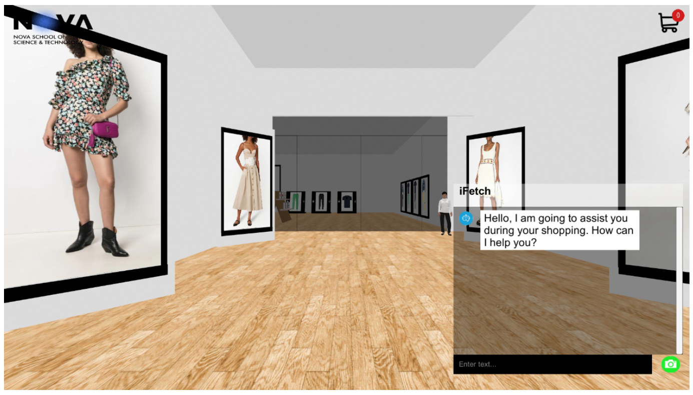

iFetch
Hello, I am going to assist you during your shopping. How can I help you?
Enter text...

Figure 3.2: User spawn in store

corner (figure 3.3). The attendant will always be available to the user in each section of the store.

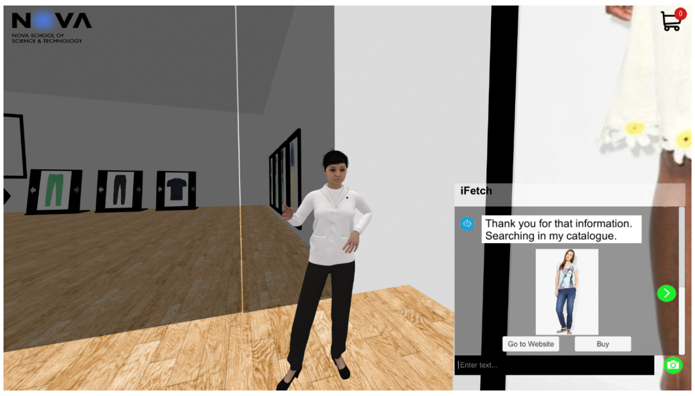


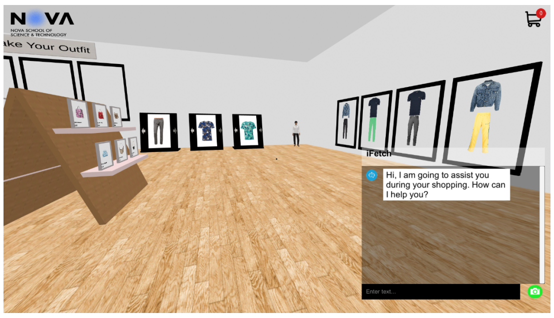
Figure 3.3: Store attendant and chat

With the progress of the conversation, the user is expected to ask questions in order to find the product that he/she wants, and some items will be recommended to him/her. To make the conversation easier, there are different types of input messages (text, images and buttons) in order to facilitate the input from the user.

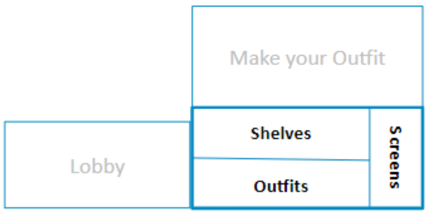
The user also has the ability to insert an image from their device with the purpose of asking for similar clothing. As he/she enters the store, there will be different sections (figure 3.4(a)). Those sections can be divided by clothing parts (jackets, jeans, shoes, etc) and/or brands.

Each individual section has its own items displayed and recommended outfits. There are different kinds of displays according to the products (e.g. if there are clothing items or jewellery/bags). In figure 3.4(b) can be seen the division that was made in this store section according to the 3 different components (screens, outfits ans shelves).

The user is able to select each item, ask for more information about it, access the


NOVA SCHOOL OF SCIENCE & TECHNOLOGY
Make Your Outfit
iFetch
Hi, I am going to assist you during your shopping. How can I help you?
Enter text...

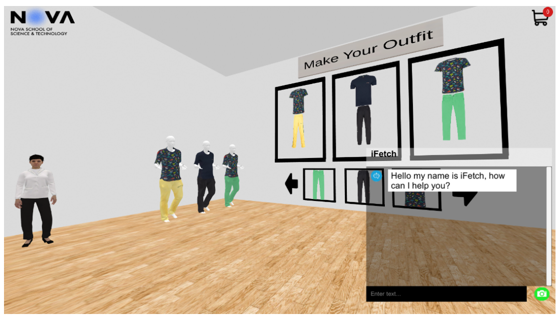
(a) Store section overview



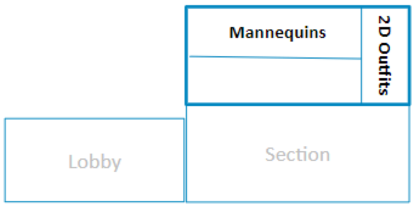
(b) Store section planning

Figure 3.4: Store section

product website or add it to the cart. There is a common area to all the sections where the user is able to see each recommended item or items that the user marked as interesting (figure 3.5(a)). The user is also able to combine them into an outfit to see if they go well together, either in 2D or 3D mode, using a mannequin to display the clothing. Figure 3.5(b) shows the make your outfit section division.


NOVA SCHOOL OF SCIENCE & TECHNOLOGY
Make Your Outfit
iFetch
Hello my name is iFetch, how can I help you?
Enter text...

(a) Make your Outfit Section



(b) Make your Outfit Planning

Figure 3.5: Make your Outfit

To provide a better user experience, the attendant has some animations, like following the user to every store section so he/she can maintain a conversation and some greeting and talking gestures when sending a message.

Summarising the application, it was decided that the system would have the following features:

- Different store sections
- A chatting functionality with the chatbot
- Different types of input messages

- Input an image to search for similar images
- Accessing recommended clothing: store walls or 3D models
- Combine items into and outfit
- Access apparel web page
- Add apparel to cart
- Access cart

3.2 Application Requirements

After analysing and deciding how the interface would work, it was important to establish all the requirements and necessary components for the application to run:

User Interface: mainly, there was the need to have an environment to develop user interfaces. It should be taken into consideration the usability and user experience factor presented in section 2.1.1 and section 2.1.3. For example, as reported, it is expected for the store to have several sections with different items. It is important that it is explicit to the user which items the user can find in each section, so he/she doesn't feel lost. To achieve this it is important to follow the heuristic regarding "Aesthetic and Minimal Design".

Another example of heuristics that was applied in this interface is "Flexibility and Efficiency of Use": the interface should provide efficient and easy ways for the user to fulfil the tasks. Less effort should be required to achieve the expected results.

Setup and Communication Server: at the start of the project, the backend of the chatbot was not yet developed: there was the need to develop a backend server to be able to have some communication and data exchange. This allows the interface to get the setup information for the store sections. This information is used for testing all the interface functionalities, including the chatbox, with a predefined list of messages. Those messages were extracted from a conversation that was saved on the server. Those messages could be pure text or multimedia messages with recommended items.

Image processing algorithms: as this interface works mainly with images, it was required to pre-process those images so they are normalised, with the same aspect ratio and with no background. Another problem that needed to be solved using image processing was the generation of the back shots of the clothing to be able to generate the 3D clothing models. This is a problem because most of the dataset from fashion websites doesn´t have images of the clothing from the back.

Integrate Interface with Backend Chatbot: at the end of this project, there was the need to establish the connection of the user interface with the chatbot. This required defining the endpoints for communication and message formats. There was also the need to deploy it and make the application available for User Testing.

3.3 Technologies

This section introduces the technologies that were used to develop this project. There are two main focuses: how to develop the interface and how to implement the image processing algorithms.

Unity: the main concept of this project was inspired by **Hubs** virtual room [1]. Hubs is a platform used to create virtual rooms that can be used for a different way of performing meetings with friends, as it provides voice and text chatting, with the ability to customise the room. The user is able to "share content with others in your room by dragging and dropping photos, videos, PDF files, links, and 3D models into your space". Each user has its own avatar and they can navigate the room and interact with each other.

But, as Hubs is a recent project, and it is hard to find online support for it, its also difficult to deduce if all the expected features were able to be implemented using it, but it was considered as a basis to take into account when developing the chatbot interface.

Given that, it was decided to explore another technology: **Unity**[2]. Unity is a cross-platform game engine mainly used for mobile and desktop applications but also allows the user to export their applications as WebGL to be integrated into a website. With Unity, 2D and 3D games can be developed. The most recommended language for Unity projects development is C, but other languages like JavaScript and Python can be used too. In this project was used C as it is mostly used and has better online support.

Unity is based on Scripts that are associated with the game objects to control their behaviour, and uses **Visual Studio**[3] as the main IDE for the development.

Mixamo: in the store, it is expected that the attendants have some kind of animation to promote a better user experience, as it gives the user the idea that the attendant is really talking and interacting with him/her. To accomplish this, it was used **Mixamo**[4]. Mixamo is a character and animation dataset from **Adobe**[5], that allows to

[1] https://hubs.mozilla.com/
[2] https://unity.com/
[3] https://visualstudio.microsoft.com/
[4] https://www.mixamo.com/
[5] https://www.adobe.com/

import 3D Avatars to Unity and also provide a dataset of various animations to set the attendant behaviour.

Pix2Surf: for the generation of the 3D mannequins was used **Pix2Surf** presented in section 2.2 [21]. This technology allows to generate the 3D clothing textures given the 2D images from the clothing (front and back shots). Those textures can be used to apply on 3D clothing meshes that represent the t-shirt or the trousers, for example.

Google Colab: the images used to generate the clothing textures to apply to the model needed to be pre-processed. This was done using **Google Collaboratory**[6]. Collab allows to execute **Python** scripts in the browser, without the need of any configuration, the users have access to a remote machine to run their scripts and it allows an easy way of sharing the project. Collab was used to do all the pre-processing using **Python** and some libraries like **OpenCV**[7] and **numpy**[8] .

[6]https://colab.research.google.com/
[7]https://pypi.org/project/opencv-python/
[8]https://numpy.org/

4 Implementation

This chapter describes the process of conception and development of this application. This project has two main focus: the user interface that was developed using the Unity Game Engine, and a second component related with image processing and the generation of the 3D clothing models. The first section will start by giving an overview of the overall application features and then describe the implementation of each part in more detail.

4.1 Interface Features

This section describes the chatbot interface features:

Loading Screen: When starting the application, the user is presented with a loading screen. This is used for the system to gather all the information needed to setup the store displays. This information is related to the products of the store items. More details will be given in section 4.2. Figure 4.1 shows an example of this screen.

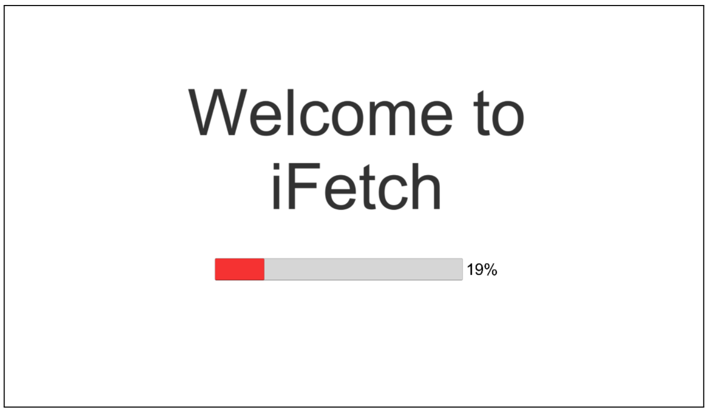


Figure 4.1: Loading Screen

Chatbox Features: as referred before, the main goal was to develop a chatbot interface that would not be the chat box in the corner of the screen that is used in every chatbot all over the internet. For that, this project started by adding to the chatbox some features mentioned in section 2.1.4.

As this interface is for a clothing store chatbot, for most of the clothing parts the user will need to choose a size and in some cases the color of the item. So, to facilitate the user selection, when the user textually confirms to the chatbot that he/she wants to add an item to the cart, the chatbot will prompt a **message composed of buttons** with all the sizes available for that clothing.

This feature will be useful for clothing that uses different measurements. For example, some clothes might be marked with a number and other with letters (S, M, L, etc) or, depending on the country, the user might be used to some measurement rule from his country that is not that same used in the website. In this way, the error of the user inserting a value that is not valid is avoided.

Figure 4.2 shows an example of the chatbox, in this case the use of the size buttons when the user selects an item to buy.

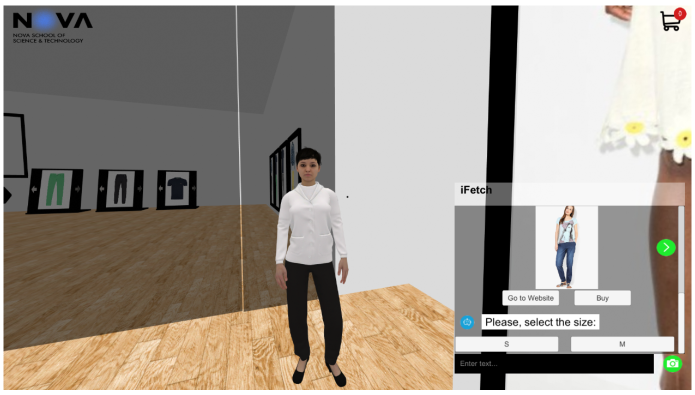

Please, select the size:

Figure 4.2: Use of buttons and recommended items in chatbox

Another type of message consist in a **list of recommended items**. Each item has some shortcuts and the user can also click on them to see its full details. This type of message is similar to the one presented in [26].

The chatbox also has a type of **message used to confirm the item** that the user is referring to. If the user responds "No", a list of items will be shown to the user and he/she will have to select one of them. This message types is used to solve a problem that will be explained in the next sub-element. Figure 4.3 shows an example of it.

One last functionality that the chatbox has is the ability to **input images** from our device, using the green camera button in the corner of the screen. This will open the device explorer and the user will need to select a photo.

Recover Items in Field of View: During the development of the backend of the chatbot (work done by other colleagues, presented in section 4.5.1) and thinking

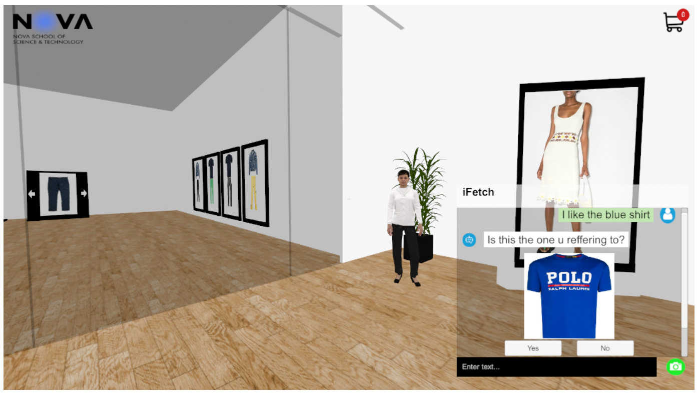


Figure 4.3: Confirm item message

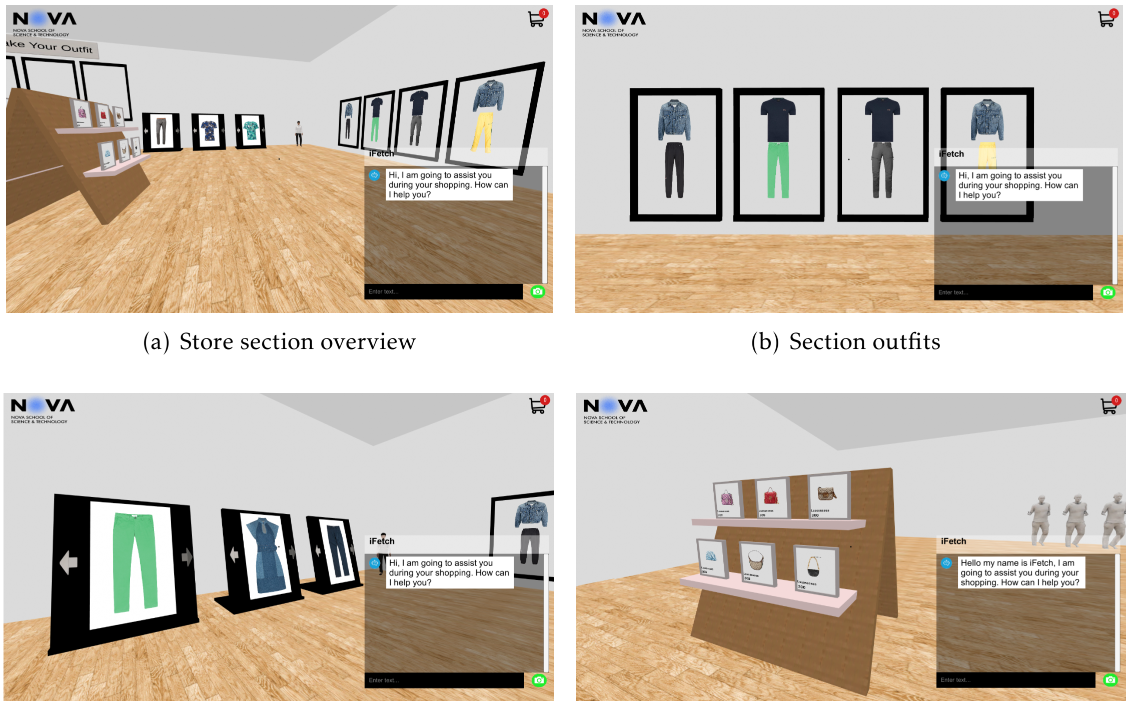
of the later integration of with the interface, emerged a problem that needed to be solved. During a conversation, and thinking of the fact that there are multiple items around the store, it might happen that the user says something like: “I want to see more pants like this ones” or “I like the color of this shirt”. In these two quotes, the user should be referring to an item that he/she is seeing in the interface, either in the store sections or recommended items that show in the chatbox, but the backend has no information about which one the user is referring to. The information that the backend might have is all the items that are in the store at that moment.

To help solve this problem, an algorithm was developed in order to return all the items that are in the user’s field of view. In this way, instead of returning all the items that are currently in the store, the pool of items is significantly reduced, making it easier to search for the item the user is referring to.

After the backend process it, a confirmation message like in figure 4.3 will be sent to the user. If the user says "No", a list of the items that are in its filed of view will be shown in the chatbox and the user will be asked so select the one that he/she was talking about.

Store Section: the interface was designed so that there are multiple sections in the store. A section is composed of: some pre-defined outfits; display screens that show some section items: those can be the current trends or items that are on sale, for example. There are also some shelves used to display items like jewellery or purses and the user can click on each item to see its details. An example of a section is shown in figure 3.4(a) or with more detail in figure 4.4.

Store Attendant: as the user enters the store, there is an attendant that follows him/her through the store, so the user can have a conversation in each section. In most of the images of this of the interface, can be seen a woman that represents the chatbot and is following the user.



Hi, I am going to assist you during your shopping. How can I help you?

Hello my name is iFetch, I am going to assist you during your shopping. How can I help you?

(a) Store section overview (b) Section outfits

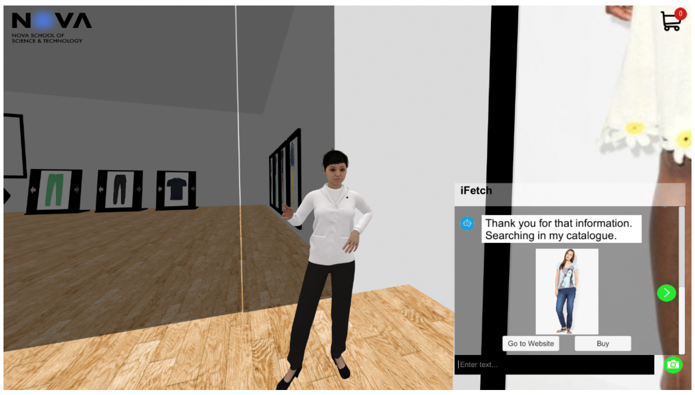
(c) Section screens (d) Section shelf

Figure 4.4: Store section - different features

This was achieved using *Unity Navigation System* [1]. According to the Unity documentation, the navigation system is composed of three main components:

– ***NavMesh***: "a data structure which describes the walkable surfaces of the game world and allows to find path from one walkable location to another in the game world".

[1] https://docs.unity3d.com/Manual/nav-NavigationSystem.html



Thank you for that information. Searching in my catalogue.

Figure 4.5: Store attendant

- ***NavMesh Agent***: "help you to create characters which avoid each other while moving towards their goal. Agents reason about the game world using the *NavMesh* and they know how to avoid each other as well as moving obstacles".
- ***NavMesh Obstacle***: "allows you to describe moving obstacles the agents should avoid while navigating the world. A barrel or a crate controlled by the physics system is a good example of an obstacle".

An example of a scene and the different components of the *NavMesh* can be seen in figure 4.6.

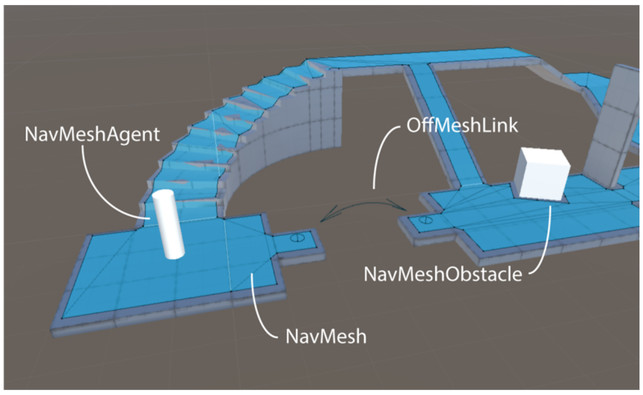


Figure 4.6: NavMesh overview

Recommendation Wall: apart from each store section, there is a common place where items recommended to the user and all the ones that the user marked as interested are presented. There, the user is able to see the items side by side and combine them in an outfit to see if they go well together (eg., if some pants go well with a t-shirt). This can be done by clicking and dragging an item to the intended outfit. If the items are t-shirts, trousers or shorts they can be seen in the section 3D mannequins. For being able to use those 3D mannequins, it was needed to use the Pix2Surf clothing texture generator, using pre-processed clothing images. This processed is explained in more detail in section 4.3

An example of this section can be seen in figure 4.7 and figure 4.8. On the bottom side of the wall in figure 4.7 there are three frame with the recommended items with arrows on the sides so the user can choose the one he/she intends. Then, the users drag it to the outfits on the top and when the mouse is released, the texture will be saved on the corresponding outfit and mannequin on the side.

Mannequin Details: this is part of the recommended wall section. Here, the user can customise the mannequin outfit in an easier and quicker way. On the right side,

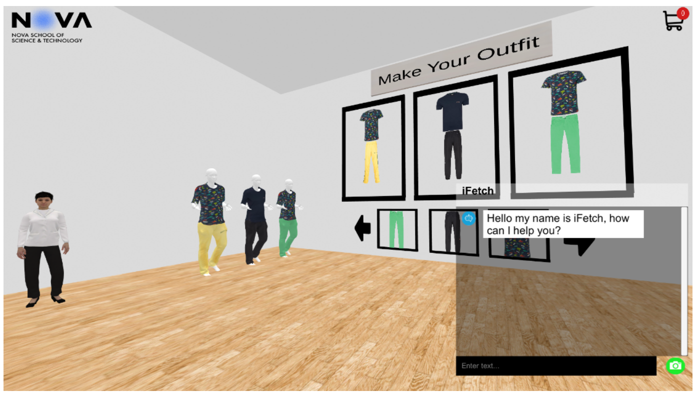

NOVA
NOVA SCHOOL OF SCIENCE & TECHNOLOGY
Make Your Outfit
iFetch
Hello my name is iFetch, how can I help you?
Enter text...

Figure 4.7: Recommended Wall Section

there will be all the items that belong to the recommended section, divided in bottom and top clothing parts. The user can select the item that he/she intends to see in the mannequin. This can be seen in figure 4.9.

Recommended Clothing: in the chatbot, there are two ways that some items can be suggested to the user: by the understanding from the chatbot of which clothing the user wants by the textual description the user gives, or by finding similar clothing to an image that can be upload via chatbox. As mentioned above, those items are displayed in the chatbox and in the store's recommendation wall. Figure 4.2 shows the recommended items in the chatbox on the right corner.

Clothing Details: for each item that the user has access to in the interface, he/she is able to click it to see its details and some action buttons that show in a card on the left of the screen. There the user can see more pictures of the product, its price, brand and description. He can also choose a size and add the item to the cart, add it to an

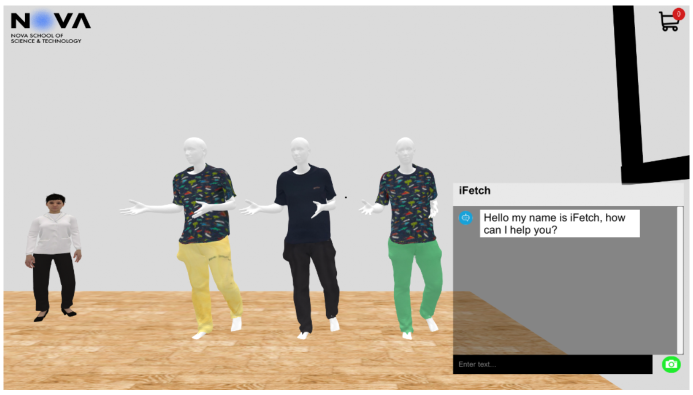

NOVA
NOVA SCHOOL OF SCIENCE & TECHNOLOGY
iFetch
Hello my name is iFetch, how can I help you?
Enter text...

Figure 4.8: Recommended mannequins

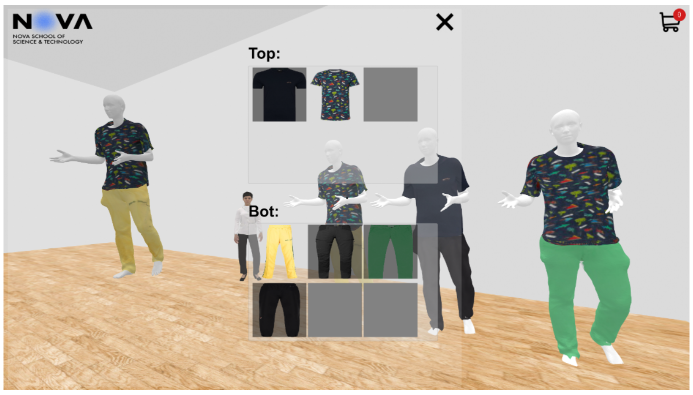


Figure 4.9: Combine outfits in mannequin

outfit from the recommended wall, mark it as interesting (consequently adding them to the recommended wall too) or visit the product website for more information. In figure 4.10 displays details of an item that was recommended to the user.

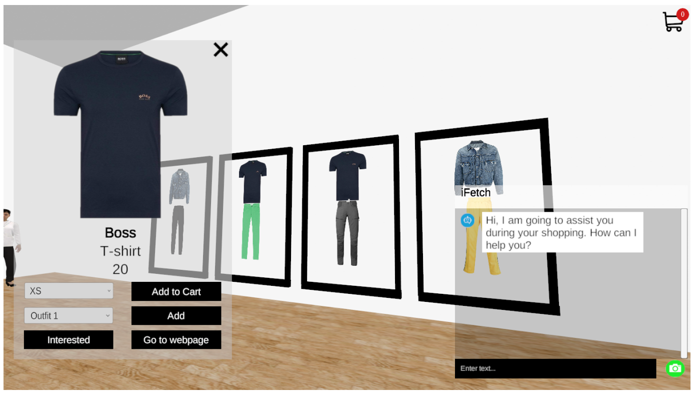
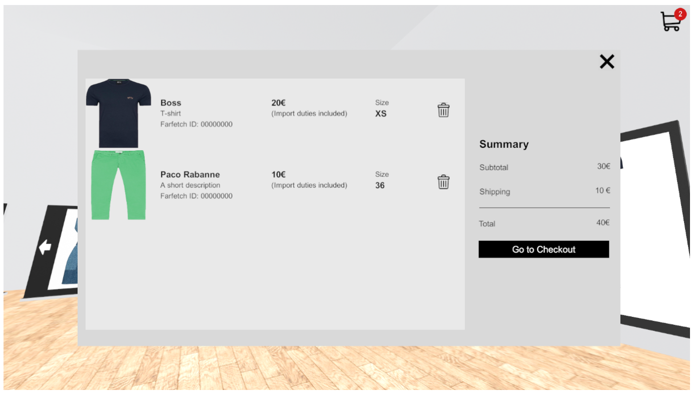


Figure 4.10: Clothing Details

User Cart: this application also has the functionality of saving the items that the user wants to buy to their shopping cart. In the cart, the user can see all the items that he/she wants to buy, their prices and the final price of its order. The user can always delete an item from the cart.

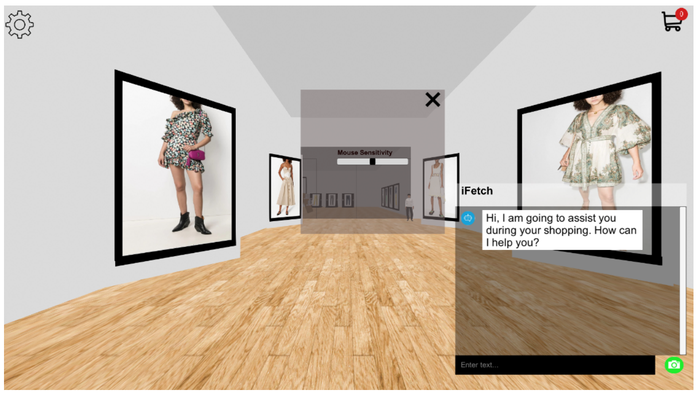
Sensibility Settings: After the final user testing phase that is be presented in section 5.4 some users suggested that would be useful if the interface had the ability to select the mouse sensibility due to the fact that the user was not used to the default value that were used. To solve this, a settings box that contains a sliding bar for the sensibility of the mouse was added. It can be seen in figure 4.12.


Boss
T-shirt
Farfetch ID: 00000000
20€
(Import duties included)
Size
XS

Paco Rabanne
A short description
Farfetch ID: 00000000
10€
(Import duties included)
Size
36

Summary
Subtotal 30€
Shipping 10 €
Total 40€

Figure 4.11: User cart



Hi, I am going to assist you during your shopping. How can I help you?

Figure 4.12: Sensibility Settings

4.2 Environment Setup - Web Server

As this project main objective was to implement an interface for a chatbot that was under development, at the star of the development process there was not yet been developed any way of communication and exchange of information between the front and backend. This information could be for setting up the store, exchanging messages with the chatbot or getting information about a product.

Therefore, to facilitate this communication, it was developed a *Python* web server for exchange of messages and section information to setup the interface. The implementation of this Web Server was accomplished using Flask [2]. Flask is a micro web framework written in *Python*. It is considered micro because it doesn't require particular tools or libraries.

4.2.1 Endpoints

The web server has the following endpoints:

[2] https://flask.palletsprojects.com/en/2.0.x/(palletsprojects.com)

/next-message: this endpoint is used to get the next message from the list of messages saved in the JSON file.

/has-message: this endpoint is used to check if there is any new message from bot

/new-message: this endpoint is used to notify the server of a new user message

/section-items: this endpoint will return all the information needed to setup each section.

/image/<path:filename>: given the file name, this endpoint will return a jpeg of the corresponding image.

/outfit-image/<path:filename>: given the file name, this endpoint will return a png of the corresponding image.

/manequim/<path:type>/<path:filename>: given the type (top or bot) and the file name, this endpoint will generate the input images for the generation of the 3D mannequin texture. This algorithm will be later described in more detail in section 4.3.

4.2.2 JSON Structure

All the information used in this webserver is extracted from a JSON file. This file has two main items: the chatbot sequence messages and the section information. The file is composed of the following items:

- **Messages:** list of conversation messages that were extracted from an example conversations between a user that is looking for some clothing items and an attendant [30]. This conversation is a sequence of messages that can either be text or images that represent the recommended items. A message has the following structure:

```
{
  "type": "images",
  "from": "Bot",
  "content": "Text content",
    "recommended": [
        {"ID": 1,
      "URL":"",
       "price": 10,
         "description":"A short description",
       "brand":"Paco Rabanne",
       "type":"bot",
       "sizes":["36", "38", "42", "44", "48"],
         "files":["1.jpg"],
       "manequim":"1_low.jpg"
       }
    ]
}

```

The *type* of the message identifies if it is either a text message or if it contains recommended items. *From* is used to identify the origin of the message (user or bot). *Content* is the actual text of the message. Finally there is a list of recommended items.

- **Item:** Each clothing item in the JSON file is represented by its *ID*, *price*, *description*, *brand*, *type* (top if it is a shirt, jacket, etc, bot for trouser, pants or shorts, dress, bag, etc.), the available *sizes*, the list of file names of the item images and for last, if available, the image that represents the texture to be applied to the 3D mannequin. The endpoints **/image/<path:filename>** and **/outfit-image/<path:filename>** are used to obtain this images.

- **Section Items:** for each section in the store, there is a JSON object composed by two objects: a list of items that will show in the section screens (figure 4.4(c)), and the section pre-defined outfits (figure 4.4(b)). Each outfit is identified by the bottom and top clothing IDs.

```
"section-items":{
      "items":[
             {
                "ID":1,
                "URL":"",
                "price":10,
                "description":"A short description",
                "brand":"Paco Rabanne",
                "type":"bot",
                "sizes":[
                   "36",
                   "38",
                   "42",
                   "44",
                   "48"
                ],
                "files":[
                   "1.jpg"
                ],
                "manequim":"1_low.jpg"
             },
             {
                "ID":2,
                "URL":"",
                "price":10,
                "description":"A short description",
                "brand":"Paco Rabanne",
                "type":"bot",
                "sizes":[
                   "36",
                   "38",
                   "42",
                   "44",
                   "48"
                ],
                "files":[
                   "2.jpg"
                ],
                "manequim":"2_low.jpg"
              }
         ]
    {
    "outfits":[
        {
            "top":7,
            "bot":4
        },
      {
          "top":7,
          "bot":3
      }
    ]
}

```

4.3 Clothing Models

Following the work with 3D models from [21] presented in section 2.2, *Pix2Surf* was used to generate the 3D clothing models to improve the user experience. The clothing models allow the user to see the clothing as he/she would do by wearing them in the store so that he/she can, for example, check if a shirt goes well with some jeans. The user can easily combine different parts of clothing with others. Figure 4.8 shows an example of the final result of this implementation.

As some brands might not agree with this way of displaying the items with the 3D models, due to image manipulation and as this technology transform 2D images in 3D and it might not achieve perfect results of texture quality, this was considered a proof of concept to extend the interface and to demonstrate this new functionality. The 3D models could be used to replace the walls where the items will be displayed or to combine both approaches, as seen in figure 4.7.

Pix2Surf already has the mannequin 3D objects that could be imported to the project. For the generation of the clothing textures to apply to the 3D model, *Pix2Surf* already has some pre-trained CNN model that, given a front and back picture of the clothing, will generate a .jpg texture to apply to the mannequin body part. An example of a mannequin and clothing meshes can be seen in figure 4.13.

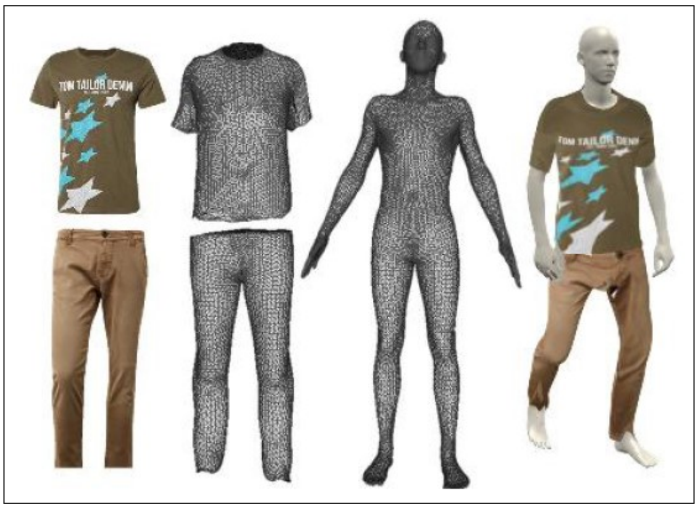
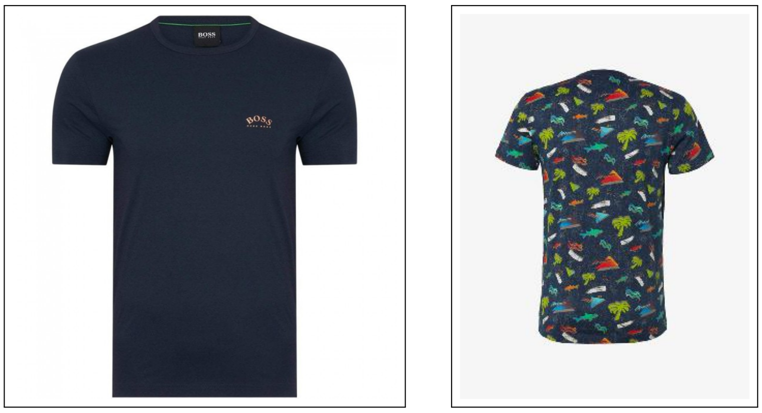


Figure 4.13: Clothing meshes and mannequin models, adapted from [21]

4.3.1 Pix2Surf Setup Requirements

To be able to get the best results of the pre-trained *Pix2Surf* models, it is important to take some things into consideration: firstly, all the images should have the same measurements and proportions, so there is the need to do some pre-processing to the dataset in order to achieve the best results. Figure 4.14 shows the differences between an image from Farfetch and an image from *Pix2Surf*.



(a) Farfetch Image (b) Pix2Surf Image

Figure 4.14: Comparison between Pix2Surf image and Farfetch image

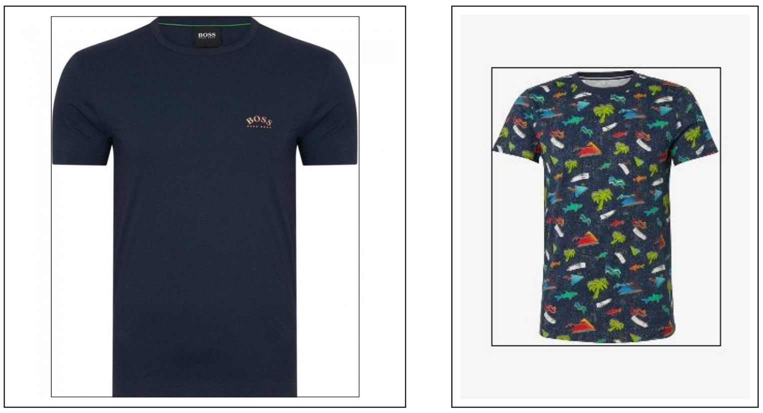
Secondly, there was a problem regarding the back images of the clothing items: to be able to generate the mesh texture it is necessary to have the front and back images of the clothing item, but most of the clothing datasets (including Farfetch website) don't have back images. All the images need to be of the clothing item with a white background, with no person wearing them. So, it is necessary to, given the front image of a clothing item, generate an image to use as the back of the clothing.

As referred in section 3.3, all the pre-processing was done in Python, using libraries like OpenCV[3], TensorFLow [4] and Keras [5]. To facilitate the libraries management, the implementation was done using Google Colab[6].

4.3.2 Re-scale Front Image

The re-scale of the clothing image has three steps: **extract clothing**, **re-scale it** and **add white borders**.

The algorithm starts by obtaining the actual size of the clothing in the Pix2Surf and Farfetch images. For that, was used an algorithm that goes over all the image pixels and finds out the minimum and maximum X and Y values where the pixels are different from the background. An example of this can be seen in figure 4.15:



(a) Farfetch Image (b) Pix2Surf Image

Figure 4.15: Clothing bounding box

Then, the clothing part of the image is re-scaled to the size of the Pix2Surf clothing. At the end, the algorithm adds some borders to the image to make it the same size as the expected. The final result can be seen in figure 4.18:

4.3.3 Generating Back Image

The generation of the back image comes after the re-scaling of the image. There are two main cases that were needed to identify when generating the back images: the image could be mainly plain, or it may be an image with patterns. An example would be the images from figure 4.18: image (a) is considered plain and (b) a clothing with pattern.

[3] https://pypi.org/project/opencv-python/
[4] https://www.tensorflow.org/
[5] https://keras.io/
[6] https://colab.research.google.com

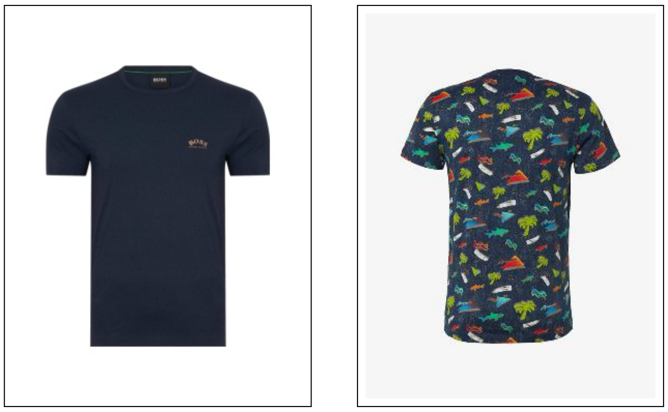

(a) Farfetch Image (b) Pix2Surf Image

Figure 4.16: Comparison between Pix2Surf image and Farfetch image after scale

A good example to understand the problem of generating the back image are t-shirts like in figure 4.17. For the back image of this clothing item, it would be preferable if it was possible to remove the center image and have an image with the same silhouette and only the main color of the image.

This is a problem that mostly applies to top clothing parts, like shirts, t-shirts or polo shirts and not to trousers or pants. In these last two, the back part is usually equal to the front part, so in these cases it can be duplicated.

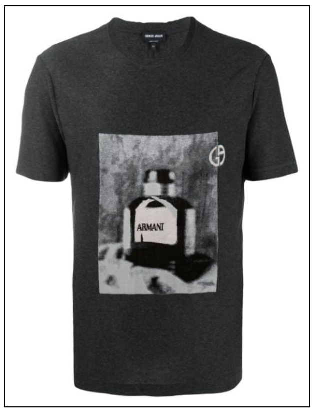


Figure 4.17: Plain clothing with image

On the other hand, for images like figure 4.18(b), there is no main color that can be extracted to generate the back image. So, the best way that we have to generate the back image, given only the front one, would be to duplicate the front image. With this, it would be maintained the texture pattern.

This would be an easy way to solve the problem, if all the images that were going to use in this project (e.g., a dataset from Farfetch) were all labeled as plain or pattern images. As the clothing items are not well labeled, there was a need to predict the clothing tag and generate the back according to it. The next subsection will discuss the methods that were used to categorise the clothing images.

4.3.3.1 Analysing Images Color Histograms

For the prediction of the clothing type, several approaches were used. The first and the technique that showed to be the most obvious was **analysing the images color histograms**. The histograms represents the count of pixels that have a certain color. Given that all the images were pre-processed and have the same proportions and number of pixels, the first reasoning was finding images that would have a color with a count above a certain threshold.

Figure 4.19 shows some of the histograms obtained for three different type of shirts.

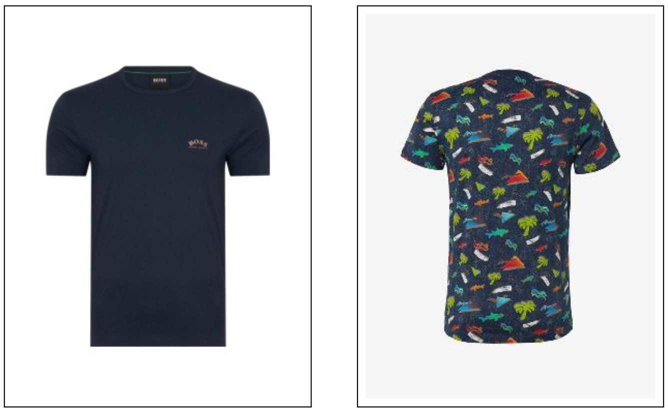

(a) Farfetch Image (b) Pix2Surf Image

Figure 4.18: Comparison between Pix2Surf image and Farfetch image after scale

Figure 4.19(a) represents the typical image that is considered a plain image. The obtained max count on this histogram is equals to 6827 pixels. On the other hand, figure 4.19(b) represents another plain clothing, but in this case, the max count is equal to 2580. This happens because of the way the two clothing reflect the light: the second image reflects more light than the first.

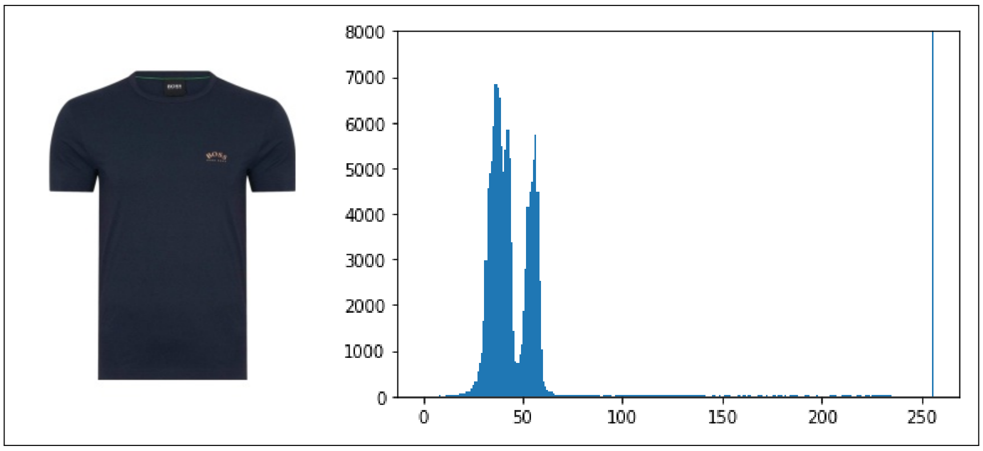
The third clothing item represents an item that is considered to be a pattern clothing because it contains some strips that need to appear on the back of the clothing. The max count of pixels is higher than the previous image, with a count of 3638.

With this, was concluded that using only the threshold for splitting the images in their categories is not the way to achieve the best results.

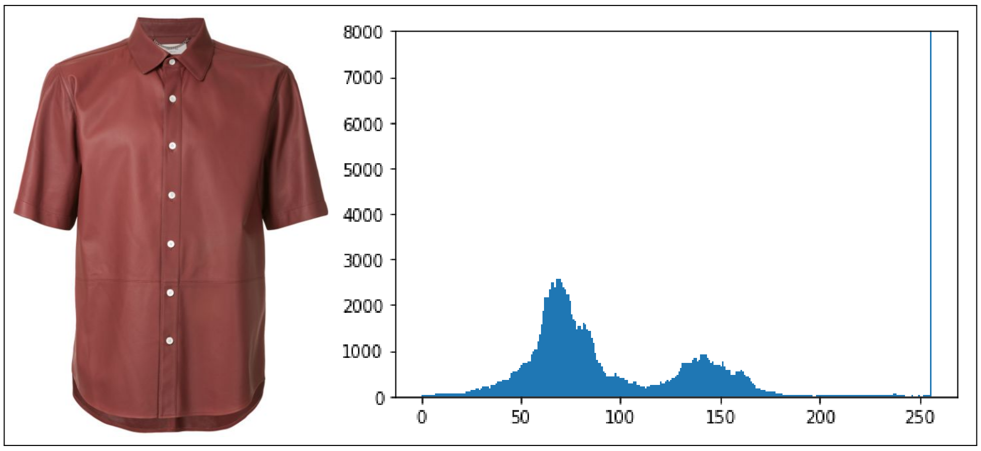
4.3.3.2 Clustering Algorithms

Another approach that was used with no success were **Clustering Algorithms**, like *DB-SCAN* or *Affinity Propagation* from *sklearn*[7]. In short, the way clustering algorithms work is finding the core samples of high density that best fits the data and expanding them

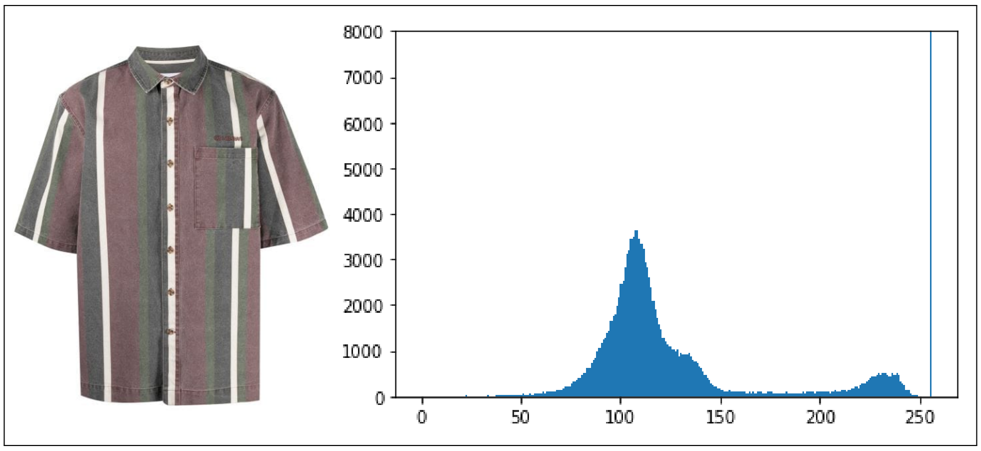
[7]`https://scikit-learn.org/`



(a) Plain image - 1



(b) Plain image - 2

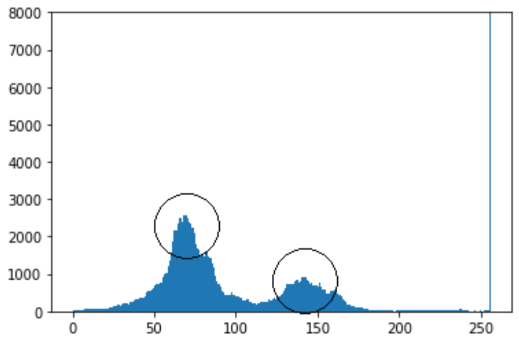


(c) Pattern image

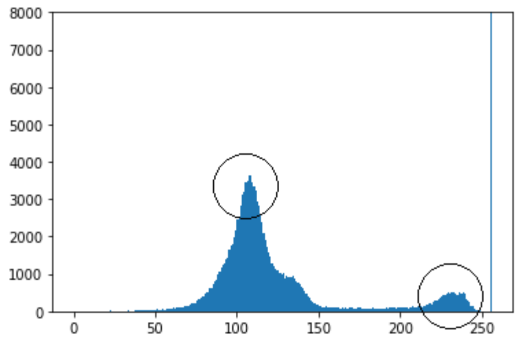
Figure 4.19: Clothing Images and Histograms

to the nearest neighbours. The goal was to split the images in categories by the number of clusters: the way of thinking was that plain images would have less clusters than the pattern ones, because they usually have less colors: normally two clusters, one for the background pixels, other for the shirt ones.

But, as seen above and in figure 4.20, this technique can't be used to solve this problem: for a plain and a pattern image, because two different clusters can be detected for both of them.



(a) Plain Image Histogram



(b) Pattern Image Histogram

Figure 4.20: Images histogram analysis

4.3.3.3 Convolution Neural Networks

The approach to this problem that turned out being the one that achieved the best results was using CNN. To develop this CNN was necessary to obtain a large enough dataset but with enough diversity of shirts to avoid overfitting, label it and use it to train the network.

Dataset: to obtain the clothing images for the dataset, was used a Python library called BeautifulSoup[8]. BeautifulSoup is a Python package used to analyse HyperText Markup Language (HTML) and Extensible Markup Language (XML) documents.

It returns the source HTML code for the webpage of a given Uniform Resource Locator (URL) as an object. Then, it was needed to manually analyse the HTML page to find which components were used to save the clothing items to be able to extract the images URL. For this dataset, different clothing categories from the Farfetch website were used, including shirts, t-shirts and polo shirts.

An example of a Farfetch URL would be: "https://www.farfetch.com/pt/shopping/men/t-shirts-vests-2/items.aspx?page=1view=90".

As the website clothing categories are composed of several items pages, the URL page number could be changed to obtain the next page and extract new items. All those images were saved and then manually categorised into plain or pattern clothing items.

After gathering all the clothing images, it was concluded that there was a discrepancy between the number of plain and pattern images, being the number of plain images around four times superior then the number of pattern images. To equalise the

[8]https://www.crummy.com/software/BeautifulSoup/bs4/doc/

dataset, some of the plain images were discarded, in a way that similar images were ignored, to have a dataset with the most diversity possible, in order to obtain the best results and avoid overfitting.

The final result of the web scrapping of the Farfetch dataset is shown in the following table:

	Plain	Pattern
Total Images	4008	1503
Training Set	1002	1002
Test Set	501	501

Table 4.1: Dataset image's composition

Create and Train the Model: for the implementations of the CNN was used Keras from Tensor Flow. The network was implemented with the structure presented in figure 4.21.

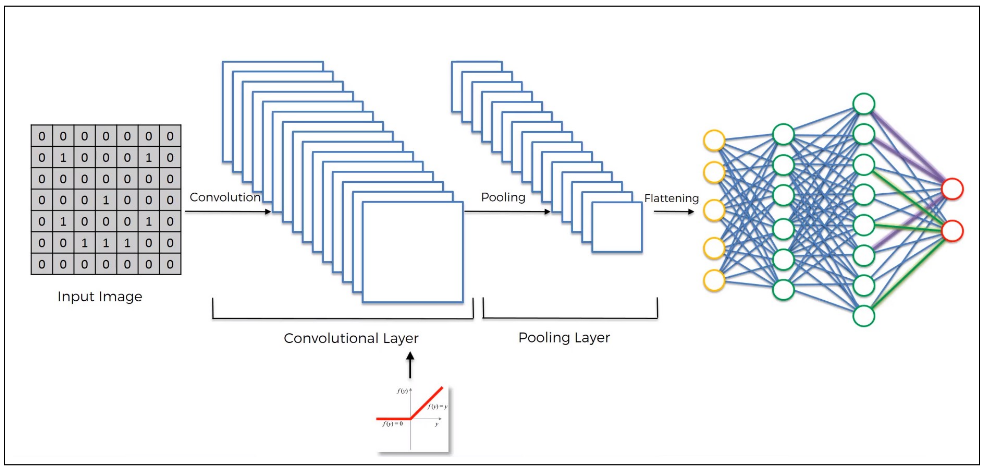


Figure 4.21: CNN structure, adapted from [11]

The first step of the CNN is the Convolution Layer. This is the feature extraction step where all the images will be processed and generated feature maps. By reducing the image size but at the same time, maintain the relevant parts of the images to be analysed, some non-relevant information of the image will be lost.

Next, there is a second layer called Max Polling where, given the extracted features, it will reduce them by pooling the important features and discarding non important values. With this, the number of parameters that go to the last layers of the network is reduced and avoid with that, avoiding overfitting.

The process of convolution and pooling is then repeated with another two layers.

The next layer of the network is a flattening layer. As this work is about images in 2D arrays, there is the need to flatten those arrays to input to the next layers of the network.

The next layer is composed of two fully connected (dense) networks. The first one contains 256 units with a ReLU activation function. The last one has only one unit with Sigmoid activation function, as the goal is to predict the category of the image between only two categories.

The last step for the CNN is the training. For this, we used the gathered dataset, splitted in train and test and 25 epochs.

Experimental Results: after the training process, the model achieved the following results:

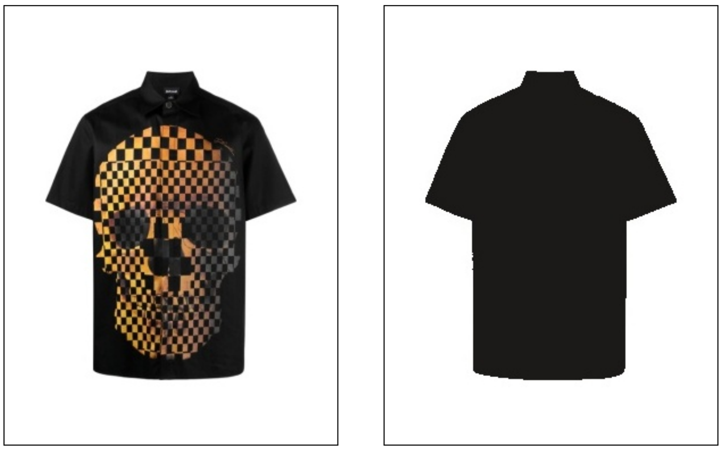
Loss	Acc	Mean Sqr Err	Val Loss	Val Acc	Val Mean Sqr Err
0.0801	0.9685	0.00224	0.7082	0.8554	0.1222

Table 4.2: CNN experiment results

The model achieved an high result of Validation Accuracy, however, in the third epoch the model achieved a higher validation accuracy (0.8745), what means that the model converges quickly. This happens because the dataset that were used is small, and easy to process.

4.3.3.4 Background Generation Result

An example of result of the process of generating the back image from a a plain clothing can be seen in figure 4.22:



(a) Front image (b) Back image

Figure 4.22: Front and back images after all processing

4.4 Background Removal

In the recommended wall, the user can combine the clothing items into outfits and see them in a frame. As all the images have a white background, and some of them with different shades of white, they have different shapes (some horizontal, others vertical), it wouldn't look that great in the frame. To solve these problems, it was used background removal algorithms combined with the clothing extraction as seen in section 4.3.2.

For the background removal, several methods were to achieve the best results: using GrabCut [29], morphological operations and using CNNs [12]. All the implementation and tests were done using Google Colab [9].

Here it is presented the work done with clothing parts like t-shirts, but the same reasoning and algorithms can be applied to every clothing part.

4.4.1 GrabCut

As seen in section 2.4, GrabCut is a semi-automated algorithm for removing the background of an image. It receives as an input the source image and a rectangle that surrounds the object to be extracted. For that, it was used the technique of clothing extraction seen in section 4.3.2.

4.4.2 Morphological Operations

This technique for background removal uses operations like Opening and Closing to manipulate the image, and also applies Blur to the images [10]. More details about this technique were given in section 2.4.

4.4.3 Convolutional Neural Networks

The last technique that was used for the background removal follows the approach from [12]. It uses a ready to work pre-trained model [25] with the same principles from Tiramisu presented in section 2.4.3.

This technique has two steps: a first pre-processing step that "detect the boundaries of objects in a photograph, cut them out, sequentially remove the background from each object in turn and subsequently collect the entire image from separate parts"[25] using neural networks. The second step called post-processing is used to improve the boundaries of the images obtained from the neural network processing.

The user has the option choose from more then one processing method according to his necessities: some methods are faster but with lower image quality, or it can be chosen a method with better quality but slower.

[9] `https://colab.research.google.com/`

[10] `https://docs.opencv.org/4.5.2/d9/d61/tutorial_py_morphological_ops.html`

4.4.4 Methods Comparison

Figure 4.23 shows a comparison between the different methods that were used. Was concluded that the method with the best performance was using CNNs, where all the clothing objects were extracted successfully. This method has the drawback of being much slower than the other two.

With GrabCut, part of the clothing close to the edges was missing but the center region was all detected as being part of the clothing. It also doesn't work well with clothing.

On the other hand, using morphological operations, it was proven to don't work well with white spots in the middle of the shirt, marking them as part of the background, but maintaining most of the shape of the clothing. It also doesn't prove to work well with white clothing as the clothing is confused as part of the background due to the similar color.

For images considered plain (second row from figure 4.23), the GrabCut also has problems with the edges, but the morphological operations can almost perfectly detect the t-shirt. With the CNN method the best results are obtained.

4.5 System Integration

An integration with the backend of the chatbot after the development of the first interface prototype. At that point, the interface was able to exchange messages with the backend and receive the responses. Those responses could be pure text messages or recommendation messages that would contain a list of items to be shown to the user.

4.5.1 Backend of Chatbot

In the scope of the iFetch project, it was also developed a backend for the chatbot. This work was part of other colleagues' dissertations, and the main goal was to develop a chatbot backend that is able to maintain a conversation with the user, give advice and help them choose the best products to buy.

In the first stage, the chatbot was only able to fulfil a simple task. It works like a slot filling [33], and will ask the user for his/her gender, their age, the product he/she is searching for and the color. then, it will send to the user a list of items according to the specification.

This work was integrated into a web server to be able to communicate with the interface. Every time the user writes a new message, a web request with the following body is sent to the server:

```
{
    "belief_state": {
        "attribute_values":{}
    },
    "current_image": null,
    "decoder": "hand_templates",
```

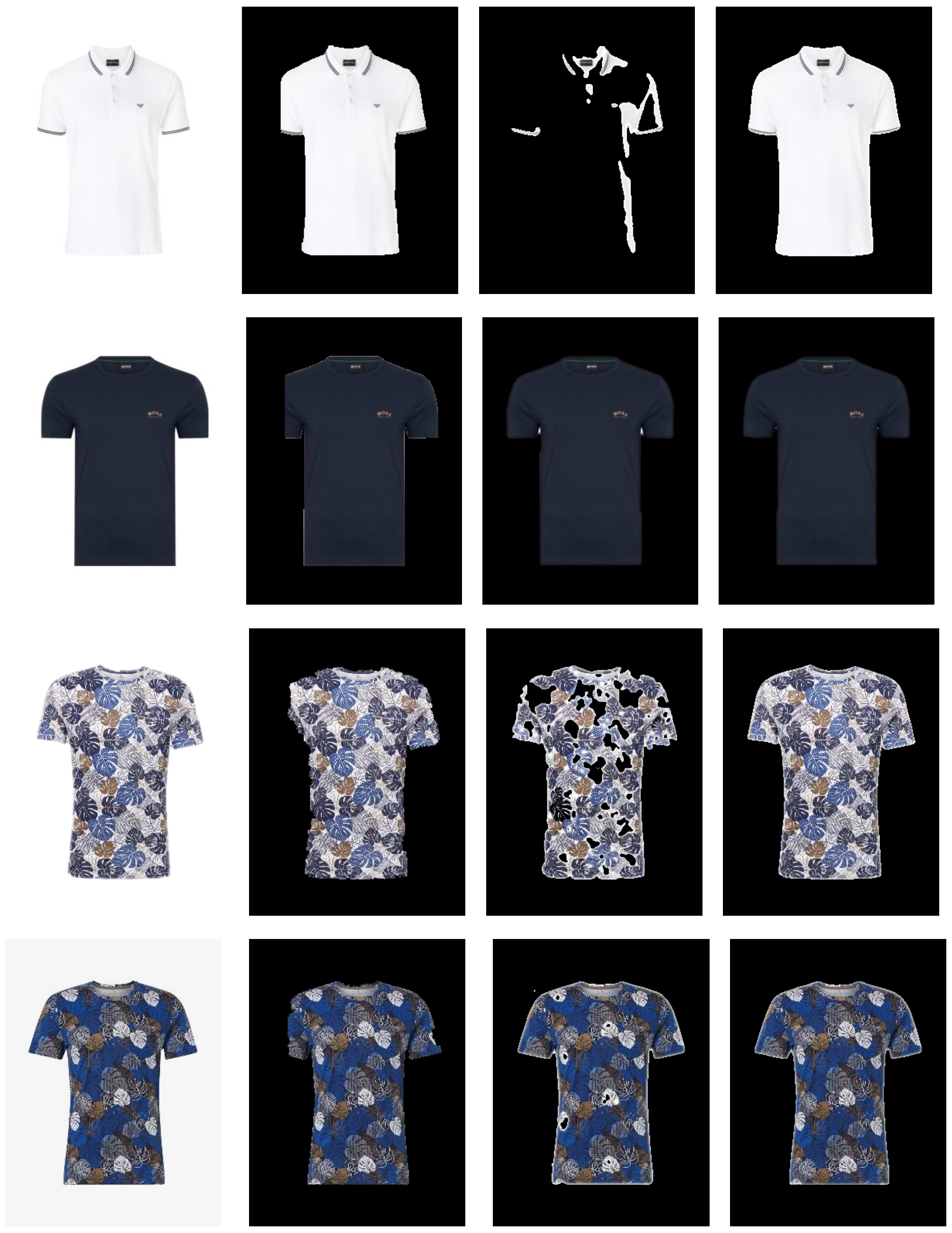

(a) Original (b) GrabCut (c) Morph. Opt. (d) CNNs

Figure 4.23: Comparison between different techniques of background removal

```
    "dst": "simpletod",
    "policy": "rule_policy",
    "policy_state": {
        "asked_age": false,
        "asked_gender": false,
        "asked_synset": false
    },
    "status": "success",
    "turns": [
        {
            "dialog_act": "GREETING",
            "extracted_info": null,
            "images": null,
            "intent": null,
            "speaker": "system",
            "utterance": "Hello my name is iFetch, how can I help you?"
        },
        {
            "dialog_act": null,
            "extracted_info": null,
            "images": null,
            "intent": null,
            "speaker": "user",
            "utterance": "I am looking for blue jeans that I can wear"
        }
    ]
}
```

As a response, the interface will receive a message with the following body:

```
{
    "belief_state": {
        "attribute_values": {
            "color": "blue",
            "synset": "jeans"
        }
    },
    "current_image": null,
    "decoder": "hand_templates",
    "dst": "simpletod",
    "policy": "rule_policy",
    "policy_state": {
        "asked_age": false,
        "asked_gender": true,
        "asked_synset": false
    },
    "status": "success",
    "turns": [
        {
            "dialog_act": "GREETING",
            "extracted_info": null,
            "images": null,
            "intent": null,
            "speaker": "system",
            "utterance": "Hello my name is iFetch, how can I help you?"
        },
        {
            "dialog_act": null,
            "extracted_info": null,
            "images": null,
            "intent": null,
            "speaker": "user",
            "utterance": "I am looking for blue jeans that I can wear"
        },
        {
            "dialog_act": "REQUEST gender",
            "extracted_info": null,
            "images": null,
            "intent": null,
            "speaker": "system",
            "utterance": "Do you have any particular gender in mind?"
        }
    ]
}
```

There are several attributes and objects in this JSON body, but the most relevant are:

- **belief_state**: represents the information that the chatbot were able to extract until that moment. In the chatbot response it can be seen that it was able to extract the *synset*(in other words, the type of clothing) and the color.
- **turns**: a list of all the messages from the conversation.
- **policy_state**: this object is used for the chatbot to track which questions were already made to the user. In the chatbot response, it can be seen that in the last turn the chatbot asked for the user gender and the *policy*_state was updated according to it.

After the first user message that was "I am looking for blue jeans that I can wear", the backend was able to retrieve the clothing type and the color and used it to fill the attributes *color* and *synset* from the *belief_state*.

To be able for the backend to generate the next message, it is required that the interface keeps track of the full conversation and sends it to the backend every time a new message is written by the user. So every web request needs to contain all the conversation messages and every attribute that was already filled/detected by the bot.

4.5.2 Images Dataset

Given the fact that at the moment of this integration, the interface had access to a dataset remotely, this dataset was not pre-process with the techniques described in section 4.3.

Another problem with the current dataset that is integrated in the backend of the chatbot is the fact that some of the items don't have a front image of only the clothing, like the one seen in figure 4.17. Most of the times the images in the dataset are images of a human being wearing those clothing items. With this type of images is not possible to generate the 3D clothing textures.

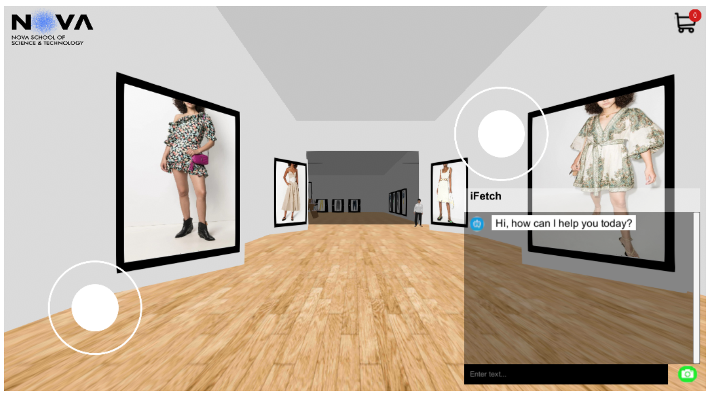

NOVA
NOVA SCHOOL OF SCIENCE & TECHNOLOGY
iFetch
Hi, how can I help you today?
Enter text...

Figure 4.24: Android Application

Given those problems, it was not possible to integrate the recommended items with the "Make Your Outfit"section.

4.6 Android Application

At the end of the development process, the WebGL application was adapted to be able to run on Android devices. From all the features that were added to the store, the only thing that needed to be adapted in this new version was the player and camera movement: in the WebGL version, the user had to use the mouse and keyboard to be able to move and look around. For the Android version, these controls were replaced with screen joysticks, where the left one is used for the player movement and the right one for camera movement. Figure 4.24 shows an example of the application canvas with the joysticks.

Section 5.1 discusses the drawbacks and limitations of the adaptation of this application for mobile.

5 Evaluation and Results

This chapter presents the tests used for evaluating the system and the obtained results. An iterative approach was adopted in the design and development of the system. This means there was an implementation phase followed by a first user testing phase, then implementation according to the user's feedback and then a final user testing phase.

The first section presents results and conclusions of performance testing of the system application.

The second section presents the results of performance tests of the techniques used for background removal.

The last subsection presents the results of the two user testing phases: the first one that used a focus group approach was used. It consisted of a system demonstration in order to validate the system concept and to gather user experience information and functionalities suggestions. The second testing phase took place after the implementation of the improvements suggested in the previous phase. The participants were able to test the interface by themselves.

The last section presents the results of performance tests of the techniques used for background removal.

5.1 System Performance Tests

When evaluating a Unity program, there are several aspects that can be taken into consideration. In the scope of this project, the two main metrics that were used to evaluate the system were the number of Draw Calls and allocated memory for the application.

The Unity plug-in called Profiler[1] was used to evaluate the system. Thhis plug-in let the developer analyse their programs in real time, giving information about the CPU usage, rendering, memory usage and several other categories. Figure 5.1 shows an example of the Profiler interface.

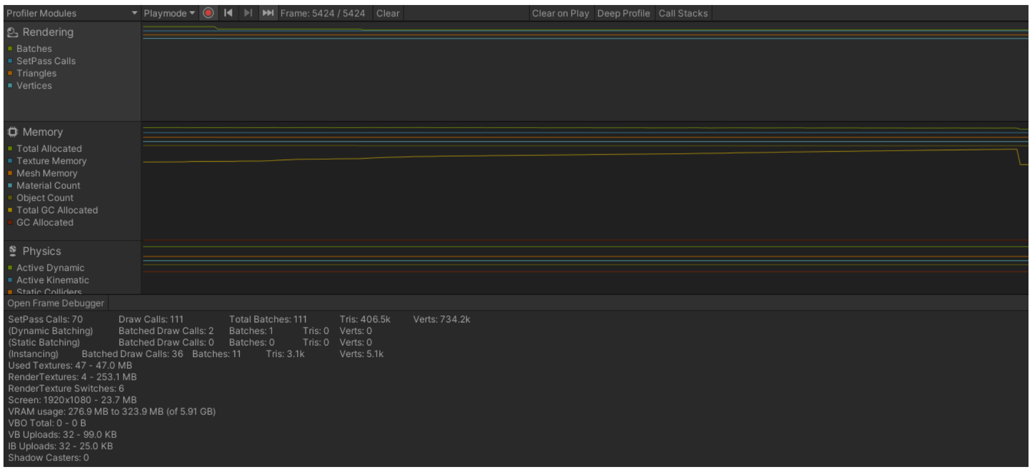
All the results were obtained using an Asus ROG GL702VM computer, with the following specifications:

[1] https://docs.unity3d.com/Manual/Profiler.html

- OS: Windows 10
- Processor: 2.6 GHz Intel Core i7 (4 Cores)
- RAM: 16 GB
- Storage: 250 SSD + 1 TB HDD 7200 rpm

For the test done for the Android Application, was used the a Xiaomi Redmi 5 Plus device with the following specs:

- Android Version: 8.1.0
- CPU: Octa-core Max 2.0GHz
- RAM: 3.00GB



```
SetPass Calls: 70        Draw Calls: 111          Total Batches: 111        Tris: 406.5k     Verts: 734.2k
(Dynamic Batching)       Batched Draw Calls: 2    Batches: 1       Tris: 0  Verts: 0
(Static Batching)        Batched Draw Calls: 0    Batches: 0       Tris: 0  Verts: 0
(Instancing)      Batched Draw Calls: 36   Batches: 11     Tris: 3.1k       Verts: 5.1k
Used Textures: 47 - 47.0 MB
RenderTextures: 4 - 253.1 MB
RenderTexture Switches: 6
Screen: 1920x1080 - 23.7 MB
VRAM usage: 276.9 MB to 323.9 MB (of 5.91 GB)
VBO Total: 0 - 0 B
VB Uploads: 32 - 99.0 KB
IB Uploads: 32 - 25.0 KB
Shadow Casters: 0
```

Figure 5.1: Unity Profiler

5.1.1 Number of Draw Calls

One of the most important metrics to pay attention when evaluating a Unity is the number of Draw Calls the system performs every render cycle. A draw call is issued every time a GameObject is shown on screen. The more objects are currently showing, the higher the number of Draw Calls. The higher the number of Draw Calls, the higher the CPU/GPU time.

Unity has some techniques that can be used to improve the program performance, one of them being Batching. In short, batching is used to group together several objects that are similar and draw them in only one single Draw Call. By similar it means for example having the same texture/material [2].

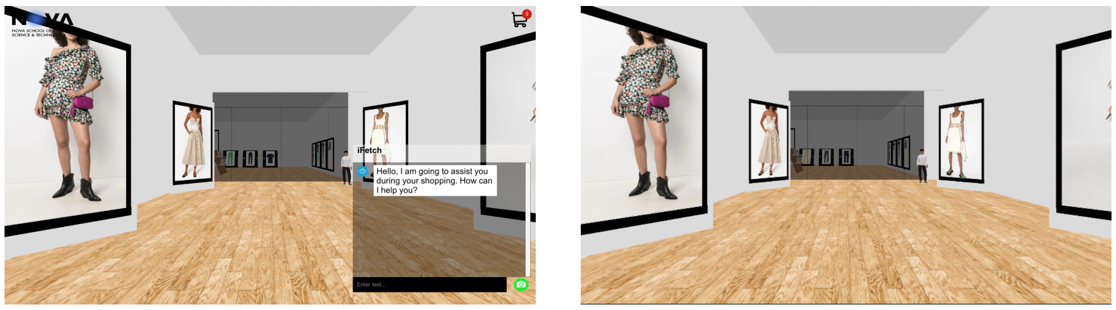
[2] https://docs.unity3d.com/Manual/DrawCallBatching.html

To accomplish this, a pool of colors was used in order to be able to combine the objects together. Another tip that was used was marking as static all the objects that don't move. Those textures can also request for a GPU instancing.

Unity also has a mechanism called Occlusion Culling [3] that only renders the objects that are currently in the field of view of the player, reducing the number of draw per cycle.

Another tip that was used before doing application tests was the illumination: instead of using several light sources like ceiling lamps, it was used the Unity global illumination system.

To test the number of draws per cycle, and given that the occlusion is turned on, there is the need to choose a location in the scene and always look in the same direction. The location that was chosen for this test was the user's initial spawn. With the optimizations mentioned above, the program was doing 111 draw calls per cycle. Then, some of the parent GameObjects were deactivated. At first, when deactivating the canvas it was noticed that it reduced the draw calls by 16.


Hello, I am going to assist you during your shopping. How can I help you?

(a) Store with canvas (b) Store without canvas

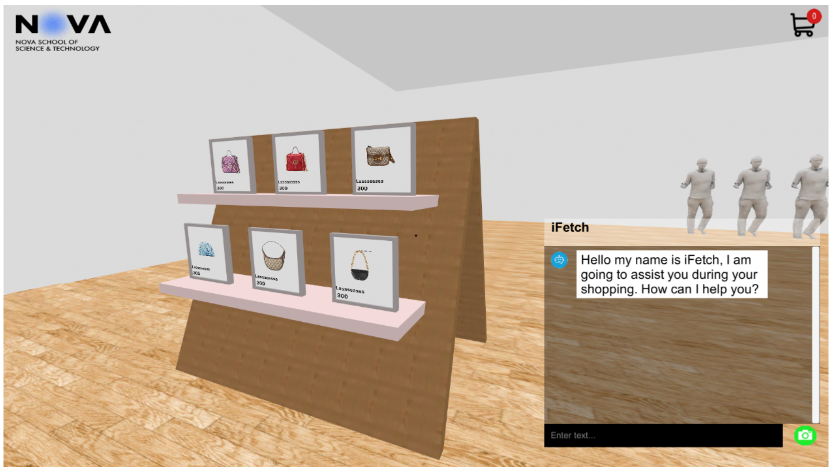
Figure 5.2: Draw calls tests

The object that made the most difference was the items shelf. This shelf is composed by item frames that can be seen in figure 5.3: there is an image and some descriptive text for each item. Every text object and image increase the number of draw calls by 1. For example, in the shelf seen in the figure, there are 12 item frames and each one has 2 text elements. This represents 24 distinct draw calls for the text, 12 for the images. When deactivating this object the number of draw calls was reduced in 38 draws. In runtime, this decrease in the number of draw calls impact the CPU time and consequently the program fps, increasing from a range of 315-330 fps to 330-370 fps.

This problem goes against the initial concept of having several store sections: this would increase the number of clothing parts that needed to be displayed and consequently increasing the number of draw calls, degrading the performance of the application.

When running the desktop version this changes didn't impact the application performance, but when running on the Android device, the joysticks worked poorly with all the

[3] https://docs.unity3d.com/Manual/OcclusionCulling.html



Hello my name is iFetch, I am going to assist you during your shopping. How can I help you?

Figure 5.3: Section item's shelf

application features. When disabling the section shelf and screens the joysticks worked as expected, proving the impact that different images can have on Unity Applications, due to not being able to optimise their processing.

5.1.2 Memory Usage

As this application was designed to run in a web browser, another metric that is important to evaluate in this project is the memory usage. As this application works around fashion products, every product will be represented by at least one image. These images need to be stored with their max resolution to improve the user experience and to show to the user how the products really are in the real world so they can make the right choice on what to buy.

In the current application, and in order to save space and improve the application performance, only two images per product is being used: the image that shows on the screens and the chatbox and the 3D texture. If the user wants to see more about each product, they have the possibility of consulting the product webpage.

The 3D texture only applies for the clothing items that was pre-processed and make part of the initial setup of the store: as seen before, all the items that are recommended to the user belong to an external dataset that was not pre-processed in order to generate those textures.

Using the Unity Profiler, it can be seen that the Unity is using 0.72GB of the system memory and reserved a total of 0.91GB. From those 0.72GB, 0.464GB are used to store textures. In those textures are included the ones from the store items. And as the conversation proceeds, more items will be displayed and more textures will be downloaded and might increase the system memory usage.

But from the analysis done, when the chatbot recommends items to the user (in this case, it recommended 8 different pairs of jeans), the texture memory usage increased from 0.464GB to 0.479GB, a total of 0.015GB for 8 distinct items, which is not that big of a difference. Supposing that, in the worst case, there is still 1GB available for textures (making a total of 1.72GB of system memory usage), the system would be able to have at least 528 different items. This amount of items seems to be an unreachable number of suggested items during a conversation.

From the tests that were done, the memory usage doesn't affect the performance of the application running in the web browser, but it can be a problem when using the Mobile application.

5.2 Background Removal Performance Tests

To evaluate the background removal algorithms, were used images with the same resolution (280x372) of the ones that Pix2Surf used in their 3D clothing generator proposal [21].

As seen in section 4.4, the GrabCut and the method that uses Morphological Operations doesn't work well with items of color white, so for the Performance Tests were used only colourful clothing items. The execution time of each algorithm obtain in the Google Colab was used as a comparison mesure.

The following table presents the Mean and the Standard Deviation values for the three methods that were used:

	Mean, μ	Standard Deviation, σ
GrabCut	1.21700	0.47631
Morph. Opt.	0.65825	0.11560
CNN	30.46975	0.12903

Table 5.1: Background removal performance results

Although the CNN method turned out being much slower then the other two methods, and as seen in figure 4.23, when comparing the image quality of the extracted object, it works perfectly for all the items, keeping all their margins and working well with white clothing.

As the removal of the background is a process to done offline, it is worth spending more time processing the images in a way that it improves the user experience without downgrading the system performance.

5.3 Preliminary User Test

This study occurred after the first system integration with the backend server. It was necessary to understand how the user would appreciate an interface with the functionalities presented and to detect possible problems.

5.3.1 Participants and Evaluation Methods

For this study, the target audience were 9 students and 2 professors from Faculdade de Ciências e Tecnologia, NOVA University of Lisbon, and 3 other participants from outside of the University.

The participants aged from 21 to 55, with an average age of 27.286 and a standard deviation of 9.895, 3 female and 11 male.

This phase was done via online, using the focus group technique, where the users were presented with an interface with most of the functionalities working. Each section took around 10 to 15 minutes.

Firstly it was explained the project that this dissertation makes part of, a description of the work done until that moment, and then a short demo of the interface functionalities was made. A questionnaire took place after the demo.

This can be considered an acceptance testing phase because the main objective was understanding the need that the users have in using chatbots, how they would appreciate an interface with the functionalities that were presented and gather opinions about possible ways of improving the user experience.

5.3.1.1 Results

From this testing phase, it was concluded that there was a higher percentage of participants that prefer using online shops (35.7%) instead of doing it in physical stores (21.4%), but most of the participants say that it depends on the type of product they are willing to buy (42.9%).

Despite that, most of the participants said that they have never used a chatbot to help him in an online store (85.7%), because "the functionality wasn't available" or because "In reality, there were never any difficulties that required the help of a chatbot". From the participants that have already used a chabot, 2 out of 3 say that it was helpful.

Using a scale from 1 to 5 that goes from totally disagree to totally agree, 42.9% answered that if they would need the help of a chatbot in online shopping, they would prefer using an interface like the one presented to them were they were able to interact with the products, agreeing (71.4%) with the fact that a section to combine the products into outfits would help them make a choice in what to buy.

From this section, the participants agree that it would be preferable to use 3D models instead of presenting the clothing only in 2D.

5.4 Final User Tests

In this phase, the participants were able to experiment the WEBGL application in the web browser. It was necessary to test how the user would interact with the interface, in order to recognise problems that the interface might have and ways to solve them.

5.4.1 Participants and Evaluation Methods

For this study, the target audience participants aged from 22 to 56, with an average age of 31.454 and a standard deviation of 13.38, 4 female and 7 male.

In this testing phase, the users had access to URL where the prototype where exposed and were asked to explore the interface and accomplish the following tasks:

- **Cenario 1:** The user will enter the store, select some pants and t-shirt and mark them as interesting (more then 1 from each type). Then, user will look for the "Make Your Outfit"section and combine the different clothing into outfits. After that, he/she can see their outfits displaying on the mannequin. The user can click them and access another way of customisation.
- **Cenario 2:** The user will enter the store, get close to the attendant and ask for blue jeans. Then he will select one of the recommended items and add them to the shopping cart. The user will check if the item was added to the cart.

This testing phase questionnaire included 9 questions from the previous testing phase, excluding some questions that in the first testing phase turned out being redundant. It also included 16 out of the 27 questions from the Questionnaire for User Interface Satisfaction (QUIS)[4], using only the questions that most fitted the presented work, and all the questions of the System Usability Scale (SUS)[5] questionnaire, allowing to do a direct comparison with other systems.

5.4.2 Results

From this testing phase there were several aspects that were tested. One of those were understanding how the interface would work using a different environment. In some cases, one could conclude that the internet connection speed was the main factor that affected significantly the loading of the setup information. This is due to the fact that the interface requires several clothing images that need to be downloaded before the user is able to use the interface. When faced with this problem, the users were asked to try loading the interface with Microsoft Edge browser, since it was the one used during the development of the prototype, and showing to be faster, although not as fast as during the development phase.

To understand better the loading issued, it was also accessed that the interface were not overtaking the computer resources (CPU and RAM).

Despite this loading issue, the users agreed that after the loading phase, they were able to control the interface well issued, being able to accomplish the tasks they were asked to.

[4]Available at: `https://garyperlman.com/quest/quest.cgi?form=QUIS`

[5]Available at: `https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html`

From the target audience from this testing phase, there was a part of them that did not adapt to the controls used in the interface, mainly the fact that they had to use a mouse and there is the need to click on the right mouse button to leave the player movements control and access the mouse cursor. They would forget that the controlling system was disabled and try to move anyways. From those 5 users, 4 were from gender female and one from gender male, age 56, and they are not used to this type of controls that are similar to video games. Also some of them referred that most of the usual users don't usually have a mouse and only use the track-pad, and for those users would be even harder to use the interface. From the users that were used to this type of controls reported that was easy to learn how to operate the system.

One suggestion to solve this problem that was given by one of the users were trying to use a control system that only uses the keyboard and letting the mouse only for clicking in the screen buttons and displays, being able to move and not concerning about the clicks to activate and deactivate the mouse cursor.

Another user mention that he would like to be able to select the mouse sensitivity because he was not used to the default value.

Overall, the users agreed that the system functionalities were well integrated, they were able to easily understand the context and objective of the system and in most cases considered the application tasks straightforward and easy to use. Despite the lack of resources to work with (only text and clothing images), they appreciated the work done and most of they said they would use this system in comparison to the usual chatbox, because it provides more features that could be helpful.

When evaluating the system according to the System Usability Scale, it scores a total of 68.75 that is slightly above the average (68), and can be considered a system that lays between OK and Good. Since part the target audience of this testing phase was expected to not be used to the controlling system, it is expectable that they would decrease the SUS score.

6 Conclusions and Future Work

This chapter presents the conclusions from the work developed and presented in this dissertation and future work and improvements to this solution.

6.1 Conclusion

The main focus of this dissertation was creating an interface for a clothing store's chatbot. This interface uses multimodal information in order to improve the user experience. The main objective was developing something different and better than the usual chatbots that are all over the internet.

A 3D virtual room was developed to emulate the physical store. The user has access to the chatbox to ask their question, and has the possibility to look around the store for possible items that he/she would like to buy. He can also select some items and plan an outfit, being able to see them either in 2D or in a 3D mannequin.

A challenge in the development of this system was the limited resources that were available to be used in the interface: a chatbot for the clothing industry is usually composed of text messages and recommended items, represented by 2D images. With those images, there was the need to process them in order for all the images to have the same size and aspect ratio.

For each item, there were only images from the front, and there was the need to be able to generate a new image that would be used as the back image of the clothing item. A CNN was used for identifying the type of patterns that the shirts have in order to decide how to generate it.

This system was developed using the game engine Unity, and Google Colab for all the image processing, with libraries like OpenCV, numpy, keras and GrabCut.

By analysing this dissertation's document, one can assert that the platform's requirements were fulfilled. Despite the restriction of reserved rights for commercial use from Pix2Surf[21], an integration as proof of concept was made.

From the User Testing phases, although there are some ways of improving the system in order to please all the different users with different experiences, one could conclude that the results were positive and the users appreciated the presented way of interacting with a chatbot.

It can also be concluded that the technologies that were chosen were well suited for this application, although there were some limitations related to Unity, as seen in chapter 5.

In sum, one could conclude that the goals and contributions for this dissertation were achieved but also left space for improvements in some aspects detected in the testing phase.

6.2 Future Work

Although the system requirements have been achieved and implemented, there are some functionalities and limitations that could be improved or added.

One of the improvements that could be made to the system would be the system loading speeds. As seen before, to start the application, there are several images that need to be downloaded in order to display them on the store. Depending on the internet connection speed, it might take several minutes for the store to load, because for every image, there will be a web request to download it. A solution that would compress those images and send in less requests would increase the speed of the loading.

This improvement is also related to the poor performance of the mobile application. The amount of images needed is using a lot of the mobile's RAM and degrading its performance. As the mobiles have smaller screens, one solution could be decreasing the images' size.

As a result of the final system testing phase, it was suggested to implement another controlling mechanism. These controls would use only the keyboard for the movement and looking of the player, leaving the mouse/trackpad for selecting the items on the screen. This system would benefit users who don't have a mouse or don't want to use one.

Finally, as this interface represents a virtual environment of a clothing store, it would also be interesting to try adapting this application to a VR platform. This type of systems allow users to have a more immersive experience and get even closer to the emulation of the process of buying products in a physical store.

Bibliography

[1] Nov. 2020. URL: https://onlim.com/en/the-history-of-chatbots/ (cit. on p. 1).

[2] T. Alsop. *How many people have access to a computer 2018*. Mar. 2020. URL: https://www.statista.com/statistics/748551/worldwide-households-with-computer/ (cit. on p. 10).

[3] D. Bahdanau, K. Cho, and Y. Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *CoRR* abs/1409.0473 (2015) (cit. on p. 10).

[4] M. (Bain. *Virtual body doubles are part of Amazon's plan to keep you out of clothing stores*. URL: https://qz.com/1946328/amazon-launches-made-for-you-program-offering-custom-t-shirts/?utm_source=email&amp;utm_medium=daily-brief&amp;utm_content=10416874&amp;fbclid=IwAR10wEl1Q97jz-gEvQWjYwkVTIGFNswI3sDSJvFhYYkvFiugal0wwNPCNYI (cit. on p. 19).

[5] *Chatbots - The Beginners Guide to Chatbot Technology*. Oct. 2020. URL: https://www.drift.com/learn/chatbot/ (cit. on p. 2).

[6] L. Cui et al. "SuperAgent: A Customer Service Chatbot for E-commerce Websites". In: Jan. 2017, pp. 97–102. DOI: 10.18653/v1/P17-4017 (cit. on pp. 1, 10).

[7] Datahacker.rs. *006 morphological transformations with OpenCV in Python*. Aug. 2020. URL: http://datahacker.rs/006-morphological-transformations-with-opencv-in-python/ (cit. on p. 18).

[8] P. b. S. R. Department and A. 27. *E-commerce share of total retail sales*. Aug. 2020. URL: https://www.statista.com/statistics/534123/e-commerce-share-of-retail-sales-worldwide/ (cit. on p. 10).

[9] *Ecommerce Statistics 2020 [updated monthly]*. Aug. 2020. URL: https://ecommerceguide.com/ecommerce-statistics/ (cit. on p. 10).

[10] M. van Eeuwen. *Mobile conversational commerce: messenger chatbots as the next interface between businesses and consumers*. Feb. 2017. URL: http://essay.utwente.nl/71706/ (cit. on p. 5).

[11] K. Eremenko et al. *Machine Learning A-Z (Python amp; R in Data Science Course)*. URL: https://www.udemy.com/course/machinelearning/ (cit. on pp. 13–16, 46).

[12] W. Fang et al. “DOG: A new background removal for object recognition from images”. In: *Neurocomputing* 361 (2019), pp. 85–91. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.05.095. URL: http://www.sciencedirect.com/science/article/pii/S0925231219308963 (cit. on pp. 18, 48).

[13] Farfetch. *MultiModal Conversational Agents for the online fashion marketplace*. URL: https://ifetch-chatbot.github.io/ (cit. on p. 2).

[14] M. Fischer and M. Lam. “From Books to Bots: Using Medical Literature to Create a Chat Bot”. In: June 2016, pp. 23–28. DOI: 10.1145/2933566.2933573 (cit. on p. 1).

[15] D. Griol, J. Carbó, and J. Molina. “An automatic dialog simulation technique to develop and evaluate interactive conversational agents”. In: *Applied Artificial Intelligence* 27 (Oct. 2013), pp. 759–780. DOI: 10.1080/08839514.2013.835230 (cit. on p. 5).

[16] G. Huang, Z. Liu, and K. Q. Weinberger. “Densely Connected Convolutional Networks”. In: *CoRR* abs/1608.06993 (2016). arXiv: 1608.06993. URL: http://arxiv.org/abs/1608.06993 (cit. on p. 18).

[17] P.-S. Huang et al. “Learning deep structured semantic models for web search using clickthrough data”. In: Oct. 2013, pp. 2333–2338. DOI: 10.1145/2505515.2505665 (cit. on p. 10).

[18] S. Jégou et al. “The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation”. In: *CoRR* abs/1611.09326 (2016). arXiv: 1611.09326. URL: http://arxiv.org/abs/1611.09326 (cit. on p. 19).

[19] H. Joshi. “Proposal of Chat Based Automated System for Online Shopping”. In: *American Journal of Neural Networks and Applications* 3 (Jan. 2017), p. 1. DOI: 10.11648/j.ajnna.20170301.11 (cit. on p. 1).

[20] C. .-C. J. Kuo. *Understanding Convolutional Neural Networks with A Mathematical Model*. 2016. arXiv: 1609.04112 [cs.CV] (cit. on p. 13).

[21] J. Lei et al. *Pix2Surf: Learning Parametric 3D Surface Models of Objects from Images*. 2020. arXiv: 2008.07760 [cs.CV] (cit. on pp. 11, 22, 28, 39, 40, 58, 62).

[22] N. Meinshausen. “Quantile Regression Forests”. In: *Journal of Machine Learning Research* 7.35 (2006), pp. 983–999. URL: http://jmlr.org/papers/v7/meinshausen06a.html (cit. on p. 10).

[23] P. b. S. O’Dea and A. 20. *Smartphone users 2020*. Aug. 2020. URL: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (cit. on p. 10).

[24] K. Oh et al. "A Chatbot for Psychiatric Counseling in Mental Healthcare Service Based on Emotional Dialogue Analysis and Sentence Generation". In: *2017 18th IEEE International Conference on Mobile Data Management (MDM)*. 2017, pp. 371–375. DOI: 10.1109/MDM.2017.64 (cit. on p. 1).

[25] OPHoperHPO. *Ophoperhpo/image-background-remove-tool: a tool for removing background from photos with neural networks*. URL: https://github.com/OPHoperHPO/image-background-remove-tool (cit. on p. 48).

[26] C. Pricilla, D. Lestari, and D. Dharma. "Designing Interaction for Chatbot-Based Conversational Commerce with User-Centered Design". In: Aug. 2018, pp. 244–249. DOI: 10.1109/ICAICTA.2018.8541320 (cit. on pp. 9, 11, 22, 30).

[27] S. Quarteroni and S. Manandhar. "A Chatbot-based Interactive Question Answering System". In: 2007 (cit. on p. 8).

[28] W. L. in Research-Based User Experience. *The User Experience of Chatbots*. URL: https://www.nngroup.com/articles/chatbots/ (cit. on p. 7).

[29] C. Rother, V. Kolmogorov, and A. Blake. ""GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts". In: *ACM SIGGRAPH 2004 Papers*. SIGGRAPH '04. Los Angeles, California: Association for Computing Machinery, 2004, pp. 309–314. ISBN: 9781450378239. DOI: 10.1145/1186562.1015720. URL: https://doi.org/10.1145/1186562.1015720 (cit. on pp. 16, 48).

[30] A. Saha, M. M. Khapra, and K. Sankaranarayanan. "Towards Building Large Scale Multimodal Domain-Aware Conversation Systems". In: (2017). eprint: arXiv:1704.00200 (cit. on p. 37).

[31] *samples/cpp/grabcut.cpp*. URL: https://docs.opencv.org/3.4/d8/d34/samples_2cpp_2grabcut_8cpp-example.html (cit. on p. 17).

[32] J. Seering et al. "Beyond Dyadic Interactions: Considering Chatbots as Community Members". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–13. ISBN: 9781450359702. DOI: 10.1145/3290605.3300680. URL: https://doi.org/10.1145/3290605.3300680 (cit. on p. 1).

[33] A. Siddique, F. Jamour, and V. Hristidis. "Linguistically-Enriched and Context-AwareZero-Shot Slot Filling". In: *Proceedings of the Web Conference 2021*. WWW '21. Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 3279–3290. ISBN: 9781450383127. DOI: 10.1145/3442381.3449870. URL: https://doi.org/10.1145/3442381.3449870 (cit. on p. 49).

[34] W. Surakarin and P. Chongstitvatana. "Predicting types of clothing using SURF and LDP based on Bag of Features". In: *2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. 2015, pp. 1–5. DOI: 10.1109/ECTICon.2015.7207101 (cit. on p. 16).

[35] S. Vaddadi et al. *Developing Chatbot Wrapper for Online Shopping: A Case Study of Using Generic Mobile Messaging System*. Apr. 2020. DOI: 10.36227/techrxiv.12061602 (cit. on pp. 8, 9, 22).

[36] V. Verma. *10 usability Heuristics to Design Better Chatbots*. Nov. 2019. URL: https://uxdesign.cc/10-usability-heuristics-to-design-better-chatbots-654223552533 (cit. on p. 5).

[37] *What are Norman's design principles?* URL: https://www.educative.io/edpresso/what-are-normans-design-principles (cit. on p. 6).

[38] Q. Zhi and R. Metoyer. "GameBot: A Visualization-Augmented Chatbot for Sports Game". In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI EA '20. Honolulu, HI, USA: Association for Computing Machinery, 2020, pp. 1–7. ISBN: 9781450368193. DOI: 10.1145/3334480.3382794. URL: https://doi.org/10.1145/3334480.3382794 (cit. on p. 1).

A
Questionnaire Results of Preliminary Tests

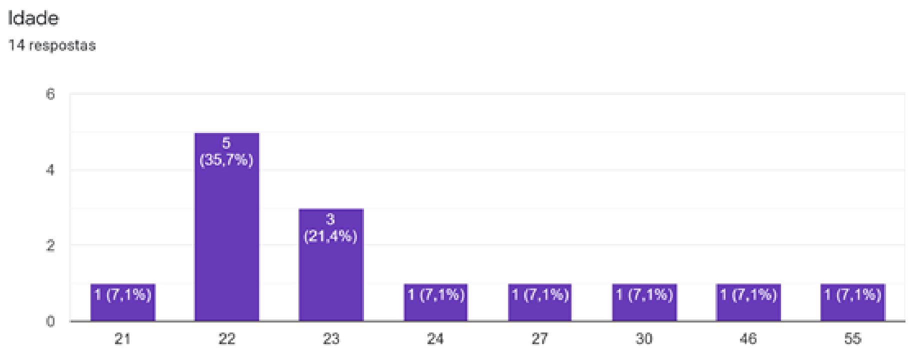


Figure A.1: Age

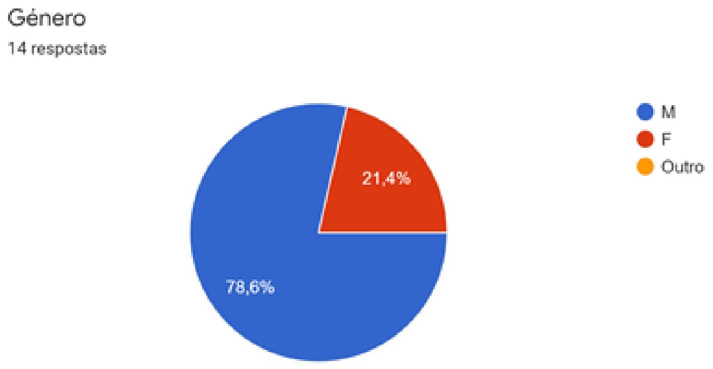


Figure A.2: Gender

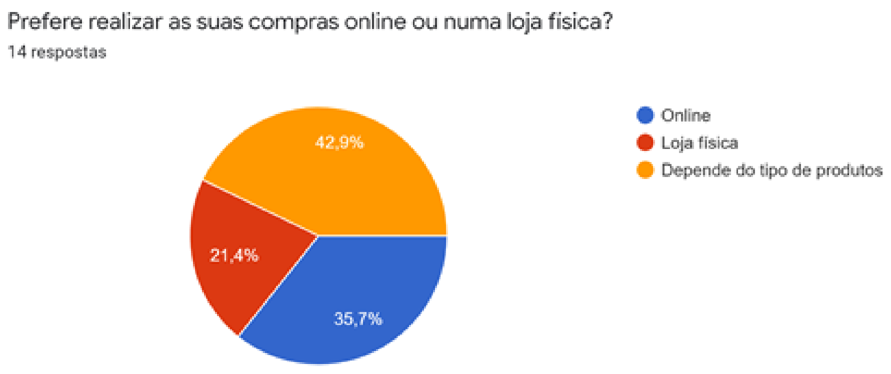


Figure A.3: Do you prefer to make your purchases online or in a physical store?

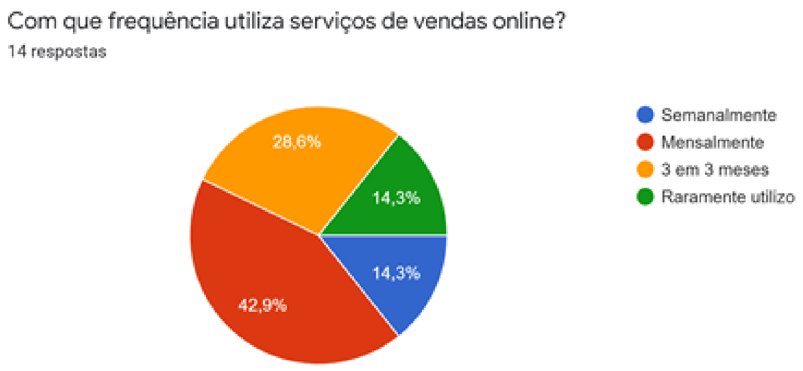


Figure A.4: How often do you use online sales services?

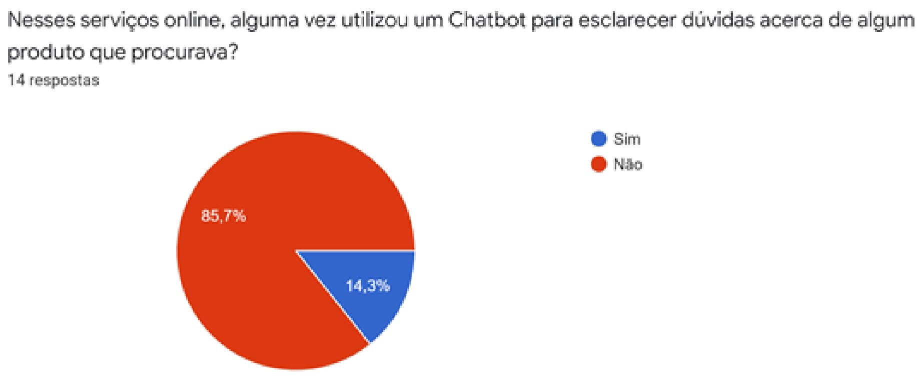


Figure A.5: In these online services, have you ever used a Chatbot to clarify doubts about a product you were looking for?

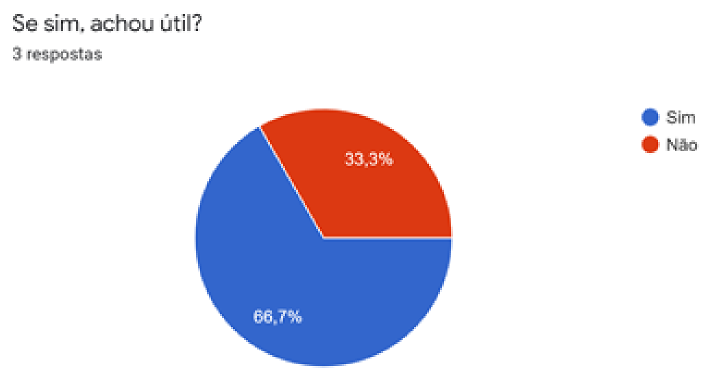


Figure A.6: If so, was it helpful?

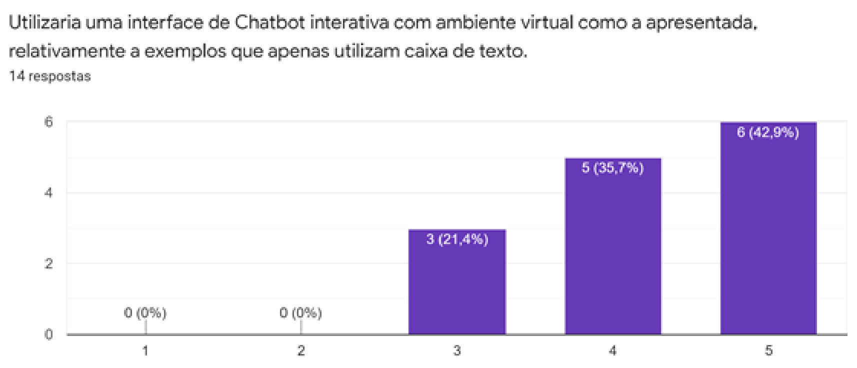


Figure A.7: I would use an interactive Chatbot interface with a virtual environment like the one shown, regarding examples that only use textbox chatting

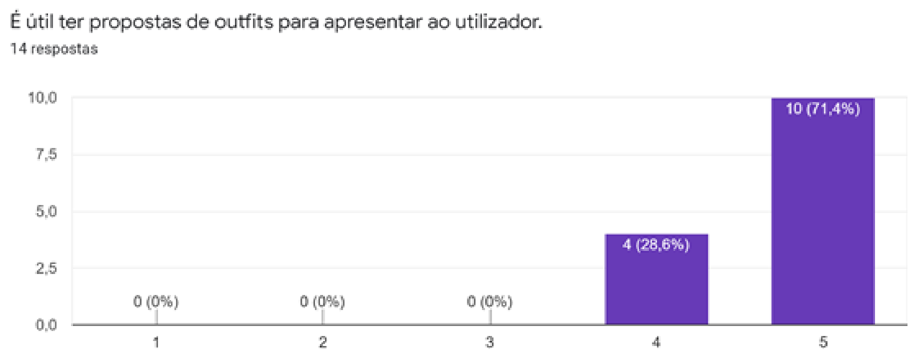


Figure A.8: It is useful presenting outfits proposals to the user.

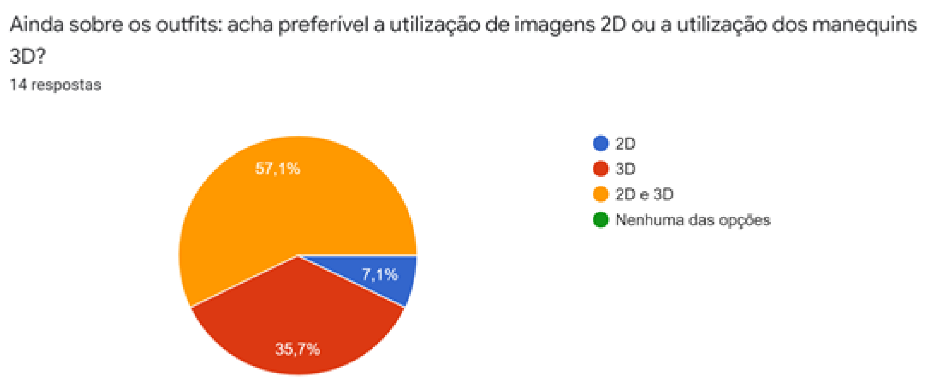


Figure A.9: Do you think it is preferable to use 2D images or use 3D mannequins?

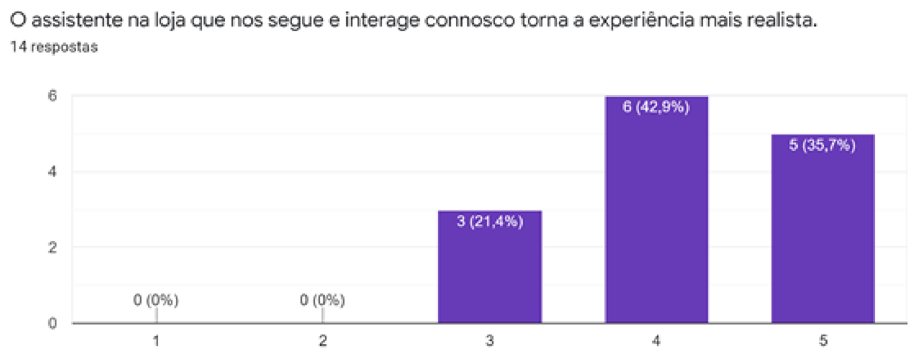


Figure A.10: The shop assistant who follows us and interacts with us makes the experience more realistic.

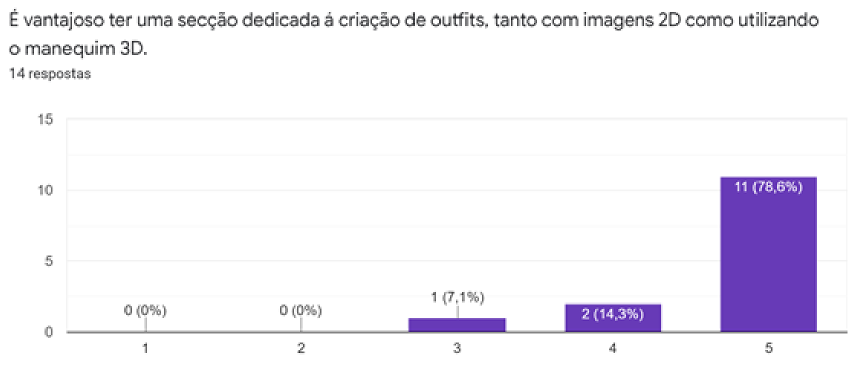


Figure A.11: It is advantageous to have a section dedicated to creating outfits, both with 2D images and using the 3D mannequin.

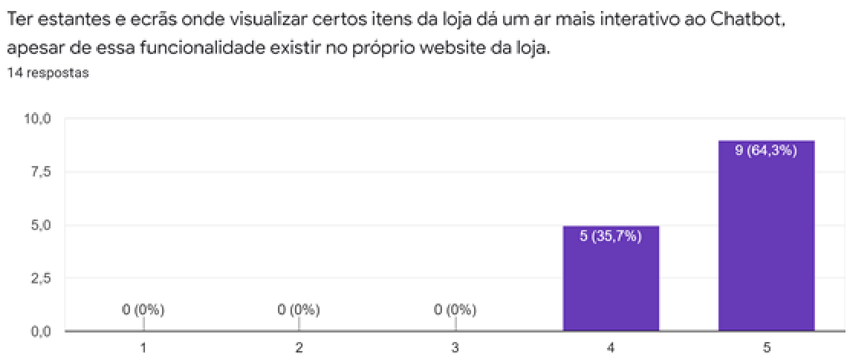


Figure A.12: Having shelves and screens where you can view certain store items gives a more interactive feel to Chatbot, although this functionality exists on the store's own website.

B

Questionnaire Results of Final Tests

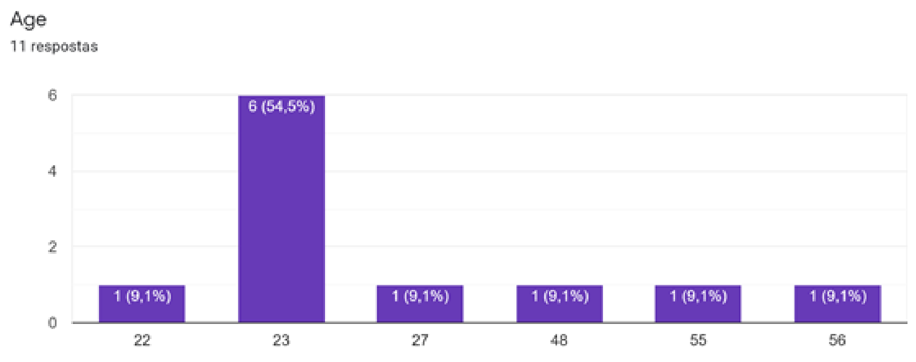


Figure B.1: Age

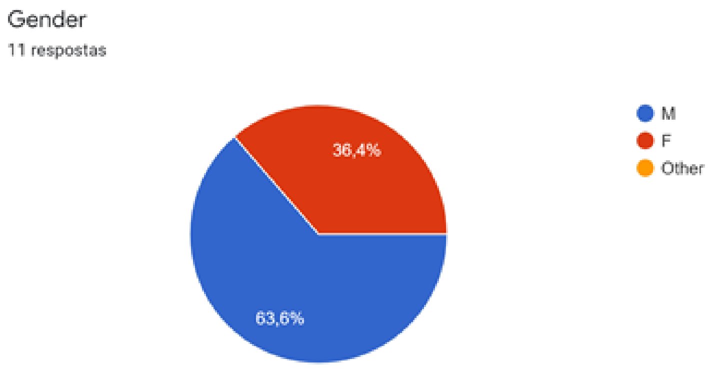


Figure B.2: Gender

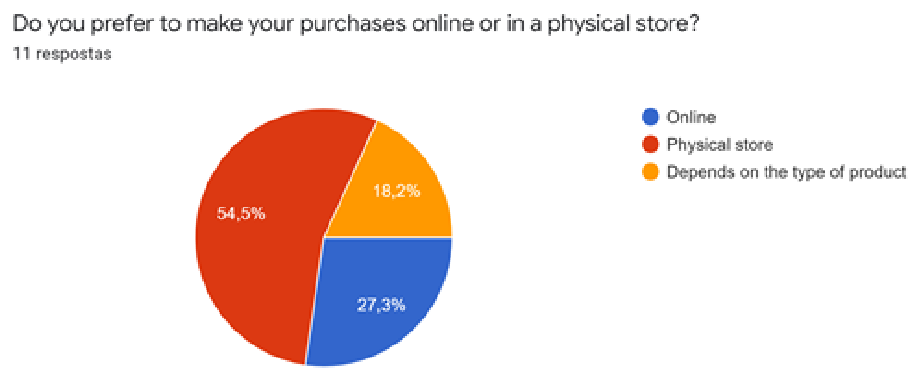


Figure B.3: Do you prefer to make your purchases online or in a physical store?

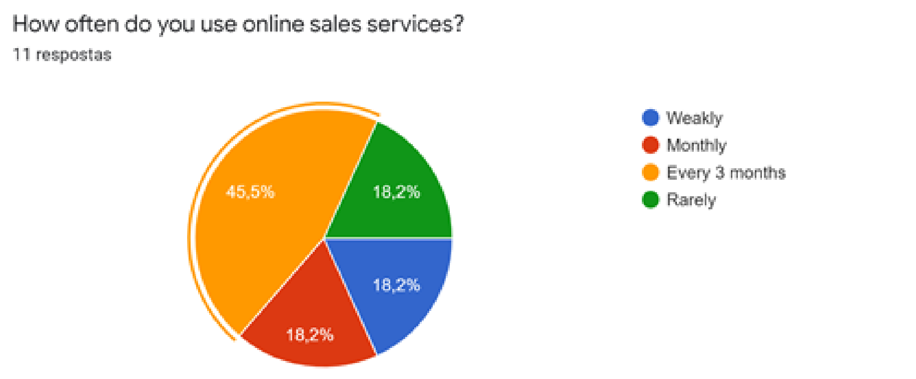


Figure B.4: How often do you use online sales services?

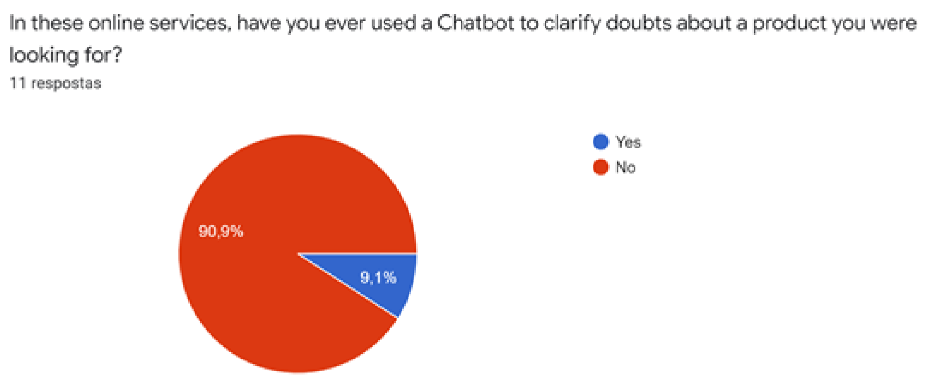


Figure B.5: In these online services, have you ever used a Chatbot to clarify doubts about a product you were looking for?

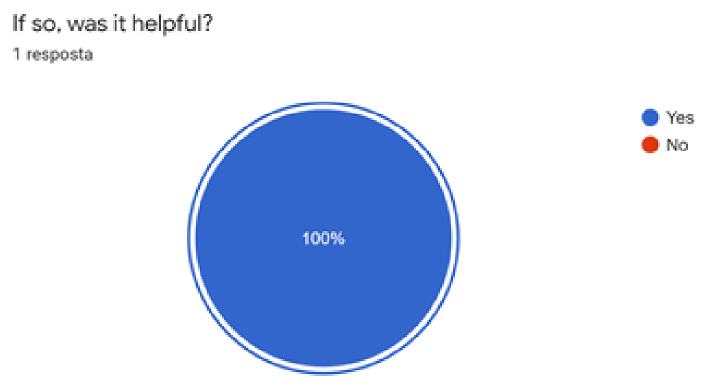


Figure B.6: If so, was it helpful?

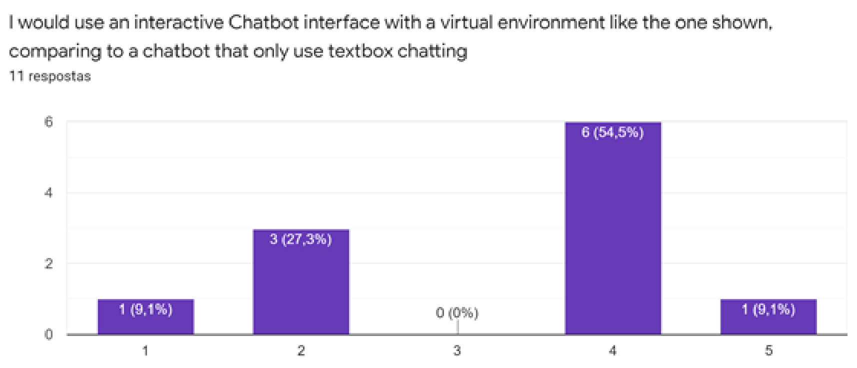


Figure B.7: I would use an interactive Chatbot interface with a virtual environment like the one shown, comparing to a chatbot that only use textbox chatting

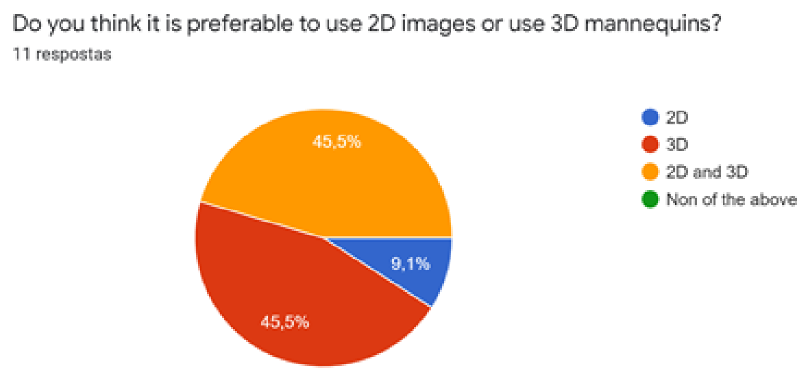


Figure B.8: Do you think it is preferable to use 2D images or use 3D mannequins?

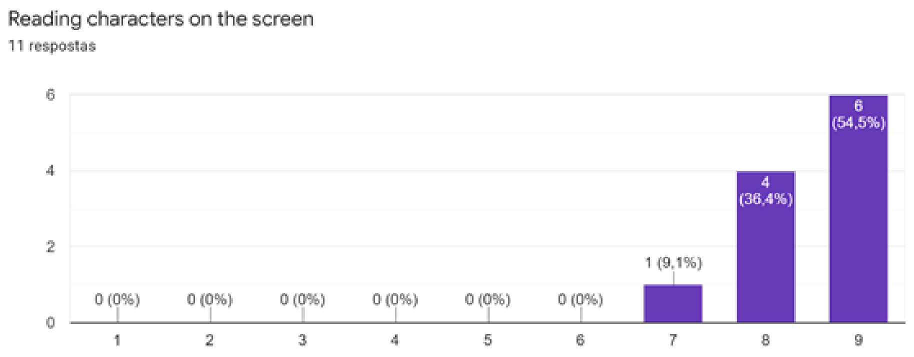


Figure B.9: Reading characters on the screen

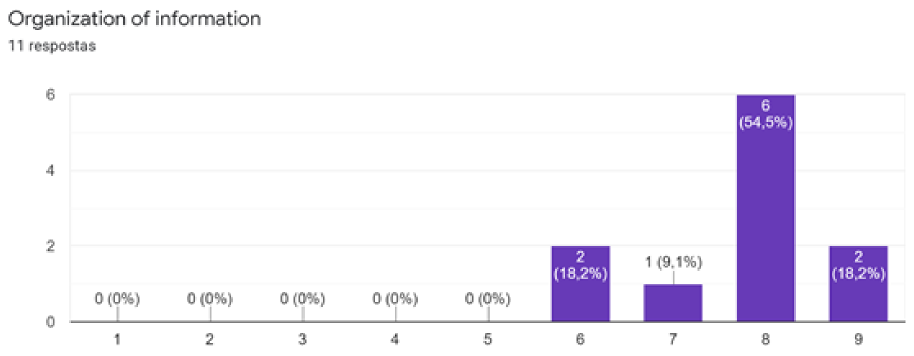


Figure B.10: Organization of information

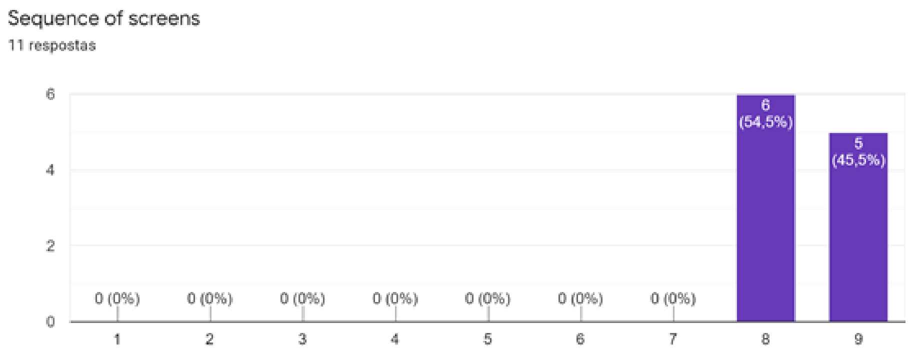


Figure B.11: Sequence of screens

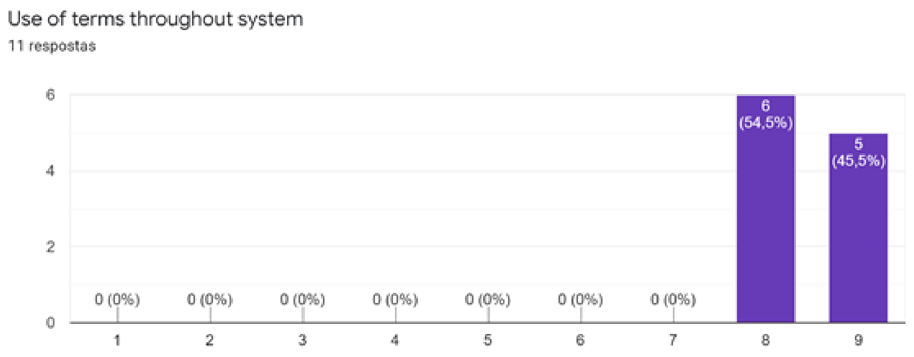


Figure B.12: Use of terms throughout system

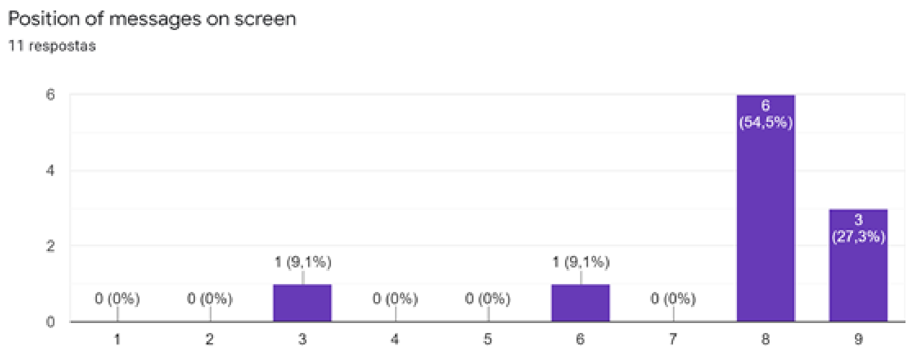


Figure B.13: Position of messages on screen

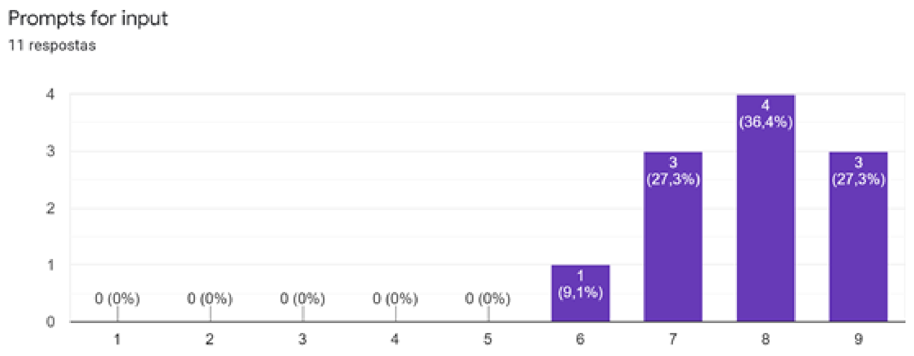


Figure B.14: Prompts for input

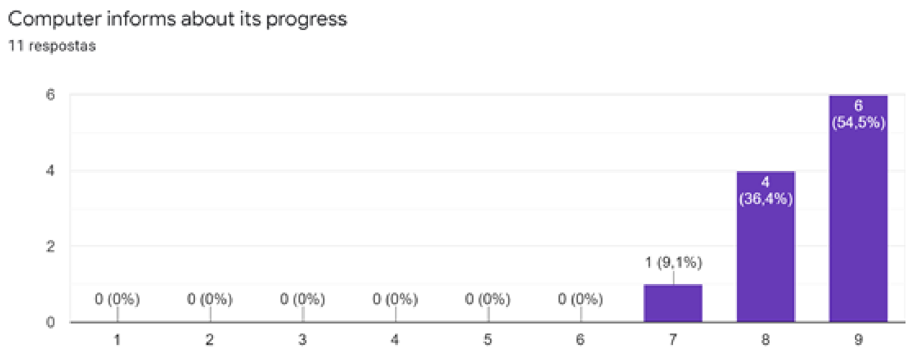


Figure B.15: Computer informs about its progress

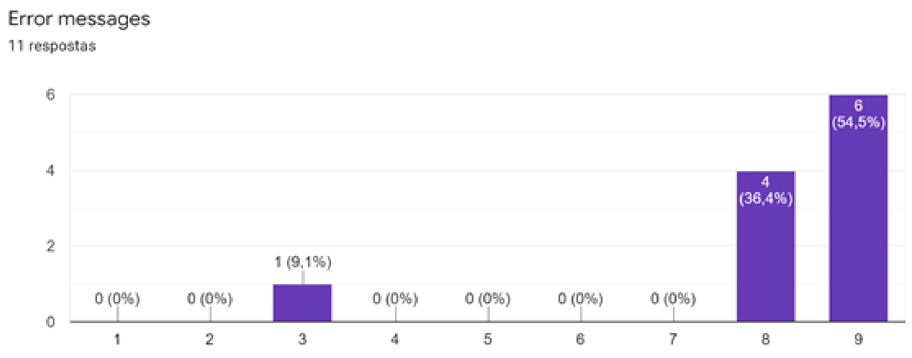


Figure B.16: Error messages

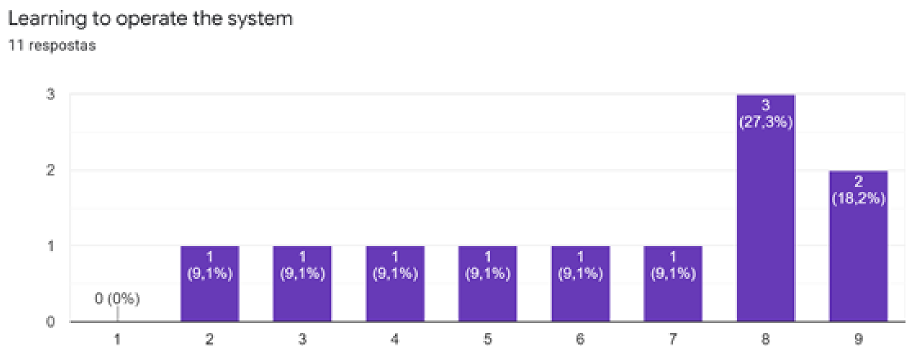


Figure B.17: Learning to operate the system

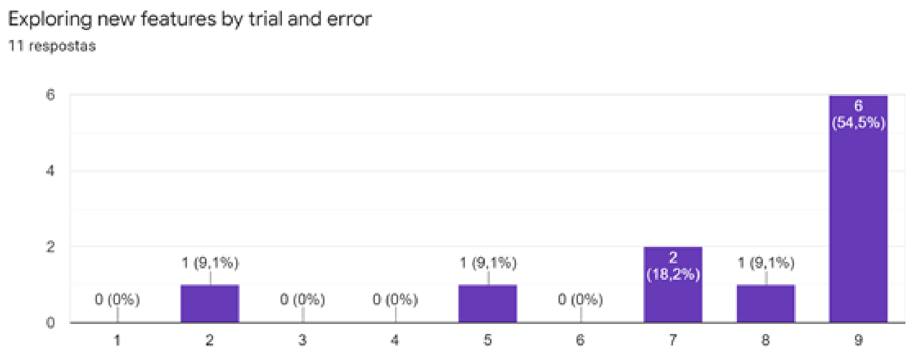


Figure B.18: Exploring new features by trial and error

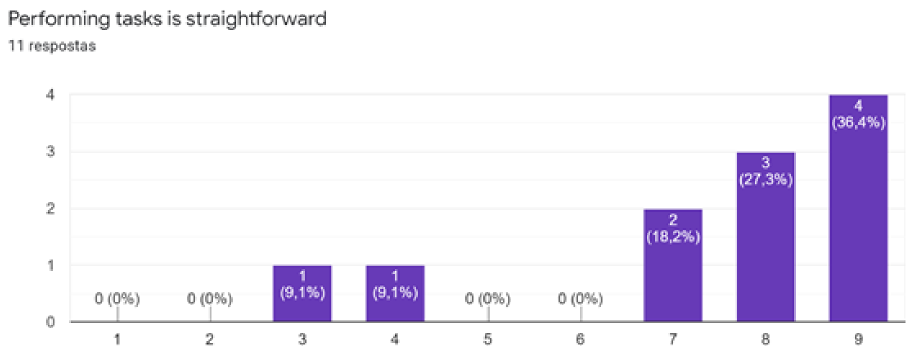


Figure B.19: Performing tasks is straightforward

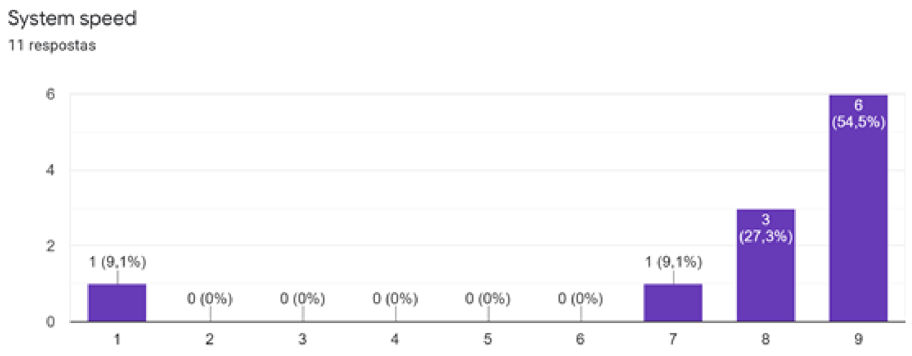


Figure B.20: System speed

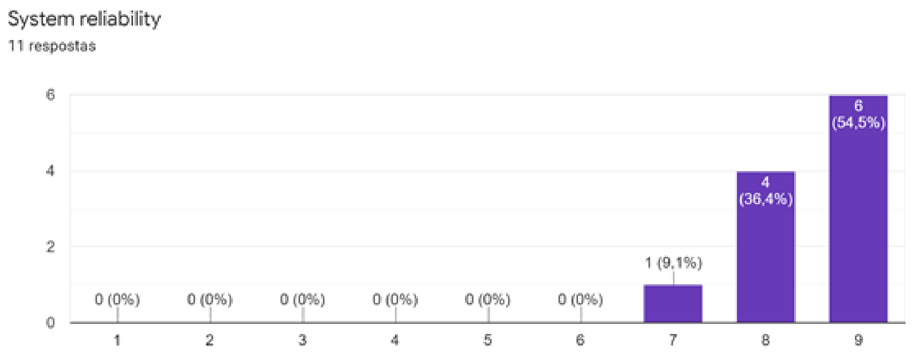


Figure B.21: System reliability

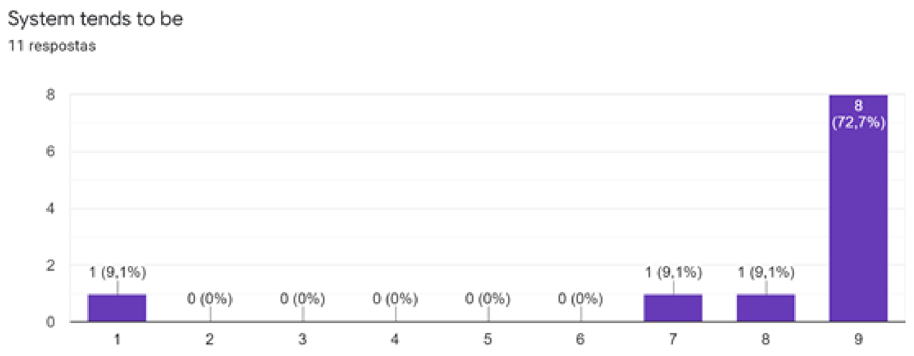


Figure B.22: System tends to be quiet/noisy

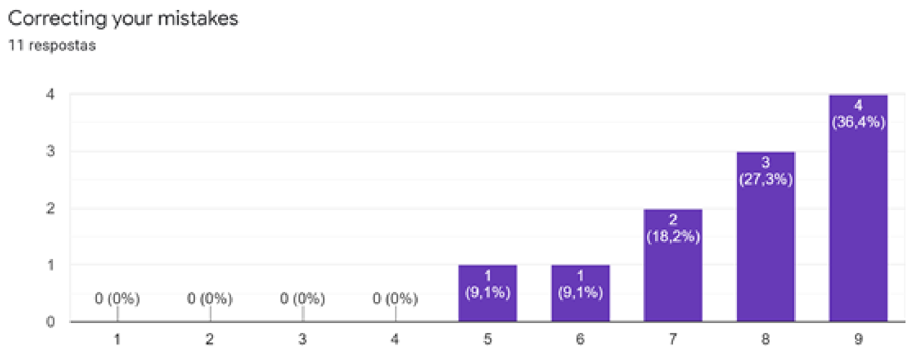


Figure B.23: Correcting your mistakes

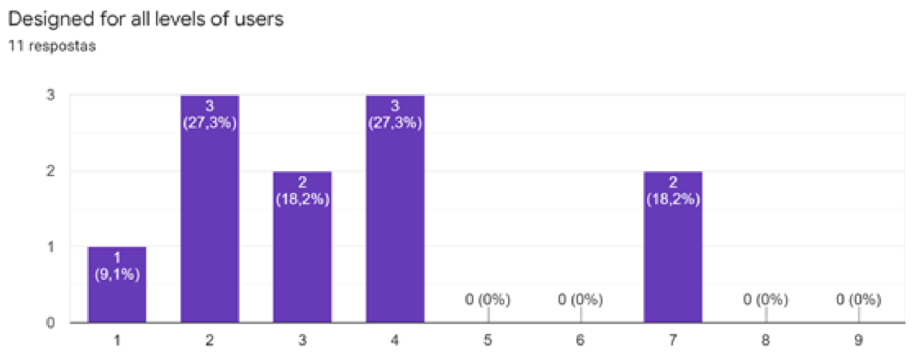


Figure B.24: Designed for all levels of users

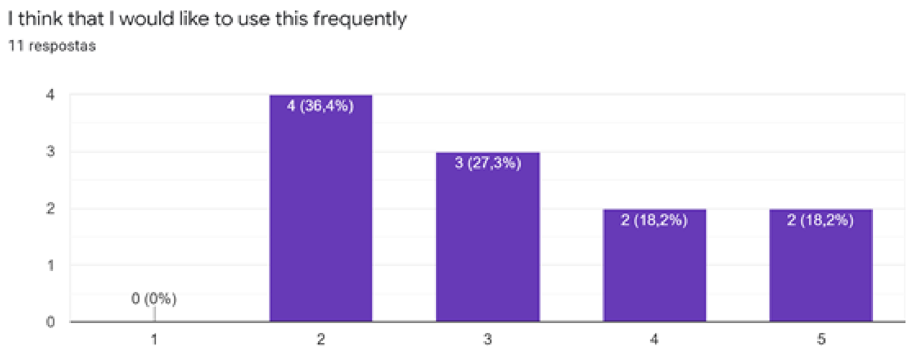


Figure B.25: I think that I would like to use this frequently

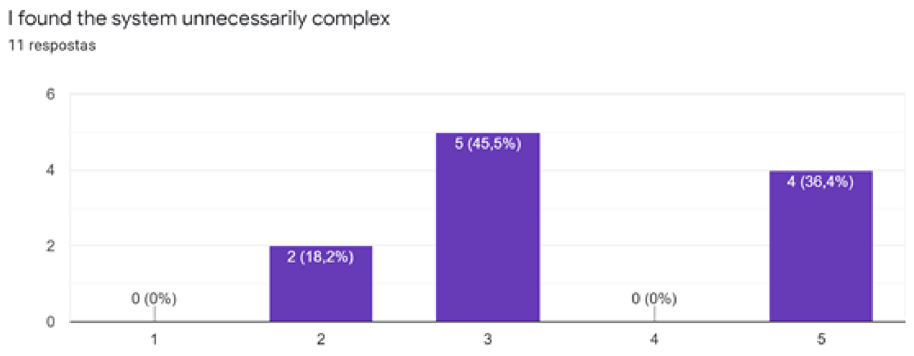


Figure B.26: I found the system unnecessarily complex

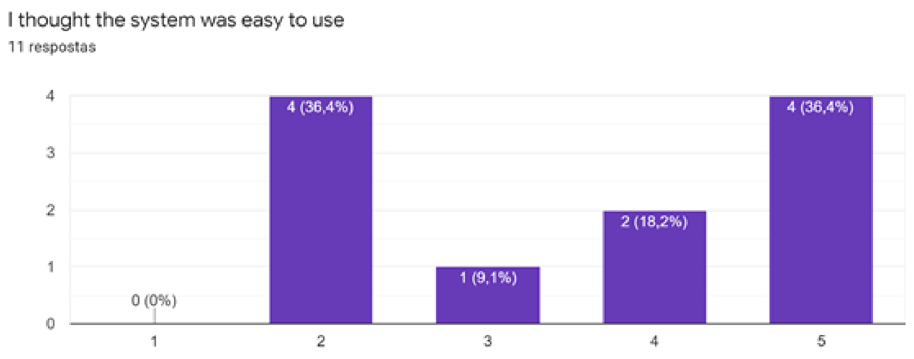


Figure B.27: I thought the system was easy to use

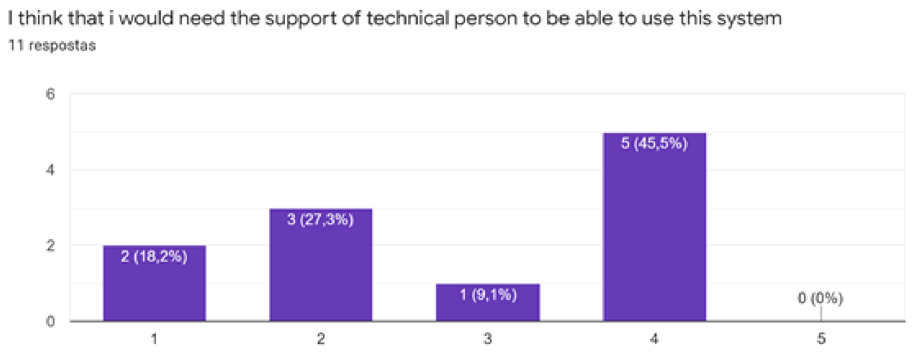


Figure B.28: I think that i would need the support of technical person to be able to use this system

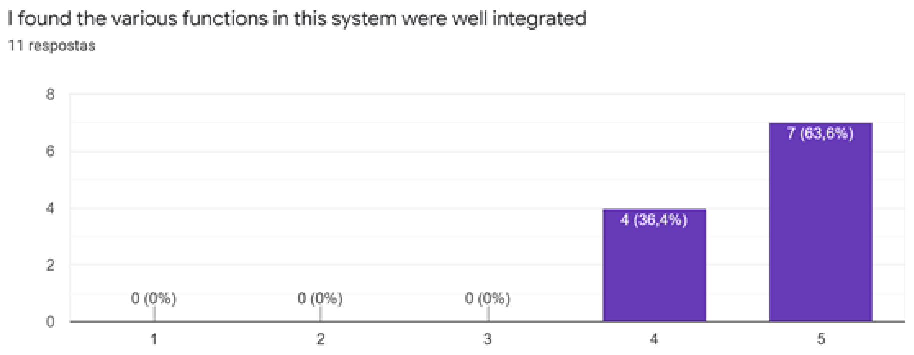


Figure B.29: I found the various functions in this system were well integrated

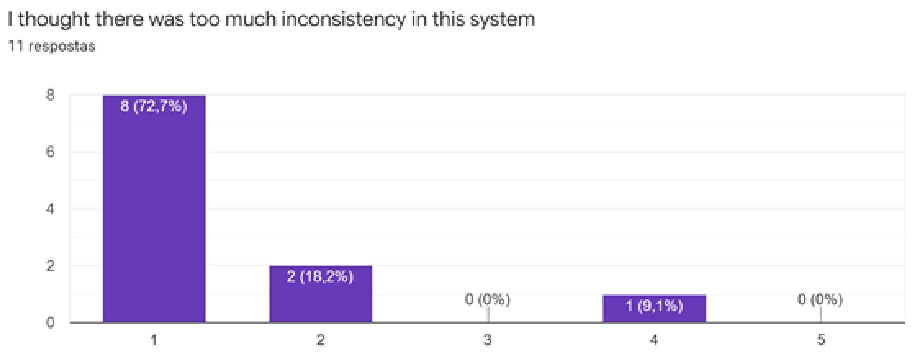


Figure B.30: I thought there was too much inconsistency in this system

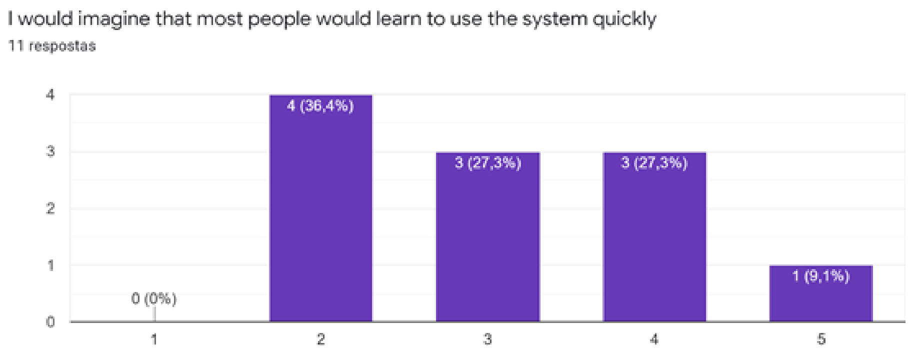


Figure B.31: I would imagine that most people would learn to use the system quickly

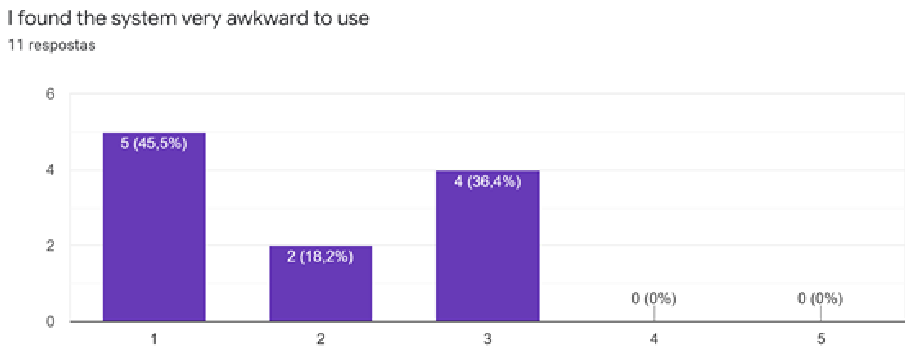


Figure B.32: I found the system very awkward to use

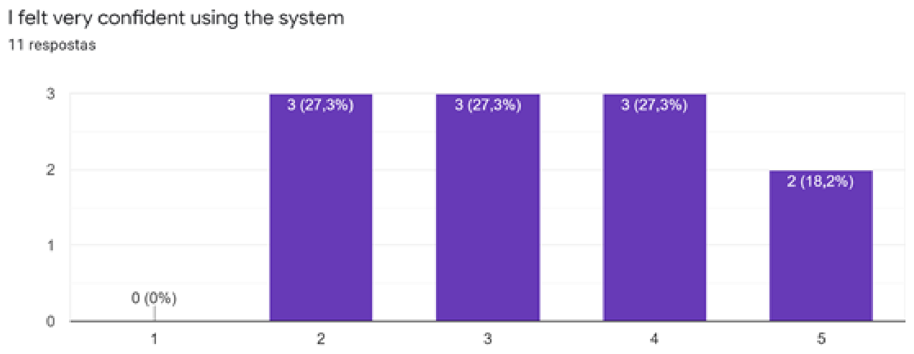


Figure B.33: I felt very confident using the system

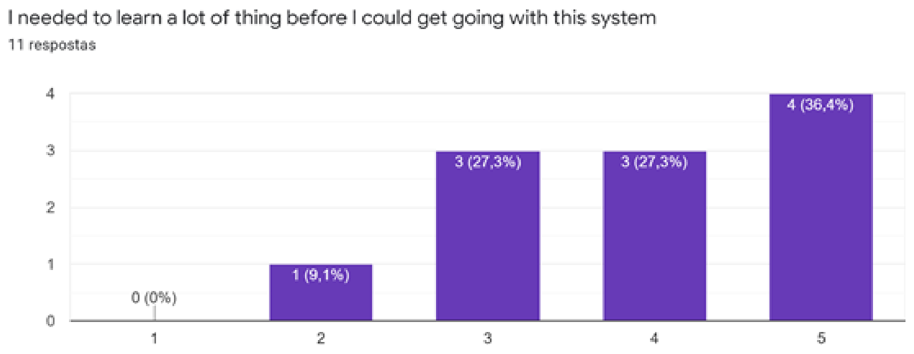


Figure B.34: I needed to learn a lot of thing before I could get going with this system