A personal fashion adviser application with a social component

Margarida Filipa Leal Correia

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur
Prof. Sandra Pereira Gama

Examination Committee

Chairperson: Prof. João Madeiras Pereira
Supervisor: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur
Members of the Committee: Prof. Hugo Nicolau
Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur

October 2017
Abstract

Fashion is a domain that has a big role in the economy and culture in the world. Recommender systems are a well-known field that aims to give personalized suggestions depending on the user and items. The search for them has increased over the years.

In this dissertation, we created an application called InsideFashion that provides a fashion recommender system with a combination algorithm to be able to retrieve combination of clothes. Our application has a social network integrated in it. In our recommender system, the user can receive suggestions of clothing items depending on the temperature and weather of a selected location, occasion and emotional status and combines the items regarding the style previously selected by the user. For that matter, the person needs to have their virtual closet available which could be done in our application by adding the items providing their photo and their description. As specified before, we also had a social network in which the users can follow other users in order to get inspiration from them.

We developed our application incrementally and iteratively performing a user study in order to connect the emotional status with clothing pieces. We also performed an heuristic evaluation in order to evaluate our application violations and develop a solution for those errors, and usability tests that allowed us to validate our application with a real user environment. To evaluate and validate our style combination algorithm, we performed a study in which we provided a set of outfits for the user to be able to distinguish between the available styles in which we conclude that 67.16 percent of the users were able to distinguish.

Keywords

Recommender systems, Fashion, Collaborative filtering systems, Content-based systems, Knowledge-based systems, Hybrid systems
Resumo

A moda é um domínio que tem um papel forte na economia e na cultura do mundo. Os sistemas de recomendação são um campo bem conhecido que visa fornecer sistemas capazes de dar sugestões personalizadas dependendo do utilizador e dos itens. A procura pelos sistemas de recomendação aumentou ao longo dos anos.

Nesta dissertação, criámos uma aplicação chamada InsideFashion que fornece um sistema de recomendação de moda com um algoritmo de combinação para poder combinar peças de roupas. Esta aplicação possui uma rede social integrada. No nosso sistema de recomendação, o utilizador recebe sugestões de peças de roupa de acordo com a temperatura e clima do local que o utilizador escolheu, ocasião e estado emocional além de que as peças ainda são combinadas de acordo com o estilo previamente selecionado. Para isso, a pessoa precisa de ter um armário virtual disponível, o que pode ser feito na nossa aplicação adicionando os itens fornecendo a sua foto e sua descrição. Como especificado anteriormente, nós também temos uma rede social na qual os utilizadores podem acompanhar outros utilizadores para se inspirarem neles.

Desenvolvemos a nossa aplicação de forma incremental e iterativa, realizando um estudo de utilizador com vista a relacionar o estado emocional com a escolha de peças de roupa. Também realizámos uma avaliação heurística para avaliar as nossas violações de heurísticas e desenvolver uma solução para os erros registados. Também foram feitos testes de usabilidade que nos permitiram validar a nossa aplicação com um ambiente real de utilizadores. Para avaliar e validar o nosso algoritmo de combinação de estilo, realizámos um estudo no qual fornecemos um conjunto de roupas para que o utilizador pudesse distinguir os estilos disponíveis e podemos concluir que 67,16 por cento dos utilizadores
conseguiram distinguir.

**Palavras Chave**

Sistemas de recomendação, Moda, Sistemas híbridos, Sistemas de base de conhecimento
## Contents

1 **Introduction**  
  1.1 Through the recommender system field and fashion domain  
  1.2 Motivation and objectives  
  1.3 Document structure  

2 **Background: recommendation system techniques**  
  2.1 Concepts  
  2.2 Techniques  
  2.2.1 Collaborative filtering  
  2.2.1.A Memory based  
  2.2.1.B Model based  
  2.2.2 Content-based  
  2.2.3 Knowledge based techniques  
  2.2.3.A Constraint-based  
  2.2.3.B Case-based  
  2.2.4 Hybrid  
  2.2.5 Discussion of recommendation techniques  
  2.3 Evaluation of recommender systems  
  2.3.1 Approaches  

3 **Background: Fashion systems**  

4 **Inside Fashion system**  
  4.1 System description  
  4.2 System architecture  
  4.3 Creation and storing user’s wardrobe  
  4.4 Recommendation task  
  4.4.1 First phase
# Table of Contents

4.4.2 Second phase .................................................. 45  
4.5 Social Network .................................................... 55  
4.6 Organisation ....................................................... 58  
4.7 Voice .............................................................. 60  
4.8 Interface .......................................................... 63  
5 Evaluation ........................................................... 67  
  5.1 Heuristic evaluation ........................................... 69  
  5.2 Usability tests .................................................. 70  
    5.2.1 Results ....................................................... 74  
      5.2.1.A Usability tests ....................................... 74  
      5.2.1.B SUS ..................................................... 76  
6 Conclusions and future work .................................... 78  
A Appendix .......................................................... 86  
  A.1 User Manual .................................................... 87  
    A.1.1 Closet ....................................................... 88  
    A.1.2 Recommendation ......................................... 91  
    A.1.3 Social Network ........................................... 92  
  A.2 Low fidelity prototypes ...................................... 93  
  A.3 Emotional status questionnaire ............................ 93
List of Figures

1.1 Comic of women and men on their daily life [6] ........................................... 4
2.1 Architecture of a content-based recommender system. Inspired on a graphic of [29] . . . . 12
2.2 Backtracking algorithm ............................................................ 15
2.3 Architecture of a case-based recommender. Adapted from a graphic of [35] ............... 16
3.1 Mirror appliance system ............................................................. 25
3.3 Fashion Coordinates Recommender System. Image from [16] ............................... 26
3.4 Getting the Look System. Image from [19] ........................................... 27
3.5 Hi, Magic Closet System. Image from [22] ............................................ 28
3.6 What am I gonna wear System. Image from [34] ........................................... 30
3.7 What am I gonna wear System. Image from [34] ........................................... 30
3.8 Personalised Clothing-Recommendation System. Image from [38] ....................... 31
4.1 Flow of information ................................................................. 38
4.2 InsideFashion architecture ........................................................... 38
4.3 System logic: algorithms ............................................................. 39
4.4 Add a piece of clothing in InsideFashion ............................................. 40
4.5 Annotation phase ................................................................. 41
4.6 Color of clothing piece ............................................................... 42
4.7 Recommendation task .............................................................. 43
4.8 Weather and temperature .......................................................... 44
4.9 Occasion ................................................................. 44
4.10 Emotional status ................................................................. 45
4.11 Example of classic style outfit ..................................................... 46
4.12 Example of casual style outfit ..................................................... 47
4.13 Example of modern style outfit .................................................... 48
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.49 Navigation bar</td>
<td>65</td>
</tr>
<tr>
<td>4.50 Add piece screen evolution</td>
<td>65</td>
</tr>
<tr>
<td>4.51 Recommender screen evolution</td>
<td>66</td>
</tr>
<tr>
<td>5.1 Sign-up errors</td>
<td>71</td>
</tr>
<tr>
<td>5.2 Color of clothing piece</td>
<td>72</td>
</tr>
<tr>
<td>5.3 Application SUS result</td>
<td>77</td>
</tr>
<tr>
<td>6.1 First view</td>
<td>81</td>
</tr>
<tr>
<td>6.2 Second view</td>
<td>81</td>
</tr>
<tr>
<td>A.1 Login page</td>
<td>87</td>
</tr>
<tr>
<td>A.2 Sign up page</td>
<td>87</td>
</tr>
<tr>
<td>A.3 Recover password page</td>
<td>88</td>
</tr>
<tr>
<td>A.4 Add item page</td>
<td>88</td>
</tr>
<tr>
<td>A.5 Add item color page</td>
<td>89</td>
</tr>
<tr>
<td>A.6 View item page</td>
<td>89</td>
</tr>
<tr>
<td>A.7 View item page</td>
<td>90</td>
</tr>
<tr>
<td>A.8 Create album page</td>
<td>90</td>
</tr>
<tr>
<td>A.9 Recommendation page</td>
<td>91</td>
</tr>
<tr>
<td>A.10 Social network</td>
<td>92</td>
</tr>
<tr>
<td>A.11 Feed page</td>
<td>93</td>
</tr>
<tr>
<td>A.12 Tagger system low fidelity prototype</td>
<td>93</td>
</tr>
<tr>
<td>A.13 Application low fidelity prototype</td>
<td>94</td>
</tr>
<tr>
<td>A.14 Application storyboard</td>
<td>94</td>
</tr>
<tr>
<td>A.15 InfoVis module low fidelity prototype</td>
<td>95</td>
</tr>
<tr>
<td>A.16 Explanation of the purpose of the study</td>
<td>95</td>
</tr>
<tr>
<td>A.17 Basic information needed</td>
<td>95</td>
</tr>
<tr>
<td>A.18 First question</td>
<td>96</td>
</tr>
<tr>
<td>A.19 Second question</td>
<td>96</td>
</tr>
<tr>
<td>A.20 Third question</td>
<td>96</td>
</tr>
<tr>
<td>A.21 Fourth question with a pop-up images exemplifying the selected style, in this case: classic</td>
<td>97</td>
</tr>
<tr>
<td>A.22 Fifth question</td>
<td>97</td>
</tr>
<tr>
<td>A.23 Sixth question</td>
<td>97</td>
</tr>
<tr>
<td>A.24 Seventh question</td>
<td>98</td>
</tr>
<tr>
<td>A.25 Eighth question with tooltip images exemplifying the selected pattern, in this case floral.</td>
<td>98</td>
</tr>
<tr>
<td>A.26 Ninth question</td>
<td>98</td>
</tr>
</tbody>
</table>
List of Tables

2.1 User-item or rating matrix ................................................................. 8
2.2 Discussion of the techniques .............................................................. 18
3.1 Describing systems regarding its features ............................................. 34
5.1 Frequency given a specific severity ..................................................... 69
5.2 Heuristic evaluation by severity and frequency ...................................... 70
5.3 Duration usability tests results ............................................................ 74
5.4 Number of errors on usability tests ..................................................... 75
5.5 SUS results ....................................................................................... 76
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS</td>
<td>Recommendation system</td>
</tr>
<tr>
<td>CBR</td>
<td>Case based reasoning</td>
</tr>
<tr>
<td>CF</td>
<td>Collaborative filtering</td>
</tr>
<tr>
<td>CB</td>
<td>Content based</td>
</tr>
</tbody>
</table>


## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Through the recommender system field and fashion domain</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Motivation and objectives</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>Document structure</td>
<td>6</td>
</tr>
</tbody>
</table>
1.1 Through the recommender system field and fashion domain

As time goes by, there is more and more information and content available on the web about any product, making it hard for the user to choose between all the available possibilities. It is impossible to see all the possible options and to choose one according to the person’s needs. The same problem arises in many contexts: bookstores – choosing a book to read –, streaming media – choosing a television (TV) show to watch–, clothing stores – choosing a piece of clothing for that special occasion –, the supermarket – the ingredients for a certain recipe. Recommender systems were created in the mid 1990’s and they basically filter available information. A recommender system is able to provide adequate suggestions for a certain user based on his preferences and tastes within a large amount of information.

Even though there are different kinds of recommendation systems, the main goal is the same: give expert advise without requiring experts. Although the domain and knowledge can be the same, there are different types of techniques that can be applied.

With the growth of the Internet and to increase sales, e-commerce providers started implementing recommender system (RS). These systems also allowed providers to better understand their costumers’ behaviour. Providers have to be careful when integrating a RS because if it does not provide good suggestions, user satisfaction decreases and the user loses trust in the system and stops caring about the recommendations provided, which leads to decreasing sales. If the system provides good suggestions, the user will be more satisfied, will rely more on the system, providing ratings that help the RS, will buy more items and will be more committed to the platform increasing user loyalty to it. There are many examples of e-commerce platforms that use RS, such as Amazon that uses collaborative filtering to provide suggestions to its users and is able to scale to millions of users and items. We cannot say that recommendation systems are specific for a certain industry or domain because they are somehow relevant to all of the domains. In our project, we aim to implement and apply a recommender system to the fashion domain.

Fashion is an important industry not only from an economic perspective but also from a cultural one. Economically, it represents an important sector in the European Union (EU) where more than six million people work. This industry tends to grow faster than the EU economy in general as noticed in the 2010 crisis. Fashion is a market where art, innovation and creativity meet and produces high-valued products, it is a way to express ourselves. This is a 295.6 billion pounds market and it includes textiles, clothing and footwear industries. There are 172 755 textile and clothing industries employing more than 1.62 million people in Europe alone. [10] [9]
1.2 Motivation and objectives

Both men and women struggle with the process of choosing outfits on a daily basis, as shown in figure 1.1, making a recommender system a support tool for their daily life.

There are two types of people, as we can see in the comic 1.1: the ones that have a lot of items in their wardrobe and the ones that have few items and each group struggle with different challenges. The task of choosing an outfit or buy one item in a store that matches their clothing can be hard. To people who have many items it is also struggling because they are unable to filter all the available items. Since they have a large amount of items in their disposal, usually they are not able to create an outfit when they have a specific event to attend or when they want to achieve different styles because they feel overwhelmed. With our RS, we intend to allow people to store all their clothing without needing to remember all of them whilst the system provides suggestions regarding their needs.

People with less available items are not able to create a great amount of different outfits. That is why they usually wear the same outfits. Being unable to innovate, they do not buy different pieces – called key pieces – that, by themselves, are able to create different styles. This RS will also be a great help for this group of people that are not into innovation in what concerns their clothing. It will help them creating different outfits and also buy different items to be able to combine them with already existing items in order to transform their look and achieve different styles. We will also have the possibility to follow other users being able to get some inspiration.

Everybody has their good and bad days and we want to create a RS that is able to provide suggestions based on the person’s mood. It will not only consider the occasion, style, weather and temperature of the day but also his/her emotional status providing personalised suggestions. The RS must be able to adapt its suggestions accordingly to each person. In order to understand a style, it is not the individual features but the combination of them that provide a style. It is not easy to consider all of them and the
suggestions rely on the combinations of the features because they have a dependency relation.

The already existing systems have limitations. Most of them only base their suggestions on one or two variables, usually, considering the colour attribute. There is one that considers more characteristics, suggests outfits based on weather, temperature, occasion and user history. But, even this one, is not able to provide innovative suggestions because it recommends previous outfits available in the user’s history, if the previous conditions – weather, temperature and occasion – are the same.

The next chapters of this dissertation will cover: RS, creativity, personalisation and natural language.

**The main goal is to create a recommender system in the fashion area that suggests outfits and takes into account personalisation. We want to suggest specific outfits for each user and consider their unique tastes in the recommendation. We aim to achieve personalisation by taking into account the following aspects:**

- Weather and temperature of a certain location
- User profile
- User emotional status
- Style wanted by the user
- Occasion

Hence the main goal of this dissertation is to combine all these different aspects, to provide a personalised suggestion not only from online catalogues but also from the pieces the user owns in his/her wardrobe. With this in mind, one needs to address the following challenges/aspects:

- How are the clothes usually characterised and which attributes are usually taken into account?
- How will we characterise our clothes?
- What are the advantages of doing this characterisation manually and what if this could be done automatically?
- How will the system use the users’ clothes?
- How can the different variables and attributes be taken into account in the recommendation and combination algorithm?
- How can we achieve different styles?
- Which algorithm or combination of algorithms should be used to guarantee personalised suggestions?
Given the fact that our dissertation not only includes a RS but also a combination algorithm developed within a web application, after developing our system, we will need to perform different tests regarding those aspects in order to evaluate each one of them and verify if our system and easy to use.

1.3 Document structure

This dissertation is divided into 5 chapters. Chapter 2 is the first part of the related work in which we describe the main concepts, most used techniques and how to evaluate a RSs. It is in an overall of the RSs. Chapter 3 is the second part of the related work where we go deeper and discuss some of the already existing fashion systems. Chapter 4 is where we discuss our system and the different aspects that it takes into account. Chapter 5 presents the evaluation of our system in its different phases. Chapter 6 is the discussion of the future work and conclusion.
Background: recommendation system techniques

Contents

2.1 Concepts ................................................................. 8
2.2 Techniques .............................................................. 9
2.3 Evaluation of recommender systems ............................. 19
Before we go deeper into analysing fashion systems because we are developing a recommender system, there is a need to understand the main concepts involved in a recommendation system in order to be able to understand how they are related to the goals that a recommendation system has.

In addition to the main concepts, we will also discuss and overview the main techniques used when developing a recommendation system and its pros and cons in order to decide which one is the best option to our specific case.

We will also analyse how to evaluate a recommender systems in terms of which evaluations can be done and in what metrics to use.

2.1 Concepts

In a RS, we have two main subjects:

- **Users** to which suggestions are provided.
- **Items** that are the things that are suggested.

Each user should provide his/her preferences and the RS should be able to predict them. These preferences are expressed as ratings that can be unary – "has bought", for example –, binary – like or dislike –, or a scale – 0-10. These three concepts are connected and expressed in a user-item or rating matrix, such as showed in table 2.1.

<table>
<thead>
<tr>
<th>User/Item</th>
<th>Item 1</th>
<th>Item 2</th>
<th>Item 3</th>
<th>Item 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User A</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>?</td>
</tr>
<tr>
<td>User B</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User C</td>
<td>2</td>
<td>?</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

A RS has a **prediction function** that discovers the rating that a specific user would give to an item in case that he/she had not yet provided it (represented by "?" in table 2.1, unknown values). After applying this function, it would perform the **recommend task**, where it searches for the items that the user has higher probability of liking in order to retrieve the top ranked items of that list.

An information collection phase is needed to generate an appropriate user profile. A user profile can be obtained using feedback. There are different types of feedback and, when user-item matrices are used, the RS uses explicit feedback because the ratings are directly provided by the user. But a RS can also be able to infer more about the user behaviour by applying other techniques. [15]

- **Explicit feedback**: This feedback only uses the ratings provided by the users to evaluate and provide a recommendation. This feedback can also include text comments or like/dislikes (binary
The main problem with comments is that, even though they are helpful and the users can better express their opinions, the system has to interpret text and tackle with the inherent problem of subjectivity introduced by using natural language.

- **Pros**: since it does not infer any kind of information, the process is really transparent assuring that the recommendations given by the system are reliable and with high quality.

- **Cons**: the user needs to rate the items that he/she has purchased, which is time consuming, and he/she can not provide well-stated ratings. The fact that it only uses the ratings is really limiting because users can be interested in other types of items even though they do not express it.

**Implicit feedback**: it monitors the user's activity, having access to his/her purchase history, clicks, time spent on webpages, among others.

- **Pros**: since it infers the user's preferences, the user does not need to do anything. Sometimes we do not buy things because we find them useful but because they are trendy. When we search something online, we have some interest on it and this technique takes that into account. The fact that it includes an inferring process means that it may have less accuracy because it does not consider information directly given by the user.

- **Cons**: it does not consider user information, such as purchases and ratings. It is a process which entails more costs because there is a need to monitor the user activity.

**Hybrid feedback**: this technique combines the previous two techniques, implicit and explicit feedback, which means that it combines the benefits of both of them. For example, at the same time that it infers user's preferences, it allows them to rate products (if they want to) and takes that into consideration.

2.2 **Techniques**

There are several techniques that can be used in RSs and we will discuss and analyse the most important ones in this section.

2.2.1 **Collaborative filtering**

This technique is the most popular one and was first introduced by Gobler et al. in 1992, “Information filtering can be more effective when humans are involved in the filtering process” [14] and it was later refined, in 1994, by Resnick et al., who wrote that two users are like-minded when they rated items alike [32]. This technique suggests items that other similar users bought or liked in the past. To see
if two users have similar taste, a function that evaluates their similarity based on their rating history is applied. In this approach, we will find users’ neighbours and use their ratings to predict the ratings of the user under studying.

We will start by presenting the two main subtechniques that are the memory and the model based.

2.2.1.A Memory based

We will be analysing the sub-techniques of the memory based.

User to user

In user-to-user collaborative filtering, the user-item matrix is stored in the system and is used directly to predict ratings for items that are new or that the user did not rated before. This prediction can be done using the user’s neighbours that are usually the ones whose ratings for the same items are most correlated to the items rated by the user.

This technique assumes that the user’s rating is based on the ones given by like-minded people. The rating of a user $u$ for a new item $i$, $r_{ui}$, is evaluated using the ratings that were given by the users that are most similar to user $u$. The $n$ users that are most similar to the user $u$, given a specific function of similarity, and that have rated the item $i$, are the $k$-nearest-neighbours. We can apply different similarity measures to find these neighbours.

We can predict the rating $r_{ui}$ in two ways:

- treat each neighbour of a specific user equally, considering that each one of them has the same similarity with that user. Since the neighbours are treated as equals, it does the average rating of the ratings given to the item $i$ by those neighbours.

$$r_{ui} = \frac{1}{N_u} \sum_{u_1}^{u,N} r_{u_1}$$

(2.1)

- or weight each neighbour using higher weights to the neighbours that are most similar to the user $u$.

$$r_{ui} = \frac{\sum_{u_1}^{u,N} w_{u_1} r_{u_1}}{\sum_{u_1}^{u,N} |w_{u_1}|}$$

(2.2)

Item-to-item

In item-to-item collaborative filtering, instead of using the similarity between users, the user’s ratings of an item are predicted based on the ratings of the user $u$ for items similar to the specific item.
In this specific approach, two items are more similar if the ratings given to those items were similar too. Instead of applying the similarity measures introduced before to the users, it applies them to the items. A rating can be evaluated, for example, as the weighted average of the ratings given by user $u$ to the items rated by user $u$ most similar to item $i$.

Discussion between user-to-user and item-to-item techniques

When analysing both user-to-user and item-to-item techniques, we compare them in terms of their benefits. Choosing between one of them depends on the context of the problem. If the group of users is larger than the number of items, one should choose an item-based. Otherwise, we should choose the user-based. We should choose the technique where the dimension space is lower. In general, the item-to-item outperforms the user-to-user because a user is much more complex than an item and the similarity between items is more meaningful.

2.2.1. Model based

One of the main problems of memory-based techniques is the sparsity problem that occurs when there are many cells with no ratings available in the user-item matrix. This problem can be caused if the item is new, because there are no ratings or because people have not rated the item yet. This problem introduces errors in the recommendation process because it lacks information to give a consistent, valid and proper recommendation to the user.

To solve this problem, the model based technique uses and analyses the user-item matrix so it can establish relations between the items and produce a pre-computed model. If this model is pre-computed, it can improve the recommendation algorithm performance. These models can be computed by association rule missing, clustering or using a Naive Bayes approach.

2.2.2. Content-based

This is a technique that has a completely different approach from the collaborative technique as analysed in [29], [18] and [37]. Instead of giving suggestions based on users that are somehow alike, it recommends items that the user has more probability of liking based on the content of the items, and not directly in the ratings as in the collaborative. That is why the collaborative is a domain-independent technique and the content is domain-dependent because it depends on the content of the items.

One of the main goals in this technique is the analysis of item properties that the user liked in the past. Therefore, one needs to produce a user profile. This user profile is made to help the recommendation
Figure 2.1: Architecture of a content-based recommender system. Inspired on a graphic of [29]

process, where the system does a match between the properties of a given item and the ones of the user profile. A profile is a set of features, usually a vector, in which each entry represents a feature. The algorithm only recommends the items that are more similar to the ones that were higher rated or purchased by the user in the past.

Basically, as illustrated in figure 2.1, it has a content-analyser component that will study the item characteristics organising its descriptions. With the help of the profile learning component, a user profile will be built. This profile depends on the description of the items that the user liked in the past. Whenever it makes a recommendation, the filtering component will analyse the user profile built previously by the profile learner, to provide adequate suggestions accordingly to the specific user. If the user gives feedback, it will improve the future suggestions since his/her profile will be richer by having more information available.

We divided this in three sections. First, we present the two main phases, building a user profile and recommend an item. We conclude with a discussion.

**Building user profiles**

As we have mentioned before there is a need to, for each item, build an item profile and create an user profile that is based on these item profiles. Consider that a user liked a set of items $i_1, \ldots, i_n$ then:

- The simplest way to create a user profile is by averaging the item profiles.

$$User\ profile = \frac{i_1 + \ldots + i_n}{n}$$ (2.3)
• However, this formula simplifies the problem because it builds a profile that does not take into account that a user likes an item more than other. To take that into account, we can use a weight average of the item profiles, in which each item has associated a weight that depends on how much a user likes or dislikes the item, using its rating.

\[
User\ profile = \frac{w_1i_1, \ldots, w_ni_n}{n}
\]  

(2.4)

• Another possibility is to normalise the weights by using the average rating, in a scale, given by the user.

By doing that, we achieve negative weights for items that are underestimated, meaning that they are below the average of the user’s rating, and positive when they are above.

**Recommend items**

Now that a user profile has been built, we have to go through every item in the catalogue and recommend the ones that are more similar. For that, we can use, for example, the co-sine similarity because the user profile and item profile are both vectors.

\[
U(x,i) = \cos(\alpha) = \frac{(x,i)}{||x||i}
\]

(2.5)

where the \(\cos(\alpha)\) is the similarity between them, the higher the co-sine, the smaller \(\alpha\) is, the more similar the items are.

This approach can be used in different contexts but it is most used in documents to recommend, for example, articles.

**Discussion of content-based technique:**

This technique creates a user profile where the users’ characteristics are based on the items they liked, ignoring the opinion of other users. Because of this, it is able to recommend items that are new and unpopular. Because it is based on the user profile, if it changes, it is capable of adjusting the recommendations. However, being based only on the features of the items means that we need to have a rich description of those features, which may not always be possible if there is no metadata about the items available - limited content analysis. It can overspecialize in the same type of items as the ones specified in the users’ profiles, introducing no serendipity. But it can recommend items to users that have unique tastes because it is based on the user profile. In the beginning, until the user has rated or
bought anything, it suffers from the cold-start problem to build user profiles.

2.2.3 Knowledge based techniques

The main goal of this approach is meeting the user requirements with a system that has knowledge about the specific domain of the item. The domain experts are the ones who give the knowledge and the engineers are the ones that formalize that knowledge into a representation that is usually called the recommender knowledge base.

The knowledge based technique is, as seen in some papers [4] [36] [13] [27], divided into two similar sub-techniques because both need user requirements and return a solution that fulfill them and, if there is no solution, the user can change their specifications. The main difference between these two techniques, case-based and constraint-based, is the way they evaluate recommendations. While the constraint-based approach uses the rules that connect user specifications with item properties, the case-based uses similarity metrics.

Sub-techniques

As said before, there are two types of knowledge based techniques, which will be analysed in the following subsections.

2.2.3.A Constraint-based

In this technique, there are two main concepts:

(Vc, Vprod): Vc refers to the user requirements and Vprod refers to the item properties. These are the variables of our problem and they depend on the user and on the item.

(CC, CR, CF, Cprod): CC are the constraints that represent user requirements, CR restrict the possible options of user features, CF defines filter conditions that specify the relationship between the user and item properties and Cprod are the restrictions that describe the instances of an item, Vprod.

The problem takes into consideration a set of constraints that are specified in a certain language.

- Conjunctive database query: the query given as input is a conjunctive query and the system returns the items that fulfils the user requirements.
• **Ranking items:** The process finds items that fulfill the specified user requirements. Sometimes several items fulfill those requirements, so items need to be ranked.

In constraint-based, the task of recommending an item can be viewed as a constraint satisfaction problem

\[
(VC, VPROD, CR \cup CF \cup CC \cup CPROD)
\]

(2.6)

We can solve this problem using backtracking and constraint propagation.

• **Backtracking:** assigns all possible values to a variable and if all of them are not consistent, it backtracks to the parent node and change its value to try to have consistency as we can see in figure 2.2.

![Figure 2.2: Backtracking algorithm](image)

• **Constraint propagation:** the main problem of the algorithm specified before is that it revisits parts of the search space where there is no solution which reveals to be time consuming. To reduce the search space improving the efficiency of the algorithm, there are methods that can be applied that transforms the present problem into an equivalent one.

### 2.2.3.B Case-based

As shown in figure 2.3, this specific technique relies on past similar cases to solve a new problem. The system has in memory all previous cases and when a new problem arises it searches for similar ones so the solution can be reused in the new context. The system can reuse part of it or adapt those similar cases, depending on the differences that the cases have. The system stores the new case in the memory. We have to note that, all knowledge resides in the similarity function that selects that are similar with the present problem. Case based reasoning (CBR) uses the critiquing approach.

**Problems of knowledge-based techniques**

One of the problems of this technique \([2]\) is the fact that user requirements may not be well specified. If this is the case, the system may not find a suitable solution for those requirements giving an inappro-
Figure 2.3: Architecture of a case-based recommender. Adapted from a graphic of [35]

appropriate answer to the problem “recommendation not found”. To solve the problem of lack of solution, one can use query relaxation. If there are too many solutions, one can use query tightening.

• **Query relaxation**: in this specific algorithm, all the possible items are stored in a database table. The recommendation task starts with a conjunctive query that specifies the user requirements and the system is supposed to find a maximal sub-query of the original one so it can return at least one item as a recommendation. Although this was a technique that was first discovered in the context of the case-based, it can also be applied in the constraint-based. However, applied to the constraint-based, this query relaxation approach finds a maximal subset of the $CF$. These restrictions are directly related to the user so, if we retain more constraints, we can guarantee that the system will return more suitable items related to the user requirements.

• **Query Tightening**: Having too many items being recommended is another possible problem. When this situation arises, the user is asked to specify his/her requirements in a more precise way. However, to avoid this situation where the user has to waste more time, the system can show all of the options. The problem of this technique is displaying the options in a specific order that is related to the user requirements. The process starts with the user giving a specific query and this algorithm takes it as its input. Then, all the items covered by that query will be retrieved and the system asks the user to select three properties, so it can refine the query.

**Comparison and discussion of knowledge-based techniques**

This technique is really different from the collaborative and content-based techniques because it does not have to “understand” the user, as he/she specifies his/her own needs. Due to this, knowledge based techniques do not suffer from the cold start problem as the others did. This approach assumes that we have a base of knowledge in the specific domain which is hard to do giving the fact that they
need to convert that knowledge to formal representations.

2.2.4 Hybrid

All techniques have limitations. To overcome them, the hybrid approaches were created. The idea is to combine some of the previous techniques optimising the system and overcoming the limitations of one technique with the strengths of another one.

2.2.5 Discussion of recommendation techniques

To create a RS, as analysed in the table 5.5, we have to decide which technique we want to apply. As analysed in the last section of the related work, there are many techniques that we can apply. After weighing pros and cons and analysing which are the main goals of the RS, one can decide which technique to use.

Collaborative filtering is the most popular and used. The comparison between the collaborative and the content-based is usually made. One of the major differences is the fact that the Collaborative filtering (CF) does not depend on the domain – content of the items – to make its predictions whilst the Content-based (CB) does. Relying on the domain, as the CB does, represents a challenge because the RS will be able to provide suggestions if there is sufficient metadata about it – limited content analysis. However, if it has a rich description of the items, it is able to provide suggestions even of items that no user has yet gave feedback. This is something that the CF is not able to do, because it relies on the user’s neighbours ratings and, if there is no item’s ratings yet available, it cannot suggest it – data sparsity. Since CB bases their suggestions only on the content of the items that the user liked in the past, when his/her preferences change, it is able to adjust the suggestions in a short amount of time but it can overspecialise its suggestions – overspecialisation.

To perform good suggestions, CF has to analyse the user’s profile because the recommendations depend on it and on the user’s neighbours profile. If it does not yet have the user profile available, meaning that the user did not rate any item yet, it will not be able to provide recommendations – cold-start problem. But, if it has a user profile with lots of items rated, it will be able to provide serendipitous suggestions – serendipity. Depending only on the user’s and neighbours’ profile, it makes almost impossible for the system to distinguish between two items that are similar but have different names – synonymy.

The knowledge-based systems are different from the previous ones because their key to provide suggestions is the knowledge model in which it relies. Since it needs to formalize the problem with the help of domain experts and create a model about it, in the beginning, the system will be able to provide better suggestions comparing with the CF and CB. But, if the learning agents are not well implemented, the quality of recommendations can be worse. Of course that, if a user changes his/her preferences, the
### Table 2.2: Discussion of the techniques

<table>
<thead>
<tr>
<th>Techniques</th>
<th>Pros</th>
<th>How</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collaborative filtering</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User and community</td>
<td>User 1 ratings and set of item ratings</td>
<td>Predict user 1 ratings and use them to predict</td>
<td>New user cold-start problem.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new items.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Quality relies on the previous data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Content based</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User and item features</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No rater problem.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Popular bias.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>It does not require genre-related recommendations.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>New user cold-start problem.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Knowledge based</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User, item features and knowledge model.</td>
<td>Infer a relation between users' needs and items' features.</td>
<td>Adjust itself if there are changes in the users' needs.</td>
<td>New item cold-start problem.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Domain experts required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Requires domain knowledge.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If there are changes in the users' needs, it requires explicit feedback.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If there are changes in the users' needs, it requires explicit feedback.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If there are changes in the users' needs, it requires explicit feedback.</td>
</tr>
</tbody>
</table>

- Quality relies on the previous data.
- New user cold-start problem.
- Requires data about other users.
- It is able to suggest new and unpopular items.
Regarding the hybrid, it combines different techniques bringing the best of both worlds, overcoming the limitations of one technique with the benefits of the other and optimizing the system performance. The difference between the different types of hybrid techniques is the way it implements their sub-techniques (implements the techniques it relies on). For example, it can run the algorithms separately and, in the end, combine their results.

2.3 Evaluation of recommender systems

After we develop a recommendation system, it is crucial that we evaluate them, as stated by [33], [17], [1], [23], [24] and [12], and the first thing that we should do is define and consider the:

- **Hypothesis:** when we perform an experiment, we are testing a hypothesis that should be defined before the experiment.

- **Controlling variables:** the variables that are not tested in a certain hypothesis should not change.

- **Generalization power:** when we perform an experiment, we want to analyze its result and if it is capable of generalizing beyond the experiment context.

A recommendation system depends on various factors such as the user’s intent, context and the interface used by that system.

We have to distinguish between the possible approaches to an experiment and the metrics that we will use to analyze the results of an experiment.

2.3.1 Approaches

We will discuss the different existing approaches of evaluating recommendation systems.

1. **Offline evaluation**

   Instead of using real users to an experiment, a pre-collected user dataset is used to simulate the user behaviour assuming that the real user behaviour would be the same as the one collected. The main goal of this type of evaluation is being able to filter the most appropriate candidates, facilitating the future work in other experiments – user studies and online evaluation – because the set of candidate algorithms is smaller.

   (a) **Pros**

   Because there is no direct user interaction with the system, this approach is the less expensive, quickest and easiest way of conducting an experiment with a large set of users.
(b) **Cons**

Having no direct user interaction with the system, we can only evaluate the items that were already rated by the user introducing sparsity data problems. We cannot consider this approach to evaluate the real user satisfaction.

2. **User studies**

Despite the fact that an offline evaluation is the cheapest and easiest approach, we cannot forget that it just simulates an user interaction with a pre-collected data set thus it is not a reliable evaluation.

The benefits of this approach is the fact that it cannot only provide quantitative information but also qualitative. It is divided in two phases:

- **1st Phase:** gather a set of users. They can be volunteers or be paid.
- **2nd Phase:** the users will perform a set of pre-selected tasks, and the evaluators will analyse and register their behaviour collecting not only quantitative metrics- for example, time spent performing a task- but also qualitative by performing a questionnaire.

We have to guarantee that the subset of users that we choose for our experiment is a representation of the real population and not just users that are interested in the test domain. Another thing that needs to be assured is that the users do not give biased responses to the system just to gain something with it (in case that the users are rewarded by participating in the study) or because they feel uncomfortable. The goal of evaluating a recommendation system is testing and being able to improve it so, we do not want to promote unreliable and false results by revealing to the user the goal and the hypothesis of the experiment before or during the test.

(a) **Pros**

It tests the real user interaction and behaviour. It is the only approach that is able to extract qualitative measures about the system that will help when the quantitative metrics are analysed, reducing the need of various trials to collect neglected measurements.

(b) **Cons**

It is usual that, whenever someone performs an experiment, it has a large set of users. It is a usual mistake assuming that, with larger sets, we can achieve better results. This can or cannot be true, it only depends on the users sample that is used. If we consider a larger set and if the users are rewarded for participating in the experiment, that can be extremely expensive. If the users volunteer themselves, usually they do that because they have interest on the subject. Meaning that it cannot represent a feasible sample of the total population because it only considers interested users and not a wide spectrum.
When we are developing an experiment, we have to select carefully the set of users as explained previously. But we also have to select the set of tasks that are more relevant. It should be a good and a short set of tasks because, to produce reliable conclusions, we must repeat each scenario a couple of times. If the number of tasks is not small, it can require more user time, exhausting the user.

3. Other tests

(a) Pilot user study

When we are testing a recommendation system that is deployed within an application, it is mandatory that we perform a pilot user study. The main goal of this test is detecting malfunctions of the application. We have to guarantee that before we perform a user study, we conduct this pilot because we do not want code errors influencing what we are testing, the recommendation system, misleading the users and failing the experiment (collecting statistical data). When we are testing a recommendation system that is deployed within an application, it is mandatory that we perform a pilot user study. The main goal of this test is to detect malfunctions of the application. We have to guarantee that before we perform a user study, we conduct this pilot because we do not want code errors influencing what we are testing, the recommendation system, misleading the users and failing the experiment (collecting statistical data).

(b) Questionnaires

As stated before, to collect qualitative measures, we should perform a user questionnaire. This can be done whenever the evaluators find more suitable. It could be in the beginning, during or in the end of the experiment. We can also do one before and one after so we can state the differences and take conclusions about it. We have to guarantee that the questions are about properties that they cannot easily measure or are not measurable, for example, if the user likes the system, and are neutral, the questions should not have a correct answer because whenever a user realizes that there is a more correct answer, he tends to choose it.

4. Online evaluation

Despite the fact that with user studies we can test the user interacting directly with the system, they are aware that they are performing an experiment. Online evaluation produces better and more feasible results, being the most realistic approach because, instead of being in a controlled environment like in a user study, the users are selected randomly and can be unaware of the test that they are performing. This approach has a real interaction between the user and the system. There is no scenario previously made. It studies the user behaviour when he/she uses the system.
and collects direct measures. In online evaluation, we are measuring how the users acts and how his/her behaviour changes when he/she uses different systems (so we can compare them).

(a) **Pros**

As stated before, in this approach, we consider real users performing real tasks. That is why, this approach is the one that better measures the true value of a recommendation system.

(b) **Cons**

If the user is testing a system that provides irrelevant suggestions, he/she may not want to use it in the future. This is especially bad if we are considering systems with commercial context.

5. **Conclusion**

We should perform an offline evaluation to reduce the number of candidates approaches, conduct a user study to analyse the user behaviour when he/she is interacting with the system and then perform an online evaluation. It should be done a sequential approach to reduce and avoid the risk of user dissatisfaction when we are performing an online evaluation.
Background: Fashion systems
Although there are multiple RSs in various areas as showed before, there are not so many in the fashion domain. Nowadays, the e-commerce has a giant importance and fashion stores are one of the many areas covered by it. Therefore, these systems could be really helpful in this domain, not only because some users will find it useful, but because, some of them, do not have sufficient fashion sense to choose an outfit by themselves or because they find doing it a waste of time. The way we dress in our daily life, can say a lot about us. If we want to go to a certain business interview, we want to look reliable and professional, so it is not a good idea to wear something really sporty, we should wear something more casual. What we choose to dress, really speaks for itself and is used to convey an image about ourselves. Accordingly, some research has already been aimed at the development of a RS in this specific domain. Its purpose is to help users to dress properly without spending time thinking about it. [7]

In this chapter, we will analyse different fashion systems individually in order to explain each of them that took into consideration different aspects and apply different techniques, some of them that we introduced in the previous chapter.

In the end, after explaining each one individually, we provide a table that organises all the information regarding the techniques applied, input taken into account to be able to overview and compare them regarding those aspects.

1. **Mirror appliance: recommendation of clothes coordination in daily life**

The mirror appliance system [26] aims to produce a suitable outfit given a certain weather and event. Besides the fact that it suggests an outfit, it also stores photos of the user’s clothing pieces. To achieve that purpose, it uses a television with a touch panel and a Universal Serial Bus (USB) camera upon the television. The images that are taken by that camera are displayed on the screen. There are four inputs to the system: user’s schedule, temperature and weather, and user’s past behaviour. An important one is the user’s schedule, that can be “casual”, “business” or “formal”, and is given by the user showing a marker to identify it. They did not want to suggest clothes based on the season, so they obtain the temperature and weather from a web-service to recommend clothes specific for these conditions. When we are working with users and want them to like our suggestions, we need to have knowledge about their preferences, that is why, whenever they recommend an outfit, they store it together with the conditions stated before-temperature, weather and event- in a network storage. This user’s past behaviour has an important role in the recommendation algorithm.

After analysing the system structure presented in the figure 3.1, we can state that: The user stands in front of the mirror and the system identifies him. Then, the user shows a marker where it identifies his schedule for the day. With the information obtained from the web-service about the temperature and weather, and the user’s past behaviour from his/her personal database, the
system evaluates and produces a suggestion. The combination part relies on the dependency of the pairs because the top pieces are coordinated depending on the bottom pieces and vice versa. The system will first search for outfits, including tops, bottoms and shoes, that were worn in the past in the same conditions because they reflect the user’s preferences. If those pieces fulfil the weather and temperature conditions, the system recommends that combination.

Overall, the recommendation algorithm is poor because it only suggests clothes taking into account the user’s schedule, the temperature and weather and the user’s past behaviour, which does not introduce innovation when the conditions are the same as previously registered in the application: the system recommends the same combination if it was used in the same conditions before. It also fails in describing the clothes because it just classifies them by temperature, weather and event.

2. ChroMirror: A Real-Time Interactive Mirror for Chromatic and Color-Harmonic Dressing

The system's [5] main goal is suggests through its digital mirror, colour combinations that are harmonic.

After analysing the figure 3.2, we can state that the system is composed of a digital camera, to capture real images in real time of the user, a LCD display, to display the reflections, and a Wii
remote to help exploring the possible colour combinations that are suggested by the system. The user will stand in front of the liquid crystal display (LCD) display and will press the reset button in the Wii remote. Then, the system will recognise the clothing regions by the identification and extraction of the three items that are most visible. After that, the system will suggest a combination of colours and if he/she wants to explore it, he/she can press the up/down button to switch between colour harmonics and right/left to switch colour tones. It is the user that makes the final decision. The user can press the reset button again so the system restarts the whole process.

Although it helps the users in trying different colour combinations, it suffers because of the poor lighting and because it does not consider the texture of the clothes – it could affect the colours perceived by the camera and make the system less accurate.

3. Fashion Coordinates Recommender System using Photographs from Fashion Magazines

The authors [16] propose the creation of a RS that is able to, given a bottom or top piece, retrieve another top or bottom piece that is related to the one given. The system uses full-body photographs taken from fashion magazines. To learn information about coordinates, a probabilistic topic model is used. It is a "generative model for discrete data", that is able to find photographs of items that are similar to the ones given by the users.

As explained in the figure 3.3, because it uses photographs from fashion magazines, there is a need to identify only the ones that are full-body. For that, a face detection algorithm is used, to be able to identify top and bottom regions. For the coordinates, it uses the topic model. Each region has a certain number of visual features but in this case, it only takes into account the colour. The authors chose this model because it is able to extract information about the relationships between different items. In this specific case, the extension of latent Dirichlet allocation, learns the relationship about the features of the top and bottom items.

![Figure 3.3: Fashion Coordinates Recommender System. Image from [16]](image-url)
Basically, it has a set of photographs and applies the face detection algorithm to them. Each region now has its visual features, and the topic model is applied to learn the relationship between them. Then a user gives a certain piece, bottom or top and the system converts it into its visual features, evaluates the proportions and then it retrieves a matching item, top or bottom.

A pro of this RS is that it is less time consuming because it does not require the metadata of the items to provide the suggestions. But, the RS could use a more accurate algorithm to detect the bottom and top regions, it could consider other visual features, and could take into account the user’s preference and occasion.

4. Getting the Look: Clothing Recognition and Segmentation for Automatic Product Suggestions in Everyday Photos The authors [19] propose a RS that is able to, given a photograph, retrieve items similar to the ones present in the figure 3.4, cross-scenario retrieval, as we can observe in the figure.

This process has two phases:

- First phase: The system detects the items from a picture by classifying regions.
- Second phase: It uses image retrieval techniques to provide suggestions of similar products, taken from clear photos, to the ones present on the picture.

First, they will evaluate the two features, colour in Red-Green-Blue (RGB) and texture in local binary patterns (LBP). Each item is characterised by a feature vector. To find similar feature vectors they use a k-nearest neighbour index- randomised kd-trees and they only provide the top of the most similar ones within a class.

Figure 3.4: Getting the Look System. Image from [19]
5. **Hi, Magic Closet, Tell Me What to Wear!** This paper [22] presents an automatic occasion-oriented clothing RS. In this RS, it is crucial to determine to which occasion the user wants to wear an outfit.

It considers two types of recommendations:

- **Clothing suggestion:** where the user only specifies the occasion and the RS retrieves the suggested outfit from his own closet, as presented on the figure 3.5.

- **Clothing pairing:** the user specifies the occasion and gives an item, and the RS will pair it with an item from an online store that matches with the given input.

The system has a model learning process in which they consider middle level clothing attributes that are a 7-multi-value and include the category and the properties. The learning process is based on the unified latent SVM that embeds and describes four rules between the attributes, visual features and occasion. These rules are the ones used to achieve the two main principles of the RS that are wearing properly and aesthetically.

6. **Clothes Recommend Themselves: A New Approach to a Fashion Coordinate Support System**

This system [11] will present the user with different purpose options and a colour palette that the user will choose. But it also bases his suggestions in the temperature and clothing that is acquired from the Internet. This colour palette is the one that will provide the user feelings. After he chooses the colour, the system will retrieve and insist in the clothes that best matches the one provided and the occasion. If someone touches certain clothes, it will insist on them. If the users chooses a certain bottom piece, the system will start to insist in a top that matches that bottom piece. When he/she finally chooses the bottom, the complete outfit is decided and the system will store it with information regarding the purpose/occasion and the date. It is a different system because unlike
the others it takes into account the possible resilience and resistance of the user when the system suggests an outfit. The system will make sure that it is the user that decides which outfit he/she wants to wear because he/she is the one who makes the final decision. Despite that, there are many things that are missing in this system. One important thing is that it does not learn the user preferences. The self-recommendation from clothes is outputted by the system voice and the content is elected from the conversation template according to the condition of arriving at the candidate.

The objects candidates, based on the same hue and value, include three colours that match the feeling given by the user. In order to make a candidate, a colour image has to be congruent. The user has to select the correct dress code to a specific situation. The clothes properties decided upon are then analysed with a certain priority. The user has a major paper on this system because it will take into account user's past behaviour. The first clothes are chosen appropriately by their purpose and are signed as a candidate. The outcome is retrievable by the voice, and the content is elected from pattern conversation.

It is a different system because unlike the others it takes into account the possible resilience and resistance of the user when the system suggests an outfit. The system will make sure that it is the user that decides which outfit he wants to wear because he is the one who makes the final decision. Despite that, there are many things that are missing in this system. One important thing is that it does not learn the user preferences.

7. **What am I gonna wear?: Scenario-Oriented Recommendation**

“What am I gonna wear” [34] is a system that uses a novel technique called scenario-oriented by the authors, where each user has an online wardrobe with all his items and outfits. The system retrieves suggestions based on the occasion (“I am going to”) and on the style that the user wants to achieve (“I want to look more”). This information is given by the user in natural language instead of check-boxes or forms. It is based on the common sense reasoning using the Open Mind Common Sense (OCMS) semantic network that provides 20 link types, they are the ones that provide the relations between events, among others. The system has a style sensor that is used to match the clothes that the user owns to the style that he/she has written. The style is based on four types of information: brand, type, materials and natural description to a specific occasion. The dimensions of the style are the following: luxurious, formal, funky, elegant, trendy and sporty. Meaning that, each tag will have a value between 1 and 10 and the result style to a certain brand, type, material, description if provided, will be the average of all the values. It uses the common sense reasoning that is able to, given any word in English, guess the style that it is talking about. Spread activation is used to achieve style inferring and for each procedure, the style value of a certain node expressed in ConceptNet will be propagated to the neighbour nodes with a factor of discount of
0.25. The system also has a sensing function that is used for clothes and occasions. It employs three relations: used for, location of, capable of receiving action, from ConceptNet. It will find the occasions to each clothes depending on their types and, when a user writes his description, the system will match them both.

“What am I gonna wear” learns the user’s preferences because the items that have similar styles are linked in an occasion network. Whenever a user decides to wear a certain outfit, the system will attach the occasion to those items and that will be spread activated through the links between the items in the network. They also learn the user’s tastes because of the user’s textual profile and the average style of the user wardrobe.

As shown in the figure 3.6, the user specifies, in English, the occasion and style that he pretends. The system will try to match the description but it will only provide suggestions if the style value is above a certain threshold. Otherwise, the system will not provide any suggestion and will leave that decision to the user. When the user agrees with the suggestion made and decides to wear it, the system stores the description about the occasion decided by the user with those suggested items, as illustrated in the figure 3.7, to keep the user preferences and achieve better results next time.

Figure 3.6: What am I gonna wear System. Image from [34]

Figure 3.7: What am I gonna wear System. Image from [34]
This system uses the common sense reasoning to achieve its goals, which are: understanding the user goals and providing suggestions based on them. It is different from CBR because it uses a general common sense corpus that acts as a generic user model, instead of a repository of relations between scenarios and products, having the advantage that it uses natural language to the description. However, choosing values and the style for certain clothes is subjective and not well accurate.

8. Personalised Clothing-Recommendation System based on a Modified Bayesian Network

This system [38] suggests combinations that took into consideration temperature, weather and occasion from a user’s personal closet depending on his/her past behaviour, his/her personal preferences and by the feedback given by him/her about previous suggestions through a bayesian network.

A bayesian network is able to use parameters about items clothing suggest clothing depending on the date and type of clothing because its nodes include information regarding the item properties extracted using RFID tags and weather.

As seen in figure 3.8, the algorithm is divided into two parts:

- first, it selects the top piece based on the clothing properties, the context (occasion, season and temperature) and the record of recent suggestions to avoid recommending pieces too frequently being able to by a specific probabilistic function assure that items will be recommend with the same frequency.
- after choosing a top piece, it will be searched a bottom piece that not only matches the top piece but also the occasion. It will combine colour and pattern being based on his/her past behaviour and feedback.

The fact that it provides a feedback feature is a great advantage because if the user dislikes a certain combination that could be regarding the colour, pattern or both, it will provide that feedback.
to the system specifying which combination property he/she dislikes and it will change the strategy never recommending that combination again.


This recommendation system [39] bases its suggestions on the user’s schedule applying a content-based method.

It will be needed to have a virtual wardrobe which implies adding pieces to it. The user will then need to describe the item features such as its color, style, occasion, function, season, among others. To be able to provide a recommendations consisting of the top two combinations, the user will need to provide his/her schedule. The fact that it is a content-based assumes that it will be identified items that suit the user preference through an analysis of the item descriptions and the user profile. This user profile is being fed by the feedback that the user provides after receiving a recommendation: if he/she does not like it, he/she can change the items to his/her favourites.

10. Style Recommendation for Fashion Items using Heterogeneous Information Network

It is a recommendation system [20] that is able to learn how to match different item pieces relying on a neuronal network.

To be able to produce this model, it was needed to have a data collection phase in which it was built a dataset of items composed by its description- category, material, pattern, and colour- and an image. The colour is extracted by a colour extracting tool and will be stored as a weighted multi-color vectors that will then be clustered using k-means clustering. Using a meta-path concept, they are able to infer the relation and ensemblence of different item combinations regarding its category, color, pattern and material.


This system [8] provides recommendations based on the occasion or on a top piece that the user want to be provided a match.

It will be needed to store the users’ items that is a time consuming and boring task. That is why this system incorporates gamification techniques to be able to address and minimise this problem encouraging the users to add their images. It uses novel interaction based on the user emotional status. Basically, it will shown the user an image and the mood name and it will then add the items that he/she would use in that specific situation. This makes the task of annotation more interactive and catchy making easier to gather the user’s items. It will then have information in which moods will the user use certain items.

12. Analysis of fashion systems
We decided to organise in a table as seen in 12 and 3.1, all the important information regarding the fashion systems such as the techniques applied and the variables inputed by the user.

After analysing all the referred fashion systems, we are able to conclude that almost all of them have a knowledge base that we talked about in the previous chapter to be able to store the items and perform queries. But there was also a system [39] that used a content-based technique that we also discussed in the previous chapter. The systems that have better results are the ones that apply learning algorithms to be able to combine pieces.

As we can see in our table 3.1, some characteristics that we took into account in our recommen-
<table>
<thead>
<tr>
<th>Papers / Features</th>
<th>Papers / Features</th>
<th>Papers / Features</th>
<th>Papers / Features</th>
<th>Papers / Features</th>
<th>Papers / Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Occasion Schedule</td>
<td>Style</td>
<td>Temperature</td>
<td>Weather</td>
<td>Color</td>
<td>Piece Photograph</td>
</tr>
<tr>
<td>&quot;Mirror, appliance&quot; system [26]</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;ChroMirror&quot; system [5]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Fashion Coordinates Recommender&quot; system [16]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Getting the Look&quot; system [19]</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&quot;Hi, Magic Closet, Tell Me What to Wear!&quot; system [22]</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;What am I gonna wear today?&quot; system [34]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;What to Wear in Different Situations&quot; system [39]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Style Recommendation for Fashion Items using Heterogeneous Information Network [20]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Hi, Magic Closet, Tell Me What to Wear!&quot; system [22]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Clothing recommender system&quot; [16]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Clothing recommender system&quot; [16]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Mirror, appliance&quot; system [26]</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 3.1: Describing systems regarding its features.
dation system are present her: occasion, style, temperature and weather. But the emotional status is only considered in the Moody Closet. Even though we also ask the user in the annotation phase in which emotional status she/he will use a certain piece, Moody Closet uses a different approach.
Inside Fashion system

Contents

4.1 System description .................................................. 37
4.2 System architecture .................................................. 37
4.3 Creation and storing user’s wardrobe ............................ 40
4.4 Recommendation task ............................................... 42
4.5 Social Network ....................................................... 55
4.6 Organisation .......................................................... 58
4.7 Voice ................................................................. 60
4.8 Interface .............................................................. 63
In this chapter, we will go into detail into our solution.

First, we will overview, as already talked in the introduction chapter, what is our application: **Inside Fashion**.

Then we will describe its architecture followed by the analysis in detail of our system – which includes our recommendation task, social network, organisation and voice feature – in which we specify our algorithms and explain our decisions.

Because we developed an application, we also took into consideration interface details that we talk in section fully dedicated to it in which we analyse and explain our decisions through the incremental and iterative development.

Lastly, we will discuss possible adaptations to use the application in personal desktops and laptops.

### 4.1 System description

As described in the introduction, we will develop a recommendation system within an application that we decided to call **Inside Fashion**. **Inside Fashion** will consider the following features in its recommendation:

- Temperature and weather of a certain location
- Style wanted by the user
- Occasion attended by the user
- User emotional status

The main idea is being able to suggest outfits from the user wardrobe according to the features that he/she selected. Because of that, we had to divide the process in two steps:

- Creating and storing user’s wardrobe
- Recommendation and combination task

It will also help the user organise and visualise her/his wardrobe. It also offers the possibility to the users of following other users.

### 4.2 System architecture

**Inside Fashion** is an application that is able to suggest outfits according to certain characteristics, specified above, chosen by the user. Also has the ability to organise the user’s wardrobe and to follow other users introducing social connections.
Inside Fashion is a distributed application. The client will be the users mobile application developed as a web-application, being available to IOS and Android operating systems, using Javascript as main language. The application server is deployed with Node.js using the express module, where it can also be found our database in MongoDB.

It is a RESTful system that will provide RESTful web services such as the login, register, recover password, annotation, recommender task, among others. All the data that is exchanged between the server and the application will be represented by JSON objects. The information flow is as presented in 4.1.

![Figure 4.1: Flow of information](image)

InsideFashion has a model-view-controller architecture, 4.2, meaning that has three different layers because it separates the different types of information as needed in an interactive application.

![Figure 4.2: InsideFashion architecture](image)

- **Controller**: it receives the requests, will execute them and transmit the changes to the correspond-
ing view. If needed, it will communicate with our Mongo database, that is the model layer. This layer is composed by all the application services such as, for example, the recommender system.

- **Model**: it will process data, manage logic and respond to the controller. Data includes all the information about the users, the pieces and the outfits.

- **View**: based on the changes made previously, it will generate a new view to the user.

InsideFashion is deployed in Google Cloud Platform \(^1\) and is available online \(^2\).

The technologies that we used, as previously mentioned, were not our first option. Initially, we used Ionic \(^3\) to develop our application. Ionic 2 uses Angular \(^4\) which was a framework that the developer never used before which lead to a period of time learning this technology. Because some features, were not available to develop in Ionic, we decided to build an web application with a view. The technologies later used to build InsideFashion were already known by the developer because she already used it before.

Next, we will talk about the creation of an wardrobe and the recommendation task followed by the other features that our system provides.

Before we go further, there is a need to clarify our solution in terms of algorithms. As analysed in the second chapter, there are many techniques that can be applied regarding RSs. As we could see in chapter 3, most of the existing recommendation fashion system use knowledge bases as we will use.

![Diagram](image)

**Figure 4.3:** System logic: algorithms

As we are constantly reminding 4.3, our goal is to suggest outfits according to four variables: weather and temperature, occasion, emotional status and style. For that to be possible, we will need to build a knowledge base to each user corresponding to her/his wardrobe. To be able to build a knowledge base one needs to annotate each piece of clothing according to its characteristics. After that annotation phase, the system is now able to provide suggestions. Since we are using a knowledge base there

---

1 [https://cloud.google.com/](https://cloud.google.com/)
2 [http://35.189.201.239/](http://35.189.201.239/)
3 [https://ionicframework.com/](https://ionicframework.com/)
4 [https://angular.io/](https://angular.io/)
are variables that our system takes into account- weather and temperature, occasion and emotional status - that will correspond to queries performed into our knowledge base. Those queries will retrieve the pieces that satisfy those restrictions. But, as previously mentioned, we aim to suggest outfits not single pieces of clothing. Therefore, it will be applied to those pieces a combination algorithm which depends on the chosen style. After the combination algorithm, one of our main contributions, is applied, outfits are retrieved. And, since our goal is to develop a system that is personalised, the user is able to provide feedback.

### 4.3 Creation and storing user’s wardrobe

![Figure 4.4: Add a piece of clothing in InsideFashion](image)

In order to provide good suggestions, there is a need to well characterise each piece of clothing as shown in figure 4.4. That is why, before suggestions are retrieved to the user, he/she will have to take photos of his/her pieces of clothing because each one needs to be characterised according to its characteristics 4.5. The major part of the characterisation will be done by the user and other automatically.

- **By the user:**

  The user will specify the type of piece, material, pattern, which occasions, temperatures and seasons will he/she wear it.

  After analysing the existing fashion systems, we decided that the user should characterise the **type** and the **piece, material** and **pattern** because the focus of this thesis is not the processing image field. Type, pattern and fabric are used in the style variable.
The user will also specify in which occasions he/she would wear those pieces of clothing because it is subjective and each user has different tastes, one user can use a piece in a formal occasion and other in a not formal occasion.

Termal sensation varies depending on the person, that is why we decided that each user must specify in which weather and temperature conditions will he/she wear that piece of clothing. We also added the season because, if the user is planning, with the help of our system, to choose an outfit for a distant day, we will not be able to know the weather and temperature - there is not yet available forecast information-, it is only possible to analyse the season and use it in the recommendation task.

- **By the system:**

  - **Colour**

    The only feature that will be known automatically will be the colour. The colour of the piece will be inferred using image-processing techniques.

    Basically, the system analysis the image and will display the computed colours in a pop-up, as shown in figure 4.6. The user will validate and choose the one corresponding to the main colour if it does not have a pattern or the background colour if it does have a pattern.

    The user validation is necessary to guarantee that the colour chosen is right given those two possible situations, patterned or not, as shown in figure 4.6. Colour will play a big role in what concerns the emotional status variable and style - more specifically in the combination phase.
4.4 Recommendation task

We have many features to take into account when providing a recommendation to the user. Because they are so different and complex, we decided that we needed to partition this task in different phases. The knowledge about the different features was not obtained at the same time meaning that we divided the work into different phases. In each phase, we analysed each one of the features.

We developed our algorithm in two phases as shown in figure 4.7. Assuming that we build a knowledge base to each user because it will need to characterise each item according to many features. We divide our general algorithm in:

- Recommendation itself: given the fact that it is a knowledge base, we will perform queries according to the options selected by the user in the recommendation task. Meaning that our recommendations are the results of the queries;

- Apply our combination algorithm to the pieces retrieved by our previous algorithm: the pieces that satisfy those requirements will be inputted to our combination algorithm. Because we consider three different styles and because it is the combination that ultimately defines the style of an outfit, we have three combination algorithms that receive those pieces retrieved previously and analyse if they match for the selected style and if they do, they will be retrieved as an outfit.

We will first explain our first phase that consists of a recommendation algorithm and after that the combination algorithm itself.
4.4.1 First phase

After we have our knowledge base built, we can perform certain queries to it regarding the temperature, weather, occasion and emotional status.

One of the features to take into account was the weather and temperature of a certain location. We get this information whenever a user specifies a location and the day. The user is able to specify a certain location because we provide a map in which he/she can select a certain position corresponding to the specific location. The system stores the longitude and latitude. A calendar is also provided for the user to specify a certain date that is stored by the system.

We use the DarkSky API to obtain all the forecast information. With the longitude, latitude and date we are able to make a request to the API to obtain the temperature information and the forecast icons. With the retrieved forecast icon we map it into its corresponding forecast. If it does not have yet forecast information available about that day, the system will only consider the season to make the suggestion. In the end of this feature, we are able to retrieve all the clothes that satisfy those values.

Occasion was other feature that integrated our system. The user specifies in which occasions he/she will wear that piece so we will retrieve pieces that the user would wear in the occasion selected.

Regarding the emotional status, we had to analyse this situation more deeply. Given the fact that this is an unexplored field, we could not find any work that showed a direct connection between

---

5https://darksky.net/dev
the emotional status and the clothes we wear. With that in mind, we performed a study in which the main goal was to understand the connection between these two fields. Even though only a simple web application was necessary, because the focus was the results of our study, we used the material design guidelines in order to have a simple and easy web application using HTML5 and Javascript. To present our questions we used a carousel slider. We did not use Google Forms due to the fact that we wanted to have questions that appeared depending on previous ones and presenting tooltips to the user so we rather spent more time building our own form instead of using one that did not give us the tools that we needed to well perform our study. We hosted our web application in the Google Cloud Platform and all the answers to this study were stored in a database. In order to analyse the answers and because they depend on each other, we had to analyse the JSON objects stored manually. The questionnaire was shared through friends and colleagues. The questions made are present in the appendix.

After analysing 110 answers, in which 61 were women and 49 men, we concluded that people wear different clothes in different moods, happy and sad. With 73 that agreed that consciously and unconsciously we wear different clothes according to our emotions, we analysed that people that are sad use less patterns and darker colors and when they are happy they use more patterned and colored clothes. With this survey we were able to obtain the answer to a question that makes a big difference in the way we recommend according to emotional status, that was: is it possible to modify user emotional status through clothing? 89 percent said that it was not possible.

---

6https://material.io/guidelines/
7https://cloud.google.com/
Regarding the analysis of the obtained results, we recommend darker and without pattern clothes when the user is sad and lighter and patterned clothes when the user is happy. Meaning that the item characteristics needed to take into account were pattern and colour. The pattern is provided by the user but the colour is obtained automatically and approved by the user. Retrieving clothes with or without patterns meant that we needed to perform a query that took that into account. But, in order to obtain darker or lighter clothes, we needed to change the colour model used. The HTML standard is RGB coded in hexadecimal. With RGB, an additive color model that considers the quantity of the three primary colors – red, green and blue–, we could not directly suggest lighter or darker items. For that, we used other model of color : HSL - hue, saturation and lightness. HSL has a field named lightness that varies between 0 and 1. Lighter colors are closer to the value 1 and darker are closer to the value 0. The model that we could use to obtain darker or lighter clothes was the HSL because one of its fields is the lightness, as shown in detail in figure 4.10.

Thereby, we used a function that performs the transformation from hexadecimal to hsl and save the lightness value being able to use it in the recommendation.

![Figure 4.10: Emotional status](image)

In a recommendation considering the weather, temperature, occasion and emotional status, the first three queries are performed as specified before and then it is performed a query regarding the emotional status. Basically, it will order the recommendation queries results obtained by the lightness value, it will then suggest the n items that are lighter or darker depending on the emotional status selected.

In the end of this process, we will have the pieces of clothing that satisfy all of the selected parameters.

### 4.4.2 Second phase

After having the results from previous queries, our combination algorithm, that is based in the style, will be performed.
We previously discussed the fact that it is not just a type of piece that gives the style, it is the combination that ultimately does that. Based on that, we decided that we would have three general styles: casual, classic and modern. We decided to focus on just three styles because there is a wide variety of styles available which characteristics are hard to define. With those three styles we can more easily know its distinguishing characteristics. For that, we did a research on each style analysing and searching outfits of each one to assure that we were able to extract the main rules.

Before we created our algorithm, we needed to explore fashion field to be able to produce an efficient and accurate algorithm and it took a lot of research.

**Classic style** focus on pieces with more conservative lines that give a clean look to a person not only because of the abundance of neutral colours and lack of patterns but also because of the type of pieces that is more formal. It is a style that is more simple but it is also exquisite, elegant and distinctive. It does not include any trends, it is a timeless look as we can see in figure 4.11.

![Example of classic style outfit](image)

**Casual style** is kind of an easy look with less worries and much more practical. Unlike the classic style that aims at being immaculate, it has a relaxed vibe providing an effortless look. A quote that would describe this style would totally be “keep it simple”. It does not pay attentions to details choosing comfortable clothes without patterns or minimalist patterns such as stripes as seen in figure 4.12.

**Modern style** is a more vibrant style that give us colourful clothes and patterns. It is all about incorporating trends. We can mix patterns and textures and colours. It gives attention to the details and the “on-fleek” characteristic that classic also provides but in a different way. People that have this style are not afraid of innovation and like to be creative in their looks, often combining different type of pieces.
that create great outfits as the one seen in figure 4.13.

After analysing each style, we came to the conclusion that it was indeed not just the pieces itself but the way they combined that really corroborate the style. With that in mind, we decided that our algorithm would work in two different phases: annotation phase and recommendation phase.

In the annotation phase, in which the user describes a clothing piece, our system will analyse its type, pattern and fabric as shown in 4.7. With those characteristics, it is able to evaluate according to each style if it has those previous requirements to belong to that style. For example, blazers are a type of piece of clothing that is usually used in classic and modern styles meaning that this piece would belong to the classic and modern database. That means that one piece can address one or more different styles and if it addresses it has an entry in each style database.

So we built model to each style that predicts if a certain piece meets the characteristics needed to belong to that style. The variables used to predict if it belongs to a style are the following item’s features, as specified before: type, pattern and fabric. Our models only handle a item not a combination of items.

The other step on our combination happens in the recommendation itself. Because we have three different styles, we have three different ways to combine pieces meaning that we have three different combination algorithms.

On the classic style, it is usual the black and white combination. That is why, when it is selected a classic style, it will try to search for pieces that are black or white and combine them to other white or black pieces. If not, it will combine pieces that have a similar background colour. In this step, we already have the pieces that fill the occasion and/or emotional status and temperature and weather.
It will basically search on the pieces retrieved by those previous queries the ones that belong to the classic style database. This is not enough because having a classic style and fulfilling the previous requirements are not the desired. We could not have random combinations of classic style pieces there is why we have to apply rules to combine them.

The main difference between the casual and classic style is the type of pieces whilst in a casual look it is usual to wear a pair of jeans, we do not see that in a classic style. It also accepts to incorporate some minimalist patterns. The way to combine them is exactly the same in terms of colours, the only difference is the fact that it can have a patterned item in an outfit.

The modern style embraces colour and patterns that is why the main difference between the other styles is the fact that it accepts the diversification and combining pieces with same pattern, different textures. Here we can mix a patterned item with a plain piece with the same background colour or white or black colour or with one with the same pattern.

Even though they are different combination algorithms, the colour combination rules are the same: a white or black piece with other coloured piece, pieces with a similar background colour. The colour of a clothing piece is defined by the HSL model meaning that different colours have different HSL codes. Because we want a similar colour, we analysed and defined a threshold in the sum of the difference of each field. If it is less than that threshold it will have a similar colour and can be used in the combination. This threshold was a result of several tries applying different thresholds and comparing the results.

\[
\left(\|\text{hue}_1 - \text{hue}_2\|\right) + \left(\|\text{saturation}_1 - \text{saturation}_2\|\right) + \left(\|\text{lightness}_1 - \text{lightness}_2\|\right) \leq \text{threshold}
\]

The main difference between the three different algorithms is the pieces in which they execute the
algorithm and then each one of them have some specific characteristics which made it necessary to separate the combination algorithm to each style.

Meaning that, to each style it will 4.14:

• **1**<sup>st</sup> In the annotation phase, it will analyse type, fabric and pattern and pre associate to each style.

• **2**<sup>nd</sup> In the combination, given the style, it will combine the patterns differently according to the style.

• **3**<sup>rd</sup> It will apply the function that is equal to all styles that measures the difference of colour between the background of the clothing pieces and will only retrieve the ones which difference is less than the threshold.

---

**Figure 4.14:** Scheme of the style logic

Because here our focus is not on the scalability, performance and velocity of our algorithm, it will only retrieve our combinations after analysing all the possibilities meaning that we can have, depending on the virtual closet, hundreds of possible outfits available to the user. This is good because we do not restrict the user to a limited set of outfits in case we defined a limit of maximum outfits. Thus, the user has the possibility to analyse each outfit and decide which one he/she will wear and save it in an album. If we set a limit, it could happen that the combination that he/she would prefer were not retrieved but if that limit was not implemented it would see them.

An important piece on any recommendation system is being able to build a user profile because that is what distinguish between users and another element that provides personalisation.

To generate a user profile, we implemented a technique for the user to be able to provide implicit feedback. Whenever our system retrieves suggestions, the user has the possibility to express if his/her impression is negative. If he/she wants to never use that outfit, he/she can press the dislike button.
meaning that that specific outfit will never be retrieved again independently of the circumstances (options selected: date, occasion, emotional status and style).

What our system does is getting the ids of those outfit pieces and store them in a database. Which means that each user has a database of outfits that she/he does not like and will never be retrieved by our system again. That database is composed by entries corresponding to the outfits – set of pieces’ ids – that he/she previously disliked. Before our system suggests outfits, it will search if they match any outfit on that database and only if they do not match, they will be retrieved.

Our algorithm retrieves combinations considering classic, modern and casual. We present some of the combinations 4.15, 4.16, 4.17 and 4.18 that were created by our algorithm depending on the style selected.

![Figure 4.15: Modern combination by our algorithm](image)

We decided to implement this solution because it was the one that seemed more accurate. This was our final solution but we had other solutions in mind. For example, in order to create outfits according to a specific style, it could be created a database to each style in which they had outfits specific to that style. Those outfits would that be found on the Internet. Then, it could be applied a function of similarity between items that the user owned and compared to the ones from the style database depending on the style selected by the user. If it met a certain similarity, it would retrieve those items because they were the ones similar to the ones stored in the database. By implementing such solution, we would not need to create a combination algorithm because the database has pre-created combinations. The main reason why we did not do that is because it was hard to have those style outfits and even harder to have those pieces alone to be able to describe them because we would also need to annotate those ones in order to in the future apply the similarity algorithm between the items- content-based techniques as discussed in the second chapter.
Building a combination algorithm is more challenging and hard but could generalise more and be able to produce better results that did not depend on the number of outfits stored in each style database. We performed a study to analyse if the users were able to distinguish between styles and that is what we will discuss below.

**Style study**

Previously, we discussed in detail how our combination algorithm was created. Given the fact that this is considered a creative process that does not have rules that could be easily mapped into an algorithm, we needed to evaluate the algorithm that we created.

For that, we had to go to clothing stores websites such as Guess \(^8\), Tommy Hilfiger \(^9\), Moschino

\(^8\)https://www.guess.eu/en/
\(^9\)https://es.tommy.com/
Figure 4.18: Modern combination by our algorithm

10, Zara 11, Bershka 12 and Stradivarius 13 to be able to collect items. We rather use this instead of photographing our own clothing pieces because it was less time consuming and the images present in websites have better quality than the photographs that we would take.

To be able to store the images, we did go through the images on the websites and store them manually in a database.

After saving them, we need to do the annotation. Given the fact that our model only takes into account in the annotation the type of piece, pattern and fabric, we only annotate our pieces regarding those aspects because it was already a time consuming task.

After that, as we know, the other step to achieve a style is in the combination. So we only performed a recommendation considering this aspect: style.

We analysed the combinations that were in overall good even in the color combination and selected a couple of combinations to present in our questionnaire.

We developed a questionnaire using Google Forms 14 in which we presented two type of questions:

• given an image, the user needed to identify the style, as seen in the figure 4.19.

• given a style, the user needed to identify the image corresponding to that style, as seen in the

---

10https://www.moschino.com/pt
11https://www.zara.com/pt/
12http://bershka.com/
13https://www.stradivarius.com/pt/
14https://www.google.com/forms/about/
Our questionnaire is available on the following link  and the screenshots are available below. It was answered by 21 users.

The first combination, 4.19, was clearly a modern one and our algorithm retrieved it as a modern, although there were 4 people, 4.20, that considered it casual.

The second combination, 4.21, was retrieved by our algorithm as a classic style and all of the users, 4.22, consider it as classic one too.

The third combination, 4.23, was a modern style outfit and 16 users, 4.24, consider it as a modern style too.

The forth combination, 4.25, was retrieved by our algorithm as a casual one and 18 users, 4.26, agree.

The fifth combination, 4.27, was also retrieved as a casual combination. This was one of the combinations that we decided to put because it could fit into two styles: casual or modern, there is a fine line between them in this specific case even though we consider it more casual than modern but we were not sure if we were influenced by our algorithm so the answers to this question would be needed to figure it out. 14 users, 4.28, consider it as casual concluding that our algorithm and us agree.

The sixth combination, 4.29, was retrieved as a classic combination. 17 users, 4.30, agree.

https://docs.google.com/forms/d/e/1FAIpQLSekFclF7FZ6pXqH4zGeX9vPlOOGZJ2VM1MdwXpm3ivsmwccvSd6u4Aw/viewform
The seventh combination, 4.31, was retrieved as a modern combination. 11 users, 4.32, agree but 6 users consider it as a classic combination. Even though this was the question where there was more divergence, it is clearly modern because of the pop of colors and patterns, that mixture gave the combination a modern look.

In this question, 4.33, 13 users, 4.34, considered the second option a classic combination as our algorithm did.

In this question, 4.35, 14 users, 4.36, considered the first option a modern combination as our algorithm did.

In this question, 4.37, 16 users, 4.38, the first option a classic combination as our algorithm did.

This evaluation was performed to assure that our algorithm in those examples as well performed and also to understand if people know the difference between different styles. The only question that we did not knew if the majority would agree was the fifth, 4.28, but 14 users agreed.

We conclude that 67.16 percent of the users were able to distinguish between styles. And that the difference between the casual style to the classic one is well perceived by the users but, the difference between modern between casual are not so well understood.
4.5 Social Network

Nowadays we live in a world where there is more social life in the Internet that in the real world. In Portugal, the use of social networks tripled in seven years. There are 83.79 per cent of the population that uses Internet, that uses social networks [31].

A social network is defined by a social structure composed by nodes and edges. Nodes correspond to actors or subjects and edges to relations between them. As a network, there is a focus in the relations between subjects rather than in the characteristics of a specific subject.

Combining social networks with computer networks, in which nodes are able to share resources, results in the creations of social network sites usually reported social media.

Social media provide resources to people to share any kind of information through virtual communities. There is a wide range of social network sites available online that meet different goals. Some of the most used social media platforms are Facebook [16] and WhatsApp [17] that count with, respectively, 2,006M and 1,300M daily users [25].

Pinterest with 175M daily users launched in March 2010. Even though it is considered a social network, CEO Ben Silbermann disagrees with that statement considering Pinterest [18] a “catalog of

---

[16] https://www.facebook.com/
[17] https://web.whatsapp.com/
[18] https://www.pinterest.pt/
ideas” [30], as shown in figure 4.39. Pinterest’s main goal is to get inspiration from posted images. Despite the fact that Ben Silbermann does not consider it a social network, the truth is that Pinterest has the characteristics of one: it has a community of 175M that can follow other users and/or his/her albums, create our own public and/or private albums, share and search for posts using tags. It also has a messaging service in which we can talk to other Pinterest’ users. Being a way of sharing posts and being able to search for specific content by using tags, Pinterest is used to share and get inspiration through their fashion posts.

Chicisimo 19 is a fashion application that allow us to share our looks, follow users’ albums and save looks to our own albums. The main goal is to help the users to decide what to wear regarding the outfit ideas shown in their app.

We wanted to develop an application that was not just a recommendation system but that also could be used to get some inspiration from others. We believe that when we connect to other people, we learn and the way we could obtain this from our application was if it had a social network incorporated.

In our application social network, we have the following notions and relations, as shown in figure 4.40.

An user has a username associated and can create his/her own virtual wardrobe by adding his/her items to it. The main goal of having the possibility to create albums is the fact that it helps to organise not

19https://www.chicisimo.com/
only the items but the outfits for later inspiration and use. But the albums’ purpose goes beyond that, at least in the social media context. The user’s albums are the way that other users see him/her because each user has a profile composed by his/her albums, as demonstrated in figure 4.43.

Instead of having friends as on Facebook, here the user follows other users as on Pinterest and Instagram, which means that the connection between users is not bidirectional but unidirectional. A user can follow other user that does not follow him/her back, as exemplified in figure 4.42.

Using followers instead of friends makes totally sense in our application because our social network goal is to get inspiration from other users which sometimes means that we can have some interest in a user but it does not have an interest in our profile. We have a feed feature, as showed in figure, in which we are able to, whenever someone we follow adds an item or outfit to his/her albums, it appears in our own feed.

If we had the friends dynamic that would mean that in the previous reported situation, even the user that is not interested in the other user profile, would see his/her updates in his/her own feed even though he/she did not want it which does not make sense in our context that is why we adopted the followers dynamic.

In order to follow other users, we joined a search tool in which we can search for other users using their usernames.
Rather than searching for the whole word inputed, we decided that it would retrieve all match cases. This decision was made because a user can only know other user's username partially. In that specific case, if our tool just showed the whole world match, it would not retrieve that specific user profile but, if existing, the one that had the username equal to the inputed word. That would mean that we had a lack on our feature.

4.6 Organisation

To be able to recommend there is a need to share the user's pieces of clothing but, being an whole application the goal goes further than the recommendation itself. We need to store the clothes and making them available to the user. We could simply have a view where all the items were shown to the user with no specific order. But we wanted to organise their closet something that sometimes it is hard in the real life as we discussed in the introduction chapter. It is hard to remember all of our items and to organise them.

As explained before, whenever a user stores an item he/she needs to photograph it and then characterise it according to: type of piece, occasion, pattern, season, among others that are irrelevant here.
So we took advantage of that previous made classification and made available a view, as we can see in figure 4.44, where the user can search for their items taking into account their classification. For example, if the user wants to see their bottom items, he/she will go to type of piece and click on the option “Bottom” and the system will retrieve all pieces which type of piece is bottom, as shown in figure 4.44. In this new view where all items meet that specific chosen characteristic, is shown the images of the items and its characteristics, as shown in figure 4.45. The same applies to all other options.

It is almost impossible in a real physical wardrobe to organise it by all of these characteristics and remember all that meet certain variables. This will help not just to organise but will also help the users in their future buys because they will have a better perception of what they actually have which is one of the problems that we specified in the motivation. The application is always available in their smartphone and they can bring it when they are shopping and compare their own items and choose the ones in the shop that have a better match with their own.

We want to reinforce something that we already mentioned in the introduction. The ultimate goal of the recommender is to help the users to learn, have a more sense of fashion and be able to apply it long-term in their daily life. There is when our application continues to help them.
To organise items and outfits in a personalised way because we wanted to create not just a general organisation but personalised because each wardrobe and person is different, we made available a feature that consists of creating different albums, showed in Figure 4.46, in which we can have items and/or outfits in it. This is good for user's organisation, for example, he/she can add specific items that he/she would wear to a special event in an album.

The application is no longer just a recommender system but a tool that helps users to organise and make better decisions regarding their shopping buys.

4.7 Voice

There are many applications that already use voice recognition software that turn speech into text. One of the advantages of using this kind of software is the simplicity and because it can minimise the time of performing an action if the software is accurate. In our application, we also wanted to incorporate a voice recognition. We have two main tasks that require user work: storing an item and
get a recommendation. Our main goal is to have an application that is easy to use and give a good experience to the user.

The task of storing an item even though is the most time consuming task, we will not use this tool here. It needs at least $2 + 9 \times 2 + 1 = 21$ clicks, one for taking a photograph and save it plus the nine dimensions needed multiplied by the number of options selected that are at least one plus one to save the item. Introducing a voice recognition here would increase the possibility of having errors because this task requires many inputs – 6 – and would be hard to the user to remember all the possibilities available in each option. Instead of increasing the productivity and having a wide range of options that could easily be misunderstood because there is no one hundred percent of accuracy in speech recognition.

Getting a recommendation, as shown in figure 4.47 only requires at least one dimension: style. There are three dimensions, maximum, style, occasion and emotional status. Style must have one of three values: modern, classic and casual, occasion can have one of and emotional status can have two values: sad and happy. Because there are less variables in question and easy to remind, we decided
to incorporate here a voice recognition API: Web Speech API in HTML5 and chosen to recognise two languages: portuguese and english. Before we decided to use this one, we tested other ones but after deeply analysing and performing tests, we decided to incorporate this one.

With the help of this API, we are able to know the text from the user’ speech. The key piece here was building an algorithm that would compare the text from the user speech to the available options. But the general algorithm could easily failed if we only allowed the user to say exactly the same words available in the checkboxes. Because of that, we build three different knowledge bases to each dimension: style, occasion and emotional status. In each one of them, to each option, we searched and stored synonyms in english and portuguese. Therefore, whenever a user presses the microphone button and speaks, our application will search if those words are in our knowledge bases. If there are in the knowledge bases, it will associate to each dimension the value of that word.

1. Style knowledge base:

   - Modern: modern, trendy, moderno, atual, actual
   - Classic: classic, classique, beto, clássico
   - Casual: casual

20https://w3c.github.io/speech-api/speechapi.html
2. Occasion knowledge base:

- Workout: workout, gym, treino, ginásio;
- Date: date, encontro;
- Wedding: wedding, casamento;
- Interview: interview, entrevista;
- Cocktail: cocktail, cocktail party, party, festa;
- Nigh out: nightout, night out, night, saída, noite;
- Daily: daily, daily basis, daily routine, routine, rotina, rotina diária, dia a dia;

3. Emotional status knowledge base:

- Happy: happy, delighted, pleased, glad, joy, lucky, fortunate, afortunado, feliz, contente, satisfeito, alegre, expansivo, jubiloso, sortudo, festivo, risonho, realizado, radiante, animado;
- Sad: sad, unhappy, miserable, unfortunate, unlucky, triste, insatisfeito, descontente, infeliz, miserável, taciturno, sorumbático, desafortunado;

We can not guarantee their full accuracy because we are not responsible for the API that is why we kept the checkboxes. Therefore, the user will only use the tool if he/she wants and feels comfortable and in case the tool does not work at first try. We give the user the power to decide what he/she wants to do or what is best for him/her.

4.8 Interface

As previously mentioned, we developed a recommendation system within an application. Because of that, there was a need to take decisions and analyse the best way to develop an interface that was easy to use even though our main focus was not in the interface application itself.

We developed our interface iteratively and incrementally beginning with developing low fidelity prototypes. Our low fidelity prototypes consisted of sketches on paper of the design of our application. It was a quick and easy way to define which screens needed to be drawn, which ideas should be implemented and how to represent them. They work as a draft not as a final work and we can always make changes to them. We needed two iterations to refine our prototype and achieve our final application look and feel [21]. If we skipped this phase, we would waste a lot more time because we would code everything and then change it again. That is why we designed PBF’s and will discuss our main decisions resulting from those designs. If our main focus in the dissertation was the interface, we could have done an heuristic evaluation through our low fidelity prototypes.
After that initial phase, we started developing our functional prototype. It was only after implementing every functionality that we performed an heuristic evaluation and usability tests to test our functional prototype.

Now, we will analyse the main decisions regarding the design of our interface and the evolution of our prototypes revealing the differences and similarities between the low fidelity and functional prototypes.

We wanted to display a menu with our options. For that we decided to incorporate a navigation bar but it would be crucial to revise the number of options to display on it. Mainly because we are developing a mobile application meaning that we have limited space horizontally and vertically. As we can see in figure 4.48, our original idea was to have a vertical navigation bar but we later decided to change it to horizontal. The main reasons were:

![Navigation bar in first low fidelity prototype](image)

**Figure 4.48:** Navigation bar in first low fidelity prototype

- In our culture, we read things in the left to right flow.
- There are well-known applications such as Facebook and Instagram that use an horizontal navigation bar, then we decided that it would be more easier to understand if we also considered that logic because the users are more familiar with.

It was challenging to decide which options to represent but we decided that they should show our main features:

- **Closet:** in which the user has the possibility to not only add his/her clothes but organise them. This screen has two sub-menu represented by two tabs where the user can organise the wardrobe and add more pieces.
Organise: the user can visualise his/her own pieces. We divided this tab according to the characteristics that the user specifies the piece: type of piece, season, occasions and patterns. For example, whenever he/she clicks on a specific season, it will show all the pieces that the user specified that he/she would use in that season.

In the first low-fidelity prototype, we considered the information visualisation module but, because of the reasons explained in the information visualisation module section, in our second low fidelity prototype we took that into account and designed another screen. In the functional prototype, we decided to represent the data in a different way through slickers instead of having to click on a general option - season, occasion, pattern - and then have a new screen with the sub-options.

We also add the possibility to create albums in which he/she can add not only items but also outfits retrieved by the recommendation tool.

Add: the user adds a piece into the application taking into account its characteristics in order for them to be stored in the closet and considered in the recommendation tool.

The general idea for the design of this screen was the same in each phase of the development, shown in figure 4.50.

Recommender: the user can specify the options to be able to receive a recommendation.

Our initial idea of representing the data of this screen was maintained throughout the development as we can see in the figure 4.51.
Figure 4.51: Recommender screen evolution

- **Feed**: in which the user can see the items or outfits updates of the people that he/she follows. It was an idea that was thought about after developing the low fidelity prototypes.

- **Users**: in which he/she can search for users and see their profiles: user name and albums. It was an idea that was thought about after developing the low fidelity prototypes.

To better understand our interface application we created a user manual that is available in the appendix.

The low fidelity prototypes and its storyboards and the functional prototypes are presented in more detail in the appendix.
5 Evaluation

Contents

5.1 Heuristic evaluation ........................................... 69
5.2 Usability tests ...................................................... 70
In our thesis, we developed an application integrated with a recommendation system that took into account different aspects such as the weather and temperature of a specific location, emotional status, occasion and style.

The fact that we had an application it meant that we needed to develop an easy to use interface and we had the need to evaluate if it was in fact easy to use. The first evaluation to be performed was an heuristic evaluation on our functional prototype done by three experts which goal was to identify which violations our system had regarding usability principles. After analysing the results of that test, we needed to overcome all the problems identified.

When all the problems were solved, usability tests and system usability scale tests were needed to be performed. In this phase, we needed to evaluate our application in the user context with real users. In this evaluation, the user will perform a set of tasks that represent the main tasks possible to perform in our application. The output of this evaluation is being able to know if the application is in fact easy to use and also understand the user experience.

Regarding our recommendation system, as specified before, it took into account different aspects. Aspects such as the temperature, weather of a specific location and occasion were inferred by the user annotation so there was no need to evaluate those aspects.

Emotional status was a variant that was different given the fact that we did not had any information about the connection between emotional status and clothing as discussed before. Because of that, we performed an evaluation with more than 100 users to be able to infer how those fields were connected. We came to the conclusion that when users feel sad they rather use clothes with darker colors and when they feel happy they use clothes with lighter colors. Given the fact that we initially performed this evaluation and implemented those results in our recommendation algorithm that meant that what we implemented was in fact depending on those results so we did not needed to, in the end, perform an evaluation regarding this aspect because it was already done.

We also consider the style. This is an aspect crucial in our recommendation system, so important that we needed to separate from the other aspects as explained in detail in section 4.4.2. We needed to implement a model that took into account the different characteristics of a piece of clothing to map it into a style. The fact that a style is not only given by a piece but also by the combination of pieces, we needed to also implement a combination algorithm that followed certain rules depending on each style. In the end, just for this feature, we had two algorithms: model and combination that were performed in different phases. To be able to synthesise many aspects into a style algorithm, we needed to do a research on the fashion field. But, because we are not fashion experts and because it is hard to turn a creative process of creating outfits into a computational process, we needed to perform a study. Our study goal was to determine if our style algorithm produced good results and if the users were able to
distinguish between styles. For that, we created a form using GoogleForms\footnote{https://www.google.com/forms/about/} in which the users will indicate which style that outfit had.

We also had other two features in our application: social network and voice recommendation. Regarding social network, we analysed the existent models that are used nowadays and that we analysed previously and decided to implement the one that meet our application reality. There was no need to evaluate it because it was a decision. Voice recommendation is not the focus of our thesis but before we incorporate it, we tested the voice recognition ourselves and we test it in a scenario given in the usability tests because it is included in the set of tasks performed by the user.

Briefly, in this chapter, we will address the different evaluations that were performed, heuristic evaluation, usability tests, and those results.

## 5.1 Heuristic evaluation

Given the fact that we developed an whole application with an user interface, we decided that we should perform an heuristic evaluation following the Jackob Nielsen heuristics \cite{Nielsen1994}. Needing a small set of evaluators it was the simplest and quickest way to acknowledge our violations regarding the usability principles.

Our procedure consisted in a heuristic evaluation session with three experts in which it was given full control and freedom to them to evaluate the system. We did not set a precise amount of time so they could explore our interface fully, explore different situations several times and give us their feedback written and verbally.

After that session, it was also performed a debriefing session in which all the evaluators were present and discussed with Margarida the issues identified, possible solutions to them and some advices regarding the general design.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
Severity & Frequency \\
\hline
0 & 0 \\
1 & 2 \\
2 & 23 \\
3 & 5 \\
4 & 0 \\
\hline
\end{tabular}
\caption{Frequency given a specific severity}
\end{table}

In the 30 problems detected by three experts, table 5.1, 18 regarded the same situation. Most of them had a severity of 2. 8 of them regarded error prevention, 6 were about recognition rather than recall and 6 about user control and freedom violations, as seen in table 5.2. The ones identified with higher severity were the first ones to be revised were the following:
Table 5.2: Heuristic evaluation by severity and frequency

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility of system status</td>
<td>5</td>
</tr>
<tr>
<td>Match between system and the real world</td>
<td>0</td>
</tr>
<tr>
<td>User control and freedom</td>
<td>6</td>
</tr>
<tr>
<td>Consistency and standards</td>
<td>3</td>
</tr>
<tr>
<td>Error prevention</td>
<td>8</td>
</tr>
<tr>
<td>Recognition rather than recall</td>
<td>6</td>
</tr>
<tr>
<td>Flexibility and efficiency of use</td>
<td>3</td>
</tr>
<tr>
<td>Aesthetic and minimalist design</td>
<td>0</td>
</tr>
<tr>
<td>Help users recognise, diagnose, and recover from errors</td>
<td>3</td>
</tr>
<tr>
<td>Help and documentation</td>
<td>0</td>
</tr>
</tbody>
</table>

- Problem: The interface does not allow the user to log out;
  Solution: Present a log out button;

- Problem: When the login fails, it does not present which field was wrong.
  Solution: Present a proper information according to the specific error.

- Problem: When the register process fails, it does not present which field was wrong.
  Solution: Present a proper information according to the specific error as shown in 5.1.

- Problem: Navbar does not present the actual icon option of menu where the user is at the moment.
  Solution: Highlight or change the color of the actual icon option of menu.

- Problem: Knowing which is the color selected.
  Solution: Show the option selected by highlighting it as shown in 5.2.

After analysing the results of those sessions, we took into consideration all of the problems identified and implemented a solution in our iterative development process in order to develop a UI with higher quality. First, we solved the problems with higher severity and then the others with lower severity.

5.2 Usability tests

After doing the development and performing an informal evaluation, usability tests were performed with 17 users in order to gather quantitative and qualitative usability metrics. With this evaluation, we aimed to obtain an overall opinion about the user experience with our application.

This phase is one of the most important ones because it is able to provide us the pros and cons of our application in terms of usability and we can improve it after performing it.
To have a good base of analysis, this phase must be performed by, at least, fifteen users which can be of any gender, age and professional education in order to include a wide range of people with different background. [21]

Before performing the test, it was explained as written in the script given to the user, that we are only testing the application and not the user because we want to understand how the users work with it. It was also explained the main goals of the application and the features and functionalities available, clarifying that it will be recorded metrics on the number of errors made and on time duration.

While performing this evaluation we assured that the conditions given were the same to all the users.

The evaluation was performed as followed:

- Before presenting the users with the evaluation part, it was given the script in which was available an introduction of the test and explained verbally the main goals of this application and which functionalities where available.
• As previously mentioned, a script was given and it had a set of thirteen questions that were wisely selected to evaluate different levels of complexity and cover all the application’ features.

• Before performing the test itself, a five time period was given to the users to explore the application.

• The tasks were performed always following the same order as presented in the script, and in the developer smartphone.

The set of tasks was:

1. Sign-up in the platform with the following data:
   Username: John Doe(male) or Jane Doe(female)
   E-mail address: johndoe@gmail.com
   or
   janedoe@gmail.com
   Password: 12345Gmail

2. Log in the platform
   Username: mags
   Password: 123456
3. Describe an item and save it in the application.
   Piece: Bottom - Jeans
   Occasion: Date, interview, night out and daily basis
   Season: Summer, spring, autumn and winter
   Genre: Female
   Weather: Clear, rain, cloudy, fog and partly cloudy
   Temperature: 10°C to 15°C and 15°C to 20°C
   Pattern: none
   Fabric: Denim

4. Make a recommendation using the following directions:
   Day: 17th of August
   Occasion: daily basis

5. Make a recommendation using the following directions:
   Location: here
   Day: 17th of August
   Occasion: daily basis
   Emotional status: happy

6. Make a recommendation using the following directions:
   Location: here
   Day: 17th of August
   Occasion: daily basis
   Emotional status: happy
   Style: casual

7. Make a recommendation by voice using the following directions:
   Location: here
   Day: 17th of August (selected)
   Occasion: daily basis (voice)
   Emotional status: happy (voice)
   Style: casual (voice)
Table 5.3: Duration usability tests results

<table>
<thead>
<tr>
<th>User</th>
<th>Q.1</th>
<th>Q.2</th>
<th>Q.3</th>
<th>Q.4</th>
<th>Q.5</th>
<th>Q.6</th>
<th>Q.7</th>
<th>Q.8</th>
<th>Q.9</th>
<th>Q.10</th>
<th>Q.11</th>
<th>Q.12</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.1</td>
<td>26.91</td>
<td>11.9</td>
<td>50.55</td>
<td>20.07</td>
<td>18.67</td>
<td>14.75</td>
<td>12.33</td>
<td>9.22</td>
<td>5.18</td>
<td>12.06</td>
<td>10.5</td>
<td>11.07</td>
</tr>
<tr>
<td>U.2</td>
<td>31.53</td>
<td>15.35</td>
<td>57.94</td>
<td>32.56</td>
<td>29.58</td>
<td>18.98</td>
<td>19.44</td>
<td>11.38</td>
<td>8.62</td>
<td>13.64</td>
<td>15.87</td>
<td>12.02</td>
</tr>
<tr>
<td>U.3</td>
<td>52.76</td>
<td>10.45</td>
<td>61.64</td>
<td>11.06</td>
<td>21.83</td>
<td>25.64</td>
<td>23.12</td>
<td>9.49</td>
<td>7.75</td>
<td>11.95</td>
<td>13.23</td>
<td>11.54</td>
</tr>
<tr>
<td>U.5</td>
<td>60.02</td>
<td>15.46</td>
<td>53.53</td>
<td>40.63</td>
<td>30.02</td>
<td>28.77</td>
<td>26.31</td>
<td>15.54</td>
<td>9.53</td>
<td>14.95</td>
<td>18.43</td>
<td>17.82</td>
</tr>
<tr>
<td>U.6</td>
<td>54.73</td>
<td>16.12</td>
<td>56.78</td>
<td>35.84</td>
<td>27.67</td>
<td>25.28</td>
<td>22.26</td>
<td>13.57</td>
<td>8.65</td>
<td>14.43</td>
<td>15.29</td>
<td>14.38</td>
</tr>
<tr>
<td>U.8</td>
<td>52.47</td>
<td>18.31</td>
<td>57.96</td>
<td>35.92</td>
<td>24.71</td>
<td>22.65</td>
<td>21.79</td>
<td>14.06</td>
<td>7.06</td>
<td>13.74</td>
<td>12.67</td>
<td>11.63</td>
</tr>
<tr>
<td>U.9</td>
<td>53.95</td>
<td>16.88</td>
<td>58.29</td>
<td>31.73</td>
<td>26.64</td>
<td>23.96</td>
<td>24.89</td>
<td>15.25</td>
<td>8.25</td>
<td>15.28</td>
<td>14.57</td>
<td>14.83</td>
</tr>
<tr>
<td>U.10</td>
<td>45.63</td>
<td>13.25</td>
<td>52.76</td>
<td>37.22</td>
<td>19.99</td>
<td>17.02</td>
<td>19.14</td>
<td>13.09</td>
<td>6.23</td>
<td>12.79</td>
<td>15.56</td>
<td>15.07</td>
</tr>
<tr>
<td>U.11</td>
<td>65.51</td>
<td>20.02</td>
<td>60.02</td>
<td>45.56</td>
<td>36.89</td>
<td>29.91</td>
<td>29.89</td>
<td>15.95</td>
<td>9.88</td>
<td>19.28</td>
<td>17.73</td>
<td>16.93</td>
</tr>
<tr>
<td>U.12</td>
<td>54.01</td>
<td>19.69</td>
<td>59.85</td>
<td>38.57</td>
<td>32.67</td>
<td>28.53</td>
<td>24.19</td>
<td>15.21</td>
<td>8.05</td>
<td>16.85</td>
<td>16.02</td>
<td>15.94</td>
</tr>
<tr>
<td>U.13</td>
<td>57.82</td>
<td>17.22</td>
<td>56.65</td>
<td>39.74</td>
<td>25.91</td>
<td>22.43</td>
<td>21.59</td>
<td>15.37</td>
<td>7.42</td>
<td>15.93</td>
<td>15.39</td>
<td>15.76</td>
</tr>
<tr>
<td>U.16</td>
<td>56.73</td>
<td>18.53</td>
<td>52.87</td>
<td>33.85</td>
<td>25.72</td>
<td>21.75</td>
<td>23.08</td>
<td>15.63</td>
<td>8.95</td>
<td>15.85</td>
<td>13.76</td>
<td>13.28</td>
</tr>
<tr>
<td>U.17</td>
<td>61.36</td>
<td>19.41</td>
<td>64.26</td>
<td>40.02</td>
<td>35.01</td>
<td>27.52</td>
<td>27.26</td>
<td>15.3</td>
<td>9.67</td>
<td>20.48</td>
<td>19.02</td>
<td>18.56</td>
</tr>
</tbody>
</table>

- **min**: 26.91, 10.45, 50.55, 11.06, 17.74, 14.75, 12.33, 9.22, 5.18, 12.06, 10.5, 11.07
- **max**: 65.51, 20.02, 64.26, 45.56, 36.89, 29.91, 29.89, 15.95, 9.88, 20.48, 19.02, 18.56
- **mean**: 51.44, 16.29, 56.35, 33.15, 26.68, 23.62, 22.6, 13.52, 7.86, 15.02, 14.98, 14.44
- **s.dev.**: 9.8, 2.92, 3.72, 8.54, 5.14, 4.38, 3.79, 2.22, 1.4, 2.29, 2.17, 2.13
- **c.int 95%**: 4.6, 0.04, 1.77, 4.06, 0.08, 0.06, 0.06, 0.03, 0.02, 0.03, 0.03, 0.03

8. Write the number of existing patterns.
9. Go to the closet and write the number of stripped clothes.
10. Write the number of existing albums.
11. Create an album named “Teste1”.
12. Search for Sarah’s profile and follow her.

After performing the questions given on the script, it was performed a quick but accurate usability questionnaire with 10 questions using the system usability scale (SUS) invented by John Brooke [3]. Because the SUS is a likert scale question, each question was ranked from 1 – strongly disagree – to 5 – strongly agree.

### 5.2.1 Results

#### 5.2.1.A Usability tests

We decided to append all the information regarding the usability tests- time duration and number of errors in just two tables so the information is well organised in tables 5.3 and 5.4.
Table 5.4: Number of errors on usability tests

<table>
<thead>
<tr>
<th>User</th>
<th>Q. 1</th>
<th>Q. 2</th>
<th>Q. 3</th>
<th>Q. 4</th>
<th>Q. 5</th>
<th>Q. 6</th>
<th>Q. 7</th>
<th>Q. 8</th>
<th>Q. 9</th>
<th>Q.10</th>
<th>Q.11</th>
<th>Q.12</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.11</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>U.12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.13</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.15</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U.17</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

min 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
max 2 1 0 2 2 1 2 0 0 0 0 1
mean 0.47 0.06 0.65 0.29 0.11 0.18 0 0 0 0 0 0 0.06
s.dev. 0.7 0.24 0.84 0.57 0.32 0.51 0 0 0 0 0 0 0.24
c.int 95% 0.33 0.11 0.4 0.27 0.15 0.24 0 0 0 0 0 0 0.11

While performing usability tests there were registered 31 errors, as shown 5.4. These errors were made mainly because this is a new application and the users had some doubts that could not be clarified during the evaluation process. Nevertheless, they were able to recover from those errors and perform the intended action. By analysing, we can easily see that for question the maximum number of errors done were two out of the 17 testers. 58.8 per cent made at least one error during their usability test. Next, we will analyse in more detail those results.

The questions with less mean time, question 8 and 9, were the ones less complex not only in perception but also in needed data to be inputed. In questions 8, 9 and 10 the user only needed to manage where they could see the answer to the question asked in the script.

Question 11 was simple and easy to perform given the fact that by performing the previous ones the user would already know where the user could add an album that is the reason why this question had a low mean duration time value.

Question 1 was the second one not only in the number of users that made a mistake but also on the mean time value. This is due to the typos introduced by the user because they did not confirm if the data introduced was correct before submitting it.

The question with higher maximum and mean time values was the question 3. This is comprehensible given the fact that this was the question in which the user needed to describe an item according to the 8
characteristics given on the script. There was a lot of information to be inputed which is of course time consuming. Even though that when there is more data to be introduced, there is a higher probability for the user to error that did not happen because no users made mistakes because they took more time to perform the action and that reflected on their response time. They learnt from the errors they made in the first question and certified the information introduced before submitting it.

The questions related to the recommendation itself were the most complex because the user did not totally understood that for a recommendation to be produced one did not needed to introduce all options. There is why 7 users in the first task made an error, in the second one just 43 percent of the users from the previous question made errors and in the last those users did not perform an error but there were 2 new users to make a mistake.

Overall the results made us realise that the errors made during the evaluation process were not major. The fact that the set of questions with higher mean and maximum values were the ones regarding the recommendation is due to the fact that a user can have a recommendatio based on just three variables (date, location and style). To clarify that, we decided that we should provide a tooltip so the user can read the instructions and know of this condition.

5.2.1.B SUS

We consolidated all the information regarding the answers to SUS test in the table 5.5. We followed the guides established by J.Brooke [3] and applied the rules to obtain our SUS application score:

<table>
<thead>
<tr>
<th>User</th>
<th>Q.1</th>
<th>Q.2</th>
<th>Q.3</th>
<th>Q.4</th>
<th>Q.5</th>
<th>Q.6</th>
<th>Q.7</th>
<th>Q.8</th>
<th>Q.9</th>
<th>Q.10</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>92.5</td>
</tr>
<tr>
<td>U.2</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>97.5</td>
</tr>
<tr>
<td>U.3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>87.5</td>
</tr>
<tr>
<td>U.4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>U.5</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>U.6</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>90</td>
</tr>
<tr>
<td>U.7</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>92.5</td>
</tr>
<tr>
<td>U.8</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>U.9</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>90</td>
</tr>
<tr>
<td>U.10</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>92.5</td>
</tr>
<tr>
<td>U.11</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>67.5</td>
</tr>
<tr>
<td>U.12</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>92.5</td>
</tr>
<tr>
<td>U.13</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>97.5</td>
</tr>
<tr>
<td>U.14</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>95</td>
</tr>
<tr>
<td>U.15</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>90</td>
</tr>
<tr>
<td>U.16</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>92.5</td>
</tr>
<tr>
<td>U.17</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>95</td>
</tr>
</tbody>
</table>
• the odd number questions results were subtracted by one
• the even number questions results were subtracted by their value from 5
• at the end, all the values were summed and multiplied by 2.5 to achieve the result in 100.

The minimal rating given was 67.5 and the maximum was 97.5 on a 100 points scale. After doing the mean of all the values obtained, our application achieved a 91.61 score as we can see in figure 5.3, which corresponds to an A in an A-F grade scale (A is the best and F the less).

![Figure 5.3: Application SUS result](image)
Conclusions and future work
Developing an entire system is hard and time-consuming. We developed a complete solution that included not only the recommendation algorithm but the combination algorithm, the social network, the voice engine and the whole application development. We also gave attention to the interface itself with the purpose to provide an application easy to use. Even though, it was designed and developed iteratively and incrementally and it is in its final version and if we wanted we could provide it as a final product and market it, there are obviously improvements that could be done.

Regarding the interface itself, it is in a final version being corrected after the heuristic evaluation and usability tests. We also developed a social network which main goal is to obtain inspiration thus making sense that it would be a platform with followers, not friends. The searching tool for users profiles is well defined, showing the username and his/her albums. The feed functionality shows whenever a user adds a piece or an outfit to an album or creates an album. The improvement that could be done would be giving the possibility to his/her followers to like a post. That improvement could interconnect with our recommendation and combination algorithm.

Nowadays, our recommendation and combination algorithm considers all the aspects that it should: temperature and weather of a specific location, emotional status, occasion and style. As explained before, our algorithm is divided into two parts: the first one being in the annotation phase and the other is also processed in two parts being the first one the result from performing queries in a user knowledge base and the other one the combination of pieces itself considering the chosen style.

We consider three styles but our implementation is open to receive more styles, it is only needed to analyse the ones to be added into our system and add those characteristics and rules in the annotation phase and combination.

One thing that we also considered was the user feedback. Whenever a user receives a recommendation, he/she has the possibility to store it in one of his/her albums and we store the outfits that he/she disliked not retrieving them again. But we could do that to the likes meaning that we would have to analyse the outfits that he/she liked and implement an algorithm able to retrieve suggestions similar to the ones that were liked. We implemented the means to be possible to do that: user storing the outfit itself in an album but we did not integrate the likes into our algorithm given the fact that we had already a lot of variables to consider. With that in mind, we could also take advantage of the improvement of liking other users’ posts to consider in his/her past behaviour. In this case, we would be using explicit feedback because we consider his/her likes and the previous recommendations that he/she stored into his/her own albums to provide the recommendation. To go even further, we could use hybrid feedback if we integrated the implicit feedback and considered monitoring the user activity - implicit feedback - that in this case would be the time he/she passed looking to other users posts: we could track which post was shown at that moment and the time spent and analyse it, incorporating the results of that analysis and the previous explicit feedback into his/her user profile.
Another improvement would be including clothing items from different online shops in the recommendation tool. If the user liked a certain item, the application would redirect him/her to the website and the user could buy it.

To improve our algorithm regarding its efficiency, whenever a recommendation is retrieved, we could store it an a database with the inputed conditions. Meaning that in a next recommendation, the system would analyse that database and see if those conditions were met in the past, it would retrieve the recommendations stored in the database. Going even further, it could be developed a similarity function that would analyse the inputed conditions and see the similarity between the ones stored in the database and if that similarity was higher than a certain value, it would adapt the case stored in the database to the actual one.

Regarding the voice engine, it is developed according to its three different corpus giving the possibility to only consider style or style and occasion or style and emotional status or style and emotional status and occasion as it is in the checkboxes. It would not make sense to incorporate a voice engine in the annotation phase but only in the recommendation. An improvement will only be needed if it was added any style, occasion or emotional status. In that case, we should revise our corpus to add this new option and his synonyms.

The fact that we developed a web-app with a view opens the possibility to use our application not only on smartphones but in tablets and computers transforming our application into a cross-platform one. The fact that our main target in our application is the smartphones was the main reason that we did not implemented the information visualization graphics presented in the low fidelity prototypes. If our application target would also be computers we could add this infovis module in which we could easily visualise the clothing items. As discussed in introduction chapter, users even if they do not own a lot of items in their wardrobe, they cannot remember all of them. That makes it hard for them to have an overview analysis of which pieces they have the most and which they need. Regarding that matter, if it the target devices were personal computers and devices, it would be advantageous to have the views that we will show next because they would be able to provide that kind of information.

The idea was to have two views. One regarding the pieces of clothing and other one regarding the colours.

In the first view, as we can see in figure 6.1, there would be a representation of a body. This view would work as following: a user could click in two parts: top or bottom and the system would retrieve the pieces according to the option selected. It would also display the number of pieces that were from that type.

In the second view, figure 6.2 it would have been presented a SunBurst with a hierarchy of colors in which the user could select the colour he/she wanted and the system would retrieve the pieces according to the color selected. It would also display the number of pieces of that color.
Another feature could have been included as an interactive mechanism. In this mechanism, the user could select a certain type of piece and a color and it would filter by those options and show the results of intersecting this restrictions.

Regarding the evaluation, even though we evaluated our application concerning its usability, it was not possible to evaluate our recommender system properly to evaluate if it was indeed accurate in what concerns suggestions to a certain user. A future work would be performing an evaluation with around 15 users in which they would use the application in a two-week period, photograph their clothes and get suggestions. With this evaluation, we could validate if our suggestions were suitable for each user considering the different recommendation variables: weather and temperature of a certain location, n occasion, style and emotional status.
Bibliography


[21] Lewis, C. Task-centered User Interface Design:


83


NEILSEN, J., AND MOLICH, R. ACM CHI’90 Conf. (1990), 249—-256.


Appendix

Contents

A.1 User Manual ................................................. 87
A.2 Low fidelity prototypes ....................................... 93
A.3 Emotional status questionnaire ............................. 93
A.1 User Manual

The following figure shows our initial application page where the user can login to access it providing his/her username and password.

**Step 1:** If the user is not already registered, he/she can click the “Register Now!” A.1.

![Figure A.1: Login page](image)

**Step 2:** In figure A.2, where he/she can fill the username, e-mail, password fields to create an account and use InsideFashion. Never forget that the “Password” and “Confirm Password” fields must be equal.

![Figure A.2: Sign up page](image)

In case that the user forgets the password:

**Step 1:** he/she can click the “Recover Password” in figure A.1 and goes to A.3.

**Step 2:** ow he/she can put his/her e-mail and it will be sent an e-mail to that address with a new password and it will have access to the app again.
A.1.1 Closet

Introduction

This functionality allows us to create a virtual closet where the user can add items, organise them in albums according to his/her preferences and visualising item clothes according to types, seasons, occasions and patterns.

Add an item to the closet:

To add an item, user should photograph it and characterise it according to the item characteristics: type of piece, pattern and fabric. He/she should choose the occasion, season, weather and temperature in which he/she would wear it. The item characteristics fields must be filled with just an option and the personal characteristics that depend on the user, he/she can choose all the options that apply.

Step 1: click on the “Take a photograph” button and photograph the item.
Step 2: To fill the type of piece, occasion, season, weather and temperature, click on each name and it will appear a pop-up where user can choose it.

Step 3: after filling all the available fields, click on “Save”.

Step 4: after clicking on “save”, it will appear a pop-up with a box of colors in which the user should choose the background/main color of the item and click on “Ok”, as seen in A.5.

View an item

To view an item, the user can search for the specific characteristic of the item which can be: season, pattern, type or occasion.

Step 1: Choose the characteristic option, then click on it, A.6. For example, the bottom items in which the characteristic chosen was the type of piece and the value was bottom.
Step 2: It appears all the items that have that characteristic option. In this example, it appears all the bottom items, A.7.

Create an album

To create an album, the user should choose a name to it.

Step 1: Click on “add an album”, as seen in A.8.

Step 2: Write the name of the album.

Step 3: The created album appears now in the albums.

View an album

To view an album, the user should choose the selected album.

Step 1: Click on the album’s name.
**Step 2:** It shows all the items – photograph and characteristics- saved in that album.

**Add an item to an album**

To add an item, the user should first go to “view an item” and then do the following procedures.

**Step 1:** Click on the plus icon.

**Step 2:** Select the album where you want to add the item or create a new album where it can be added.

### A.1.2 Recommendation

This functionality allows the users to get a recommendation based on selected:

- style or
- style and occasion
- style and emotional status
- style and occasion and emotional status

and on the temperature and weather on the selected dated of the chosen location.

It is also possible to get the recommendation through voice, as seen in A.9.

![Recommendation page](image)

**Figure A.9:** Recommendation page

The user should choose a date and the location. After that, he/she can rather select the style, occasion, emotional status or do it by voice.

**Step 1:** Click on the map to choose the location.

**Step 2:** Click on the calendar to choose the day.

**Step 3:** Click on style and choose it.

Optional:

**Step 4:** Click on emotional status.

**Step 5:** Click on occasion.
Click on the microphone icon and say the intended style and emotional status and occasion. Note that the style option is always required, the occasion and emotional status are both optional.

A.1.3 Social Network

This functionality allows the users to have their own network in which they can follow other users and view their items on their feed and add them to their own albums.

The user can search for a friend, as seen in A.10.

**Step 1:** click on the search field.

**Step 2:** Type his/her name or username.

**Follow a friend**

The user can follow another user after searching.

**Step 1:** choose an user and click on the follow button, as seen in A.10.

![Figure A.10: Social network](image)

**View friend’s album**

To view the albums of a friend, she/he should choose a specific friend.

**Step 1:** choose a friend of the ones that you follow, and swipe through his/her albums.

**Step 2:** Choose a specific album and the items will appear in a pop-up.

**View feed**

All the items that are added to other users albums are showed in the feed of the users that follow him/her.

Click on the feed icon, as seen in A.11.

**Disclaimer:** because we tested our application with a pre-set of images we do not have deployed a version with a “Take a photograph” in the option “Add item”. Instead, we have the items’ images to annotate.
A.2 Low fidelity prototypes

In this section, we present our low fidelity prototypes regarding our interface.

![Feed page](image)

Figure A.12: Tagger system low fidelity prototype

A.3 Emotional status questionnaire

In the following section, we present screenshots of our webapp developed within the scope of the emotional status study.
Figure A.13: Application low fidelity prototype

Figure A.14: Application storyboard
Figure A.15: InfoVis module low fidelity prototype

Figure A.16: Explanation of the purpose of the study

Figure A.17: Basic information needed
**Figure A.21:** Fourth question with a pop-up images exemplifying the selected style, in this case: classic

**Figure A.22:** Fifth question

**Figure A.23:** Sixth question
**Figure A.24:** Seventh question

**Figure A.25:** Eighth question with tooltip images exemplifying the selected pattern, in this case floral.

**Figure A.26:** Ninth question
Figure A.27: Tenth question

Figure A.28: Eleventh and twelfth question

Figure A.29: Thirteenth question
**Figure A.30:** Fourteenth question

**Figure A.31:** Fifteenth question