Aalborg University

Multimedia Recommendations

P3-Project

Group ds305e12:

Frederik Daniel Bank Gundersen

Frederik Meyer Bønneland

Michael Jensen

Naburan Amirthalingam

Peter Gurnæs Havbro Nielsen

Rasmus Bak Christiansen

AALBORG UNIVERSITET

Det Teknisk-Naturvidenskabelige
Computer Science and Software
Selma Lagerlöfs Vej 300
Telefon 96 35 97 31
Fax 98 13 63 93
http://tnb.aau.dk

Title:
Media Recommendation

Theme:
Programmering og problemløsning

Projectperiod:
P3, 2013

Projectgroup:
ds305e12

Participants:
Frederik Daniel Bank Gundersen
Frederik Meyer Bønneland
Michael Jensen
Naburan Amirthalingam
Peter Gurnæs Havbro Nielsen
Rasmus Bak Christiansen

Advisor:
Emmanouil Valsomatzis

Oplagstal: 10

Sidetal: 60

Bilagsantal og –art: 2 + 1 CD

Afsluttet den: 20. Dec 2013

Synopsis:

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Contents

Chapter 1

Introduction

Entertainment media is a part of almost all people's lives in some shape or form. It is something we seek out and consume almost every single day. Here is things like books, films, video games, music, etc included. For that, there exist various kinds of way to spread a piece of media to the public, like websites, application, advertising, and social groups. Here can a system which generates recommendations based on various kinds of data, like personal information, media data, and social connections, also be included. Either directly or indirectly. This kind of recommendation is tailored towards a certain person, and is there to make them aware that there might be other products which he could be interested in. Like a movie that is similar in genre to what a person previously have watched, or an add-on product to a previous purchase on a retail website.

Recommendation systems have various problems before them hindering its effectiveness, like data shortage. It is also very crucial that proper weights can be generated for the various kinds of data, so the most important aspect of a piece of media is highlighted for the individual person. And interesting feature could be if it were able to generate recommendations across different kinds of media, E.g. if you liked these books, maybe you would like this movie, then it would also require a set of suitable connections between them, to create the recommendation. Since these kinds of systems is centered around the user, it is also important to include some kind of survey to study people's habits regarding media. These surveys could also provide weights for how important certain kinds of data is. These challenges is what creates the base problem for this project.

1.1 Initiating Problem

What challenges exist for recommender systems, what is different peoples habits regarding recommendations from various sources, and to what degree is it possible for recommending media across different kinds of media?

Chapter 2

Problem Analysis

This chapter is going to start the report with the problem analysis, which was carried out during the beginning of the project. The problem analysis covers various fields which were important the projects area, and forms the foundation for the remaining of the project with a problem formulation, definition, and project boundaries.

2.1 Recommender Systems

There are 3 ways users retrieve desired information on the net; searching by browsing, searching by query and recommendation [REF].

When browsing, users go through webpage's sequentially following hyperlinks. It can be useful if you have already found the webpage containing the data you were looking for or the page that has the required hyperlinks. An example is wikipedia, where a lot of the expressions on a page are also hyperlinks to another page further explaining the expression, for anything from a name for a philosopher to a mathematical equation. But the approach is not useful if the page on which the information is located has not yet been found.

A way to locate the correct page on the web is through querying a search engine. But according to [REF] queries through search engines work best when you know the exact and right wording for the item or webpage needed. Too specific problems are polysemy; the existence of multiple meanings for a single word and synonymy; the existence of multiple words with the same meaning. This becomes especially apparent when you get stonewalled in a querying system, which means no results could be returned based on the criteria.

The last option for retrieving the desired information is through recommendations. Recommendation-based systems can suggest items, like products from a catalogue, movies or news articles for the user.

2.1.1 Background

Search tools were developed to counter information overload. It refers to the difficulty a person can have understanding an issue and making decisions that can be caused by the presence of too much information.[REF]

Traditional search engines, return the same result for the same query, regardless of who submitted it.

Personalization, adapting to each user's needs, the goal is to provide users with what they need without them asking for it explicitly.

To make the recommendations the system needs to have a profile on the user. A collection of data about the user that the system uses to calculate the recommendations from. There are two ways for the system to collect that information, either through automatic personalization or customization. When building the profile using customization the user is prompted to fill in all the information needed for the recommendation calculations.

The automatic personalization profiling on the other hand doesn't ask for input but automatically collects user behavior data while the user is browsing the website and it then tries to predict what the user would like based on that. An example of a website using automatic personalization is Amazon. The Amazon website stores information about items the user has clicked on and what items the user previously bought to try and calculate recommendations to the user.

Personalization can be broken into a data collection phase and a learning phase. In the data collection phase user data is being collected and in the learning phase the user profile is created based on the collected data. The process of learning can either happen at the time the recommendations are needed and is called a memory based system or the learning phase can be pre-build so the profile has already been generated at the time it is needed called a model-based system.

The memory based system is not very scalable and will perform badly on a system with many users and items. But making the profile at the time it is needed makes it more adaptable as it is always build on the newest collected data.

The model based system scales better than the memory based system because a big part of the learning phase has already been done offline when the recommendation is requested and will therefore be faster on a system with many users and items.

2.1.2 Business Perspective

From a business standpoint there are several common uses of recommendation systems[REF]:

Product Recommendations

A very common use of recommendation systems today is at online retailers. The websites often collect information using automatic personalization and then compare the user with similar users from which they recommend other items.

Movie Recommendations

Websites like Netflix use a lot of resources on optimizing recommending movies to the user based on the users ratings on watched movies. They even offered a prize for a million dollars to the developer who could beat their recommendation system by 10% and the prize was claimed in 2009 by the team "Bellkor's Pragmatic Chaos".

News Articles

Another use for recommendation systems is recommending news articles or blogs that might be of interest to the reader based on what they have read in the past. The system can identify important words in the documents and from those find similar documents to recommend or it can look at similar users and recommend articles or blogs based on what they have also read.

2.1.3 Recommendation Systems Overview

Collaborative Recommendations

In the collaborative system the data used to generate the recommendations takes into account models made on other users. The theory behind the collaborative systems is that "users with similar interests are likely to find the same resources interesting for similar information needs." [REF]

If you take a small group of people interested in movies who meet each other regularly, eg. at work or similar. They will talk about what new movies they have seen and recommend the ones they liked to the others in the group. After watching some of the different recommended movies each of members

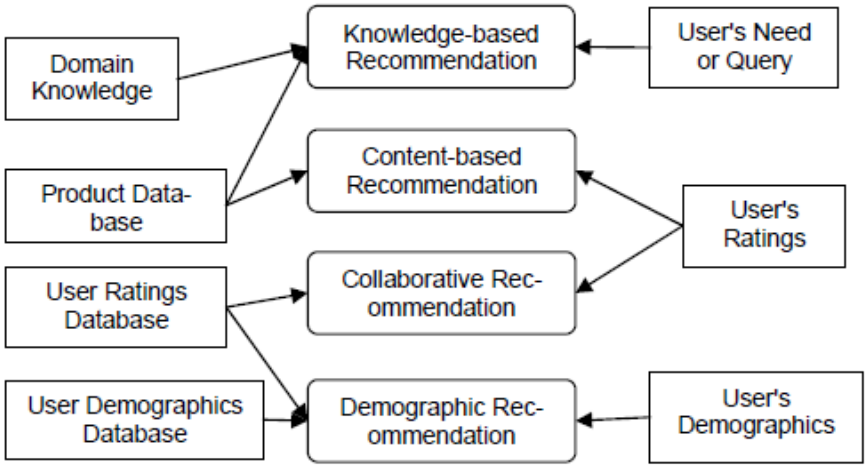

Figure 2.1: Recommendation Techniques and their knowledge sources

of the group will get a sense of who in the group recommend movies they like, who recommend movies they don't like and who recommend a mix.

In short they have identified what members of the group have similar interests so they can get recommendations from them.

What the collaborative system does is take that idea and apply it to a computer system with thousands of people. And with the speed of computers it can be made in real time and with a much higher precision than with only the small group of coworkers.

Content Based Recommendations

In the content-based system the data used to generate the recommendations comes from item descriptions, tags or significant words in a text. The system then builds a user profile of user interests based on that data. Through matching the user profile against other items the system can then make a relevance judgment that represents the users level of interest in the items. [REF]

The Demographic System

In the demographic recommendation system the data used to generate the recommendations comes from demographic data such as age, gender and occupation. The demographic data is used to generate the user profile and it is then possible to recommend items to different demographic niches based on the combined user ratings in those niches. [REF]

The Knowledge-based System

The knowledge-based system is used to solve complex problems and was

first developed by artificial intelligence researchers. The system is most commonly constructed with one or both a knowledge base and an inference engine. The inference engine represents logical assertions and conditions about the world usually as IF-THEN rules.

Early knowledge based systems was used as expert systems. An example was Mycin that was used in medical diagnosis. Later they have been adapted to the internet and is often used to classify objects in complex and unstructured data.

But as the knowledge based system works from assertions and conditions that all have to be programmed the system hits a knowledge engineering bottleneck as the amount of data it has to handle grows. [REF]

Differences

All the recommendation systems have weaknesses and strengths.

If we look at the learning based systems they suffer from cold start problems, which means they have difficulty handling new users and items. They need information about the user to work properly.

Another problem is stability vs. plasticity, once the system has learned a good deal about a user, it has difficulty changing its results. If a user decides to become a vegetarian the food shopping site he uses will keep giving him meat recommendations for a long time before reflecting his new eating habits. To try and contain that problem some developers include temporal discount in their systems, to cause older ratings to have less influence, but by doing that they risk losing information about sporadically exercised behavior. The knowledge based systems isn't learning based and is therefore the only one of the four common systems that doesn't suffer from that problem.

A huge strength for the collaborative and demographic systems is their unique capacity to identify cross-genre niches, they can sometimes recommend an item of interest to the user which is outside the users normal areas of interest. The knowledge-based system can also do this to some degree but only if the associations have been identified ahead of time and programmed into the system.

The hybrid system tries to meet some of those shortcomings by combining two or more of the systems.

2.2 Recommendation Statistics

In a study conducted by Harris Interactive [?] among 2.311 U.S. adults ages 18 and over, on how they use and feel about interaction on the internet. It was conducted between April 28 and 30 2010. They asked different question and some of them are quite interesting. Although it was conducted in America, the information can still be used in our project. The idea behind our project is to create a website that can be accessed from across the world. So different studies in different countries are useful for the information needed to determine the target group. That is why this information is useful for our project.

The first graph, figure 2.2, asks the question about what the asked person shares through social media. They were to select all that applies to them. If we take a look on what the different options was, one stands out and that is: TV and movie recommendations. In the range from 18 to 34 year old, 36 percent of them says that they share TV and movie recommendations on social media.

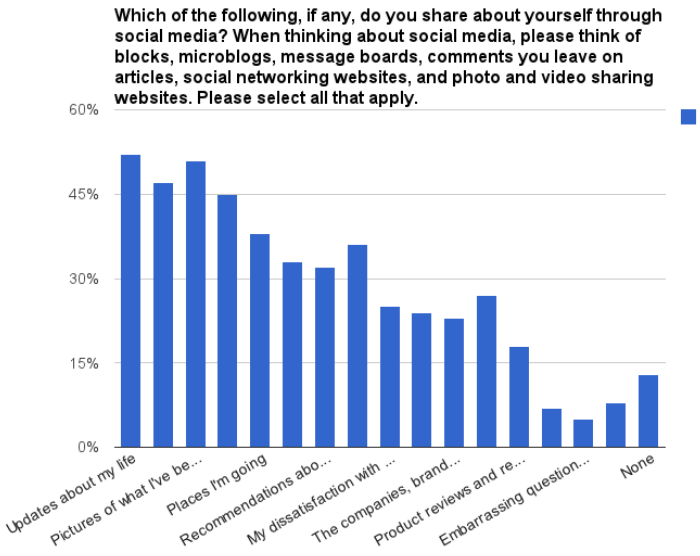

Figure 2.2: 1

The first interesting question and answer is, that in short terms asks what the person share on social media and select all that applies. If we take a look on what the different options were one strikes out and that is; TV and movie recommendations. In the range from 18 to 34 year olds, 36 percent of them says that they share TV and movie recommendations on social media.

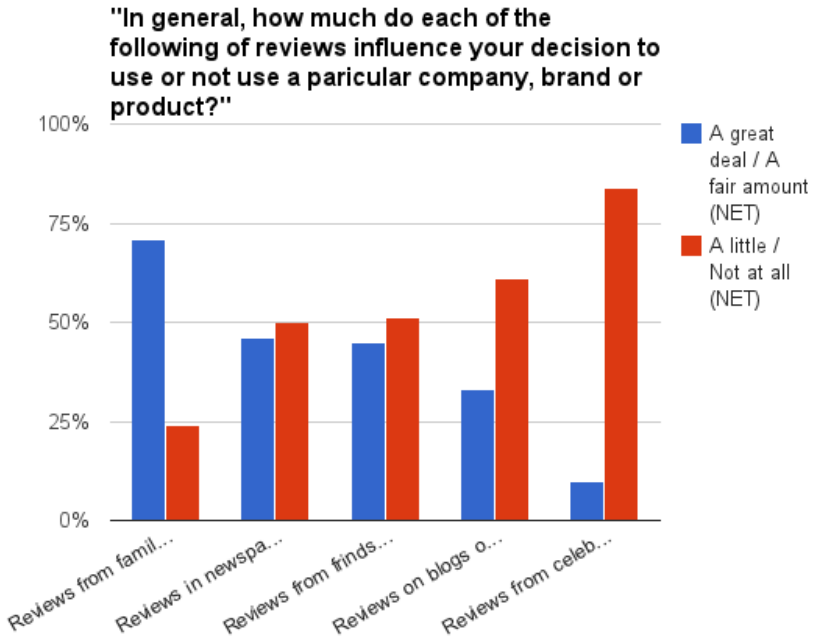

Figure 2.3: 1

The second graph, figure 2.3, asks the question about how much different things influence them. It shows that a staggering 71 percent of those asked, replied that they are influenced a great deal or a fair amount by reviews from family members or friends. That shows that social recommendation is something that a lot of people have in mind when it comes to buy or watch something.

The third and last graph, figure 2.4, shows the age group of those asked the previous question. That graph shows that the 18 to 34 year olds are more influenced by blog and social media, than the older ages were. At the same time, the older people are more influenced by their family and friends.

The third and last graph shows the age group of those asked the previous question. It shows that 18-34 year old are more influenced by blogs and social media than the older ages. While the older people have are more influenced by their family and friends.

2.3 Existing Solutions

According to Alex Iskold, from readwrite.com, there are four approaches for recommendation. The first approach is personalized recommendation which

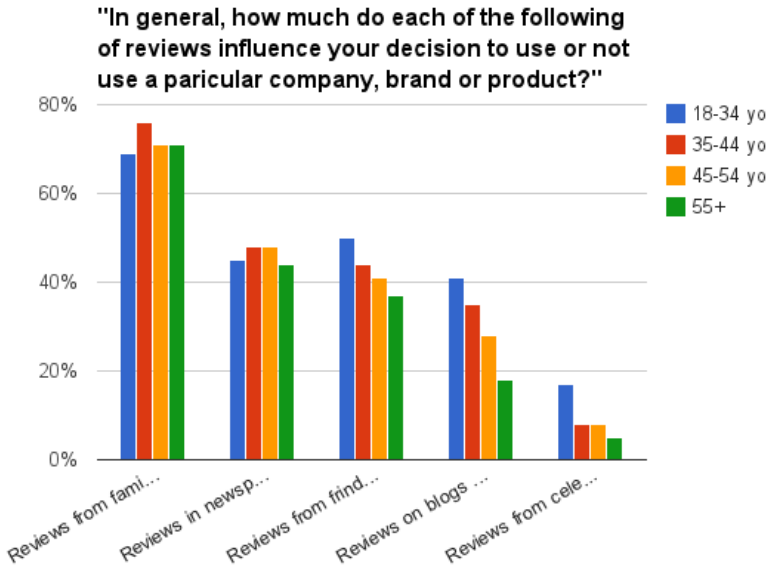

Figure 2.4: 1

is based on the individual's past behavior. The second approach is social recommendation which is based on the past behavior of similar users. The third approach is item recommendation which is based on the item itself. The last approach is a combination of the all three approaches mentioned above. With these four approaches this section will look at solutions on existing recommendation systems. [**?**]

The four approaches are:

- Personalized recommendation - recommend things based on the individual's past behavior.
- Social recommendation - recommend things based on the past behavior of similar users.
- Item recommendation - recommend things based on the item itself.
- A combination of the three approaches.

2.3.1 Solutions

Amazon is the world's largest online retailer. It started as an online bookstore but then it soon got every kind of things like movies, electronics, clothes.

Amazon system uses the combination of the three approaches. Which means the system is based on individual behavior, and either the item itself or behavior of similiar users. Amazon have a "what other customer also bought" under the item you are viewing, which is a social and personalized recommendation. A customer reviews is also possible to be view for the item, where the specific customer is available to rate and comment on the item. And based on the reviews it shows the top 3 similiar statements. Amazon gives you other recommendation based on what your history, which is personalized recommendation

YouTube

YouTube is a popular video sharing website which is under control by Google. YouTube system uses the personalized recommendation. YouTube takes your history, and other activity on the site, to give you recommendations, on which it gives recommendations for channels and videos. So basically whatever video you click on will be placed in your personal history list and based on that it will give you some recommendation.

The problem with YouTube is that even if you only watched one second of a certain video, it will be included in the recommendations to come. Unwanted videos can easily be recommended because of this. Channels also gets recommended even after only watching a single video from the channel.

IMDB

Internet Movie Database(IMDB) is a database with old and new movies. It gives information about new movie releases and you can be able to watch trailers from the movies before you decide to watch it. IMDB is a kind of personalized recommendation and social recommendation system.

It also has some based on what you've previously seen and rated. The system is to a certain degree, flawed. Despite being sort of a social recommendation, because it is based on what other people have previously seen, there is no friend list or similar, so there is no control regarding what people these social recommendations is based on. The personalized recommendations also has some flaws. As the project groups members tried seeing what it would generate, as a member and not a member of the site, it gave the same recommendations, and showed movies that have already been seen and rated.

LinkedIn

LinkedIn is a social networking website for professional occupations. It is way for people to find, and be found, for projects and/or work based on ones skills, previous experience, and descriptions from other users. It uses text analysis to find certain keywords, like "trustworthy" or "dedicated", to highlight a person's abilities. LinkedIn also has a "apply from LinkedIn" button, where users can apply for a position in a company, through their LinkedIn profile.

The system is based on item recommendation, as it is the descriptions, skills, and keywords extracted about the user, that is used for the recommendation. When the user itself search for a person or a company that will be a personalized recommendation. The system could also be some of hybrid recommendation system There can be some problems, if someone begins to write wrong descriptions about another user, both if it intentionally better or worse than it in reality is.

2.3.2 Overview

The most common recommendation type between the four existing solutions is the personalized kind of recommendations. This is followed by the social recommendation type. Item recommendation is visible on some of the solutions, but is less used compared to the other recommendation types.

Personalized and social recommendations seems to fit into collaborative filtering, while item recommendations falls into content-based filtering recommender systems. To cover all possibilities, a hybrid recommendations seems to be the best option, considering Amazon's success with this kind of recommendation system.

It seems like the personalized and social recommendations is the most important, considering what this project is aiming for. Item recommendation is based solely on data regarding a specific item, rather than a recommendation based on the person's interests, past behavior, and similarities with other people. This will be further investigated in the survey section to find out what people weigh the most, when they receive new media they might consume, or find themselves.

2.4 Connections Between Media

This section is going to be about what possible similarities in attributes there can be between different kinds of entertainment media, as a way to recom-

mend content across them. This is going to look at the content-based recommendation possibilities, which is purely based on data and attributes about the content, like genre and involved people, rather than user and friend ratings.

First we have books, movies, and videogames. These three have a clear connection as they can, and often is, adapted into each other. There exist many movies which is based on a book, like the 'Lord of the Rings' trilogy, and various videogames based on books, like 'The Witcher' and 'Metro 2033' series. For videogames, the other way around is usually a result of rising popularity, and uses novels and books for world-building and secondary plot-lines. Genre is also a clear connection, as all three has the same genres, like crime and science fiction, prevalent through them. Associated people can also provide another connection, like a script writer or director, who previously have worked on both movies and videogames. Television shows can also be fitted into these three, especially movies, who shares many of the same similarities.

For something like music, there isn't any clear connections to any of the previously mentioned entertainment media. Music does appear in the visual entertainment media, like movies and videogames, but mainstream music usually only appear in some movies, and more often than not, movies and videogames has their own scores. Genre connections doesn't apply either, as music has its own set of genres, like rock and pop. For associated people there can be some connections, like if a piece of music shares a composer with a score from a movie. When it comes to books, which doesn't have any music attached to its form of entertainment media, there is even less suitable connections to be made. It could be argued that there does exist connections between music and books, because for every topic in existence, there will be books about it. Music could appear in books as a story element, or in educational music books. There could also exist connections with biographies depicting the lives of musicians. This is still problematic as these connections are quite niche. This project's topic is also about entertainment media, so something like educational media does not apply here.

A majority of people do listen to some kind of music though, so it could possible, or more suitable, to link users with similar taste in music. Through that music can be incorporated, together with the connections that do exists. This also applies to any other which has already been mentioned, and so, their recommendations can further be reinforced and improved together with their own connections.

Another thing that could provide usable connections can be indirect connections. For example, if you have a certain book, and it has a movie adaption. This movie uses a certain piece of music, which turns out to have variable that

match with other media items which the same user likes. This can provide a whole new dimension of connections and recommendations, but also make the recommendation process much more complex.

There exist numerous ways to make connections between the books, movies, and video games, which can be used as parameters for generating content-based recommendations. For music though, there is a clear lack of connections suitable for generating proper recommendations, as it shares minor similarities with movies, videogames, and books. The connections that does exist are quite niche, and may serve more of an educational purpose, than entertainment. It is quite clear though, that music is something a majority of people consumes. So if it can't get as many connections to other kinds of media, it could work in a more social kind of way, where it takes personal information into a higher consideration, and links people who have a similar taste in music, and though that might even recommend a movie or a book.

2.5 Surveys

To find out how people generally perceive media recommendations, it was decided that there had to be collected some data from potential users. For that reason, both an interview and a questionnaire were constructed and executed. This was also done for the purpose of approximating a more precise target audience, and create possible weights based on real data, once the project's product is going to be designed. This section will look at and discuss the data collected from the interview and questionnaire, and what was learned from doing these activities.

2.5.1 Interviews

Before we decide how to create a media recommendation product, it would be a good idea to figure out, if there really is a need for such a system. Because of that it was decided that there had to be constructed a interview-questionnaire, which could give an idea on if there are any people who would be interested in using a media recommendation website. Besides questions regarding media recommendations, there was also some questions regarding user privacy rights

The project group were splitted into into teams and then discussed where it would be suitable to go, with the area our project is about in mind. The first team went to GameStop and Fona, the second team went to the library, and the third team went to the cinema. The only requirement there was, was that chil-

dren were not going to be interviewed. The interviewing was conducted with a member of the group approaching a person and ask if they were interested in giving their opinion about different kinds of media. The people who were interviewed did not know the purpose of the questions, until the last question was asked, to avoid people making their opinions out from the idea of a media recommendation system.

Altogether, 15 people were interviewed, all with different ages and jobs/educations. After interviewing, the interview groups met up, and began analyzing all the answers that they had collected from the interviewees. The goal was to find things that the interviewed people agreed on regarding media recommendation, and possibly other information, such as ideas for a possible product. It should also help make clear whether or not a questionnaire survey were required, and if it were, help formalize what it had to achieve.

There was a lot of similarities between the different people who were interviewed. One thing that people especially agreed on, was that they already use similar webpages to what this project had in mind. Another thing that was common about the answers, was that the majority of the people got recommendations from their friends and weighted them higher than critics and other people. However, one surprising thing was that they also weighed their own opinion higher than their friends, even though they haven't seen that movie or book before.

The oldest person who was asked during these interviews was a 52 year old woman, who admitted she most likely would not use such a solution. Besides her there was a pretty large agreement by people, that such a solution would be a good idea in some way and they would most likely use it. This might suggest that a possible solution would be more suitable targeted at a lower age group. As expected, regarding the questions about user privacy and rights, most people were less secure with their personal information being available for other people, as the amount of the information became more severe. From the interview responses, it could be seen that most people tend to pick up recommendations based on a personalized and social way, rather than through an item recommendation.

The following is a list of other aspects which were picked up during the interviews:

- The majority of people were mostly interested in movies and music as their entertainment media.
- It was desired that it you should be able to see how friends have rated different kinds of media.

- A 'Mood System' were suggested, a recommendation feature based on your mood.
- Recommendations from advertisements or newspapers is used, but is less convenient because it is usually not available on the fly.

2.5.2 Questionnaires

The questionnaire survey was created, mainly, to figure out some more precise weights for how people choose the media they want to consume. So not only as a way to validate the results from the interview survey, but also as variables to implement later in a solution. In total, the questionnaire received 82 responses from 52 males and 30 females. The most dominant age group was the 21 to 25 year olds, which stood for over half of all that answered the questionnaire. This can most likely be attributed to the group members entourage. There was almost an even spread between the different media, with books being the lowest and movies being the highest, in what they use on a daily basis. The questionnaire used a scale from 1 to 5, to rate how high they weigh a certain factor about a certain type of media, where 1 is a low weight and 5 is a high weight.

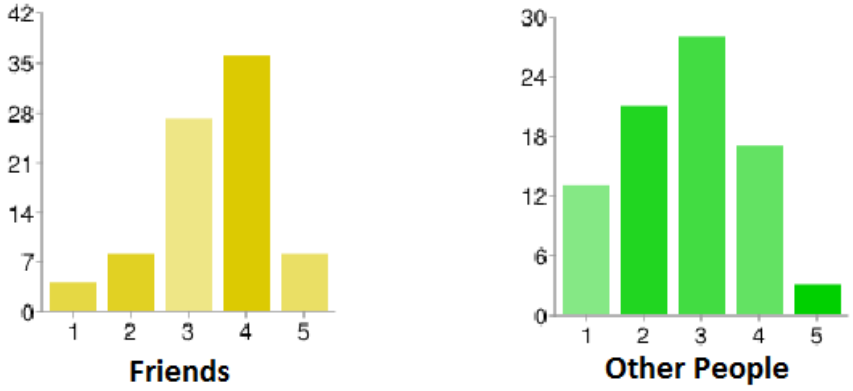

Figure 2.5: Graph showing how much people weigh recommendations from their friends and other people

The questionnaire started by asking three simple questions about how they weigh specifically peoples opinions when it comes to new media, including their friends, strangers, and themselves.. From the responses it is clear that

they weigh their friends recommendation higher than strangers, which mainly got a 4, and strangers lower with a 3. The total spread can be seen in 2.5 In the final of the three question it was asked if they weigh their own opinion higher than their friends, and the answer was clear, that they do listen to themselves firstmost, rather than to their friends, when it comes to a new piece of media, which can be seen in 2.6.

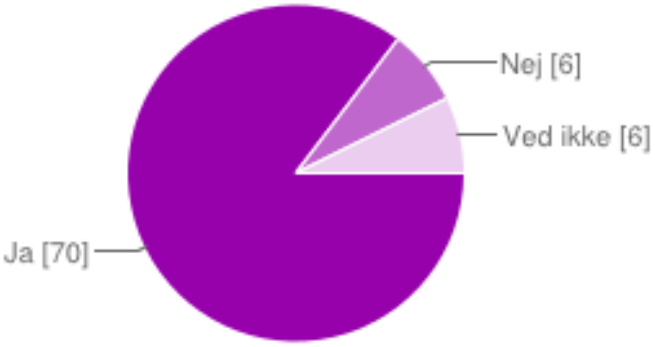

Figure 2.6: Graph showing how many people weigh their own opinion higher that any other people

Then the questionnaire proceeded to ask more specific question about different types of media, namely movies, books, music and video games. It asked them how high they weigh a certain factor regarding a type of media, like the director of a movie. Something that was present through all the different types of media, was that the genre of the media had a clear high weight, almost universally, when picking a new piece of media. Genre was also clearly superior compared to every other factor for a genre. Besides genre, there was also other, more niche, factors to take into consideration. If we look more directly on the different media we can see that:

Movies:

Factors like who directed the movie had a moderately high weight, together with which actors was part of the cast. These two was the most prominent, where the last one was other associated people.

Books:

For books, besides genre, only the author was asked about. The author had a moderate weight for people, but still seems to be quite important for many people.

Music:

When it comes to music, it seems only the specific performer of the song weights highly, while other associated people have a low weight.

Video Games:

Besides the genre, only which studio made the video game had a high weight. Unlike music and movies, actors and voice actors had a very low weights for video games.

People seems to be looking for trending and know details about a piece of media, like the name of a certain actor or actress, a studio, an artist, etc. Most people seems to favor the more apparent factors of a media, like the actors in a movie, rather than the more obscure, like voice actors in a video game. It is also clear that they weigh their own opinion higher, but still hold other peoples opinions to some regard.

2.6 Target Audience

To find our target audience we first want to examine who might be interested in our potential solution. To do this we have taken several different approaches, first we created personas and use cases to try and get an idea of what groups of people we want to target our program towards. To try and confirm our assumptions, further understand who our target audience is and also define them more accurately, we did interviews.

2.6.1 Personas and Use Cases

Through doing our personas and use cases we got a deeper understanding of who is going to be our target audience. We started by creating personas for three of what seemed like the most obvious users, while constructing situation where our project could might be useful. Along with helping form our interviews it also helped form our program, through showing what kinds of features different audiences might want.

It also helped showing us what age groups we might want to ignore. Because children and people with certain disabilities, particularly bad eyesight, impose certain restrictions on a potential solution, restrictions that is not immediately necessary for our program to function, we have decided to exclude these groups from our initial demands for the program. Note that we consider kids to be everyone younger than 17 years, because that is the oldest age restriction you can find on media(in EU at least). Based on that we have decided to focus our solution towards people in their late teens and people without

disabilities.

2.6.2 Interviews

We can see from the interviews that younger people seem to have more interest in our proposed project. One reason this might be the case is because younger people already use the internet to find new media to consume, and most also use other recommendation systems. While our sample size isn't big enough to make a definite conclusion it helped us define questions for our questionnaire, such at the what age groups we should split people into.

The interviews also hint that peoples' current employment status doesn't have a significant effect on their interest in our project. There was one person that outright said she had no interest in our project, and had only one relevant difference from the other people we interviewed which was her age. So that doesn't signify a trend based on employment. And the other 10 interviewees were a mix of unemployed, employed and had various educational levels.

To note about privacy which was a minor focus of the interviews we see that age doesn't really impact concerns on privacy. While younger people were a bit more specific on the technicals and on the different forms of data, they all agreed that they wanted to choose what data the company keeps on them. They also agreed that the data, which the company does not need to run their service, should never be stored.

The interviews not only helped us confirm that there is an interest in the project but also to specify our target audience. We now know that our focus should be towards the age group 18-50, based on the responses we got.

2.6.3 Questionnaires

The questionnaires showed that all age groups have, to some degree,an interest in some kind of media. All age groups also seems to put more weight on the more apparent aspects of a piece of media, like the cast or the artist, and especially the genre. The same also seems to apply to how much they weigh whether or not a recommendations is from a friend, and their own opinion.

There was an overabundance of responses from the 21-25 age group, which can most likely be attributed to the group members entourage. It could be seen though, that generally, as the age group became older, the responses seemed to turn more apathetic. Still, it is hard to determine a precise age group, since responses from more older people is scarce, and there isn't enough responses

to confidently determine anything. It does help validate that there is an interest among younger people.

2.7 User Privacy and Rights

For a recommendation system it is required to collect data about the intended recipients of the recommendations. The recommendations become better and more precise, as the amount of data becomes bulkier, so a recommendation system will usually try to gather as much it possibly can. This can also be called data profiling, where the intended system for this project would categorize people depending on their taste in media, personal information like age and sex, and their common friend connections. In other instances it could might also include variables like previous purchases, view history, tags, and keywords extracted through text analysis. All of these might be more revealing, and raises privacy concerns.[**?**]

Even with this amount of data gathering this project plans to handle, there will be some data privacy concerns. Users will most likely be required to register a profile or account to utilize the system, and will have to hand over personal information for verification and recommendation purposes. This is of course not any different from many other systems who does the same thing. But in this case, we're talking about a recommendation system, which poses some new challenges in this aspect. Because recommendations can be based on the data of other people, it would make it possible to deduce connections to other people. Especially if it is through some uncommon element, like an obscure cambodian film, where fewer people is connected to. This can ultimately lead to personal information being found out. This risk is further heightened if the person also has access to the database, and can place queries. [**?**]

This is a glaring challenge for any recommendation system, but is also a complex and difficult problem to solve, or even answer to. It also depends on exactly how this projects product is going to be constructed, which is not yet formalized, to properly answer these questions. At this point it might be out of this project's scope.

We conducted some interviews where we asked potential users several questions regarding the project, including questions regarding how much information they would be okay with being available, either to a company or the entire public. See Section 2.5.1 The questions was asked in a rising degree, to see exactly where people's threshold would be. Most people were okay with

their contact information being available, as it was most likely already available in some form of way. When it came to their more personal information, like interests and age, they were more uncertain. Almost all was okay with having their very personal information, like emails, chats, and pictures, available, as expected. An interesting feature a lot of them mentioned were an option to choose whether or these information was shared, for the user themselves. This could be a more user-engaging way to collect information for the recommendation system, and solves most problems regarding user rights, as the user decides for themselves what they want to share.

2.8 Project Boundary

This section is going to make a boundary on the width of this project's problem area, based on the previous problem analysis. This is done to limit the project to what is the most important, which is based on survey results, and other information collected or augmented throughout the problem analysis.

Through the surveys that was done, both the interview and the questionnaire, one of the goals was to find out what kind of recommendation has the highest appeal for most potential users. There was three kinds of recommendations which was evaluated; Personalized, social, and item. Looking at responses generated with the surveys, it became apparent that social was the dominant kind. Followed by personalized, realized with recommendation sites like IMDB, and lastly item recommendation. Recommendations based around the person itself, and its nearest social bonds, seems to be the most common kind of recommendation. Based on this, the project will be limited to focus on social and personalized recommendations. For a completely finished recommendation system, a hybrid of all three should be considered, including item recommendation, as it has its own useful features.

Regarding what kinds of media should be included in the project, the answer is not quite as clear. It was augmented for in section [REF] that there exist clear connections between books, movies, and video games, while music was the odd one out between the four. But, since it has already been made clear that the project will be based on social and personalized recommendation, this becomes less of an issue. With this the recommendations will not be generated purely on attribute connections between media, but by data collected from users and their connections to other users. Survey results from the questionnaire also back up this stance. Despite this, it is necessary to limit the project to gain more simplicity, as recommender systems is already quite complex. In a

finished product it is expected that all entertainment media forms is included, but for this project, music is going to be excluded for initially reducing the workload.

Early in the project it was considered that issues like user privacy and rights was to be tackled in this project, since a recommendation system uses personal information regarding its users to generate recommendations. As expected, the interview survey further backed up this stance, as most people who were asked were less secure with their personal information being used without their consent. A possible method was suggested in section [REF], like giving the user themselves the possibility to choose whether or not they give consent to using their personal information, with less accurate recommendations as a consequence if they should decline. Despite this, the problem with user rights is difficult to answer, as it is quite an ethical problem, and therefore out of this project's scope. For this reason, user privacy and rights will not be looked at further in this project.

The precise target audience for this project is hard to set, as entertainment media is something almost any age group consumes to some degree. What has been determined is that the project will not cater to children, and therefore no entertainment media specifically targeted towards children. This is done to prevent making the designing process more difficult, as the product would have to cater to children otherwise. You could imagine that the site could be used by parents to find children entertainment media to enjoy with their children, but for now, the project will be limited away from this kind of feature. People with physical disabilities, or other disadvantageous which prevents normal use of personal computers, is also not included in the target audience. This is done for the same reason as children.

A more precise age group for the target audience can be specified by looking through survey results. In the interview survey, a 52-year old interviewee showed disinterest in the overall concept of this project, and was less aware of recommendation possibilities that can be generated through websites like IMDB. This trend became more apparent depending on how old the interviewee was. The questionnaire survey showed, as expected, that all age groups has some form of interest in entertainment media. Still, the exact age group is hard to set, but considering what has already been stated regarding children, the lower bounds of the target age group can be set to around 17 years. The upper bounds is less clear, and can be set from anywhere between the 30'ies to the 40'ies. These bounds is put in place to figure out who the possible solution should be designed for, rather than what entertainment media can be excluded, as even a fifty year old movie can still be popular today.

2.9 Problem Formulation

In the previous problem analysis, there has been argued for various kinds media recommendation systems. It was argued that there didn't exist much when it came to media recommendations who could recommend different kinds of medias, and so this became the main goal of this project. Through statistics, and later by conducting surveys, it was determined that most asked people used more than one kind of media regularly, and saw viability in the proposed system. This also produced crucial data, which can be used to weight the importance of various attributes in media, and determine which other users in the system should have more influence on the recommendations. Various media connections was also explored, to try and build bridges between different media, which can be exploited in the recommendations process. Based on this, and the analysis, the following problem formulation has been made:

- How can you recommend pieces of media across different kinds of media, based on similarities in general item information, and similarities with other people?
- How can you construct a hybrid recommendation algorithm, based on collaborative filtering and content-based recommendation, to support recommendations across media?

2.9.1 System Definition

An IT-system used to recommend new media content, based on previously watched content, and other factors like trending or the interests of friends. The system is mainly a recommendation system, but can also function as a way to introduce an user to other kinds of media. There is also minor aspects of a social media system, like a friend list. The system must be based on a website, and can therefore be accessed from any device with a suitable browser and internet connection. The system is going to require a wide array of users, user data, and media data, to generate better and more precise recommendations.

2.10 Product Requirements

For this project there is going to be prepared an object oriented solution, in the form of a program, which allows a person to list media which he or she has consumed, and then receive recommendations for new pieces of media, based

on previous behavior. The system should be approachable, but will have a focused target group from the late teen years into the thirties. The system should function as a way to manage and keep track of previously consumed media, but also as a way for the person to discover new media, based on the recommendations the system generates. Besides data collected from the users previously watched media, which is content-based recommendation, there is also going to be recommendations based on other people who also use the system, which is collaborative recommendations, making it a hybrid recommender system. The product is also going to prepare a fitting graphical user interface, in the form of a web page.

- Functional Requirements
  - It has to be possible for the prototype to generate recommendations based on a hybrid recommender system.
  - The prototype should have a fitting graphical user interface for the chosen target audience.
  - The prototype has to be usable, and as a minimum, be able to run locally on a regular desktop PC with an internet browser, e. g. Google Chrome.
  - The prototype should implement a rating system, where indicate their satisfaction with a piece of media based on a scale from e. g. 1 to 5.
- Non-Functional Requirements
  - The prototype, together with the associated project report, have a shared deadline the 21th of December 2013.
  - The prototype has to be written in the C# programming language, following object oriented programming.
  - The prototype and the project has to support the requirements and learning goals which the curriculum prescribe.
- Solution Goals
  - The hybrid recommendations should help the system create more correct recommendations, by applying both content-based and collaborative recommendation.

– It should be possible for recommendations to be across different kinds of media, e.g. from data regarding movies into a video game recommendation.
– Recommendations should have additional weights added, to favor certain aspects of a media like which person it originated from or preferences, based on survey results.
– It should be possible for the users themselves to alter the recommendation process, to some degree, through preferences and indicating unwanted factors in a piece of media.

Chapter 3

Design

This chapter is going to cover the initial that was carried out, to form some sort of structure for the system, and determining methods to achieve this. There is going to be looked possible mathematical methods to support recommender systems, the proposed design of the graphical user interface, and patterns to implement said user interface.

3.1 Algorithm Design

Through the previous analysis of different approaches to making a recommendation system it was recognized that it was needed to do a variation of the hybrid recommendation model, the hybrid model being a combination of the content-based and collaborative forms of recommendation. It was decided to approach it very departmentalized. This means the two methods won't directly interfere with each other, which this leads to them remaining fairly simple while still getting the benefit of the hybrid model. This however also means the feedback loop will be slightly weaker. But with the gained simplicity the slight precision loss from the weaker feedback loop is, at least for this projects prototype, worth it.

The algorithm ended up being split into three main steps. See Figure [REF]:

- Collaborative filtering
- Content-based filtering
- Merge

We will go more into detail about the mechanics of the collaborative and content-based filtering algorithms in the following part of this section. The

basic idea of this structure is that the two filtering algorithms each pick a list of the best suited media recommendations for the selected user, based on some kind of weight or coefficient. Then in the merge part the two lists is merged into one list, creating a more precise list than the two recommendation forms could do on their own.

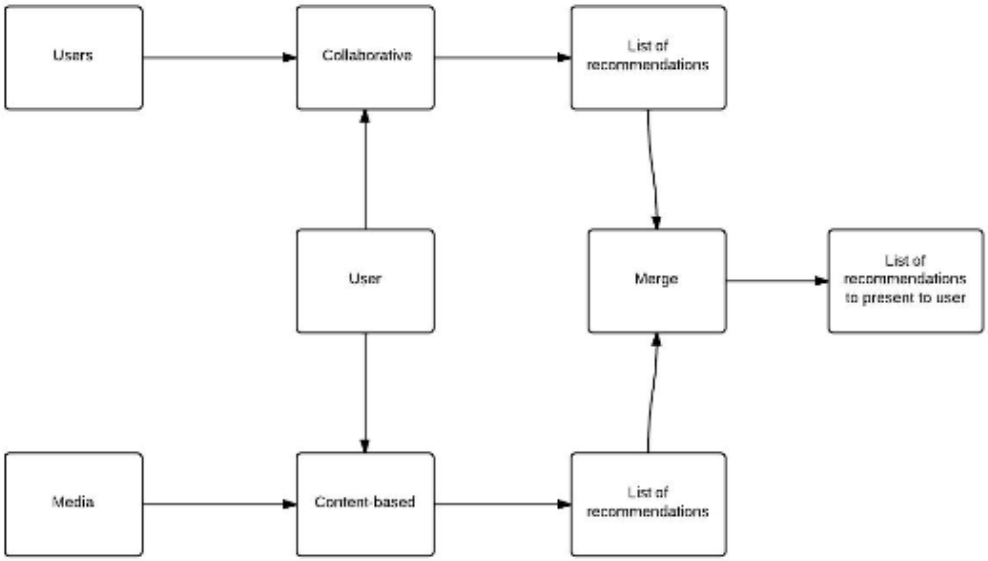

Figure 3.1: The general structure of the recommenendation algorithm

3.1.1 Collaborative Design

Pearson

The pearson product-moment correlation coefficient, or just pearson correlation coefficient, is a way to measure the linear relation, or dependence, between the variables of two sets. This measure can be used to determine the relationship between things like age and blood sugar, height and efficiency in basketball, and how similar people are, based on their taste in media. [**?**]

The pearson correlation coefficient creates a linear line of best fit for the two variables, and based on this linear line of best fit, returns a coefficient, which is how much the data of the two variables deviate from this linear line. In Figure

3.2 you can see what kind of pattern of data will generate positive, negative, and no correlation using the pearson correlation coefficient. [**?**]

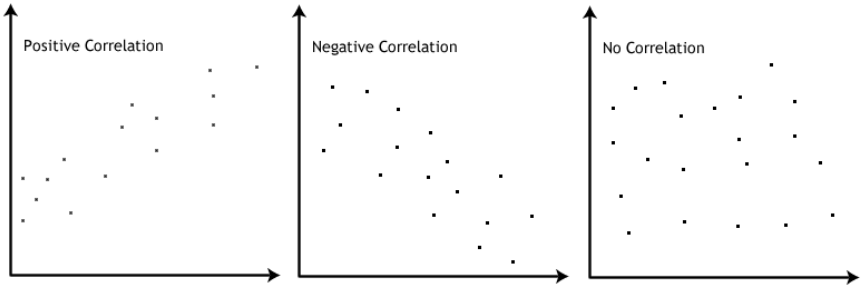

Figure 3.2: Showing how two data sets can give differnt Pearson results

The returned coefficient can be in the range of -1 to 1, where -1 is a complete negative relation, 1 is a complete positive relation, and 0 is no relation. Inside this range, there is varying degree of positive and negative relation, with the range of -1 to 0 being negative, and 0 to 1 being positive. This can be seen in Table 3.1. Pearson very rarely returns perfect relations -1 and 1, and no relation 0. [**?**]

Strength	**Positive Relation**	**Negative Relation**
Weak	0.1 to 0.3	-0.1 to -0.3
Medium	0.3 to 0.5	-0.3 to -0.5
Strong	0.5 to 1	-0.5 to 1

Table 3.1: Pearons strength scale

In a graph representation, the positive relation would be a linear line going upwards, the negative relation would be a linear line going downwards, and the no relation would be a horizontal line. This can be seen in Figure 3.3. A perfect 1 or -1 relation would be if every point in the graph was placed on the line. [**?**]

In the context of this project and its possible product, the two variables which would be examined by pearson, is two different people, who uses the product. Pearson would then find the relations between these two people, and use that coefficient as basis for the recommendation process. The data sets it would take in from the two persons, would be the rating they've given to the same pieces of media. A point on the graph representation would then be two ratings for the same piece of media. The returned coefficient is then going to

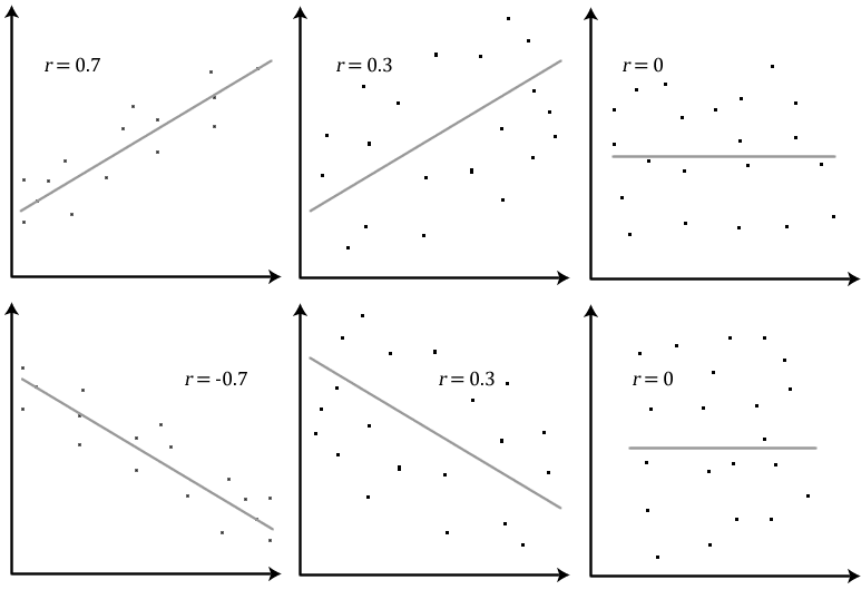

Figure 3.3: Showing line representations of data sets, and their correlation

be how similar their rating are, and their rating habits. Using the rating given to the media in the product means that pearson is going to supplement the collaborative filtering in the recommendation process.

The calculation to create the pearson correlation coefficient can be seen in Figure 3.4. In this calculation, x is the set of ratings from the first person, y is the set of ratings from the second person, n is amount of media items which is examined, and r is the returned pearson correlation coefficient. [**?**]

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\, n\Sigma x^2 - (\Sigma x)^2 \,][\, n\Sigma y^2 - (\Sigma y)^2 \,]}}$$

Figure 3.4: How the Pearson coefficicent is calculated

Spearman

Besides the pearson correlation coefficient, it was also considered to use the Spearman's Rank correlation coefficient, which in many ways works similar to pearson, but has some significant differences. Actually, Spearman is defined as the Pearson correlation coefficient, but between the ranked variables instead, which is a different way to handle the two sets of data. [**?**]

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}.$$

Figure 3.5: How the Spearman coefficient is calculated

The spearman correlation coefficient starts by finding two ranking lists, one for each of the sets being processed. The highest rank is either assigned to the lowest or the highest variable in each set, depending on if the set is sorted ascending or descending. The same applies to the lowest variable in each set. Every variable pair should then have a respective rank, reflecting their individual position in the sets they came from. Next, to find the d^2 variable seen in Figure 3.5, you find the sum of the difference between all the ranking pairs squared. See Table 3.2 for an example. In the example, the sum of d^2 is 194, n is the sample size of 10, and returns a correlation coefficient of -0.176, which is a low correlation. [**?**]

One more thing to note about spearman is that if you have tied rankings, you have to give them the average of the ranks they otherwise would have had, and use a different way more similar to pearson to calculate the coefficient. [**?**]

The coefficient generated by spearman works on the same scale pearson does, so Table 3.1 can again be referenced for the meanings of the coefficients. The spearman correlations coefficient is based on the monotonic relationship the two sets of data create[**?**]. A monotonic relationship is when the graph representation of the data sets is either always rising or always falling, each respectively giving the perfect and negative perfect coefficient. The variable pairs which deviate from this relationship will worsen the coefficient.

Comparison

While pearson works on a linear relationship, spearman's correlation coefficient works on a monotonic relationship. As seen in Figure 3.6, the spearman

x	y	x **Rank**	y **Rank**	d	d^2
86	0	1	1	0	0
97	20	2	6	-4	16
99	28	3	8	-5	25
100	27	4	7	-3	9
101	50	5	10	-5	25
103	29	6	9	-3	9
106	7	7	3	4	16
110	17	8	5	3	9
112	6	9	2	7	49
113	12	10	4	6	36

Table 3.2: Spearman example on the correlation between IQ and amount of TV watched

correlation coefficient will return a perfect relation in the case that the monotonic relationship holds. Spearman is also softer, compared to pearson, in a sense, since deviations has less of an impact, which can easily ruin a high scoring coefficient with pearson. In the context of this project, this property doesn't suit what the algorithm is supposed to find. With spearman, it could return a perfect relation, even when the two users haven't given the same piece of media the same rating. The figure also shows the pearson correlation coefficient for the same set of data, which returns a strong, but not perfect, positive relation. This makes more sense in this context, as the peoples ratings is similar, but not exactly the same. Also, the algorithm should strive for the highest possible coefficients, and best possible matches, and therefore is spearmans more softer approach not optimal. Because of this pearson was chosen to be used in the project.

Pearson does have some problems, which can be attributed to the same problems collaborative filtering has. If there is a lack of a suitable large size of data, then pearson can generate inaccurate, and possibly wrong coefficients. For example, if two people have a small number of consumed media in common, small deviations in ratings from each person will have a larger effect on the returned coefficient. It is a problem which will fix itself slowly with time, as more data accumulates and becomes available to the calculation. Also, if there is only two pairs of variables, pearson will never return anything but perfect and no correlation coefficients. Because of this it should be considered to implement a secondary part to the recommendation algorithm, to combat this possible weakness in the collaborative filtering, and work as supplement

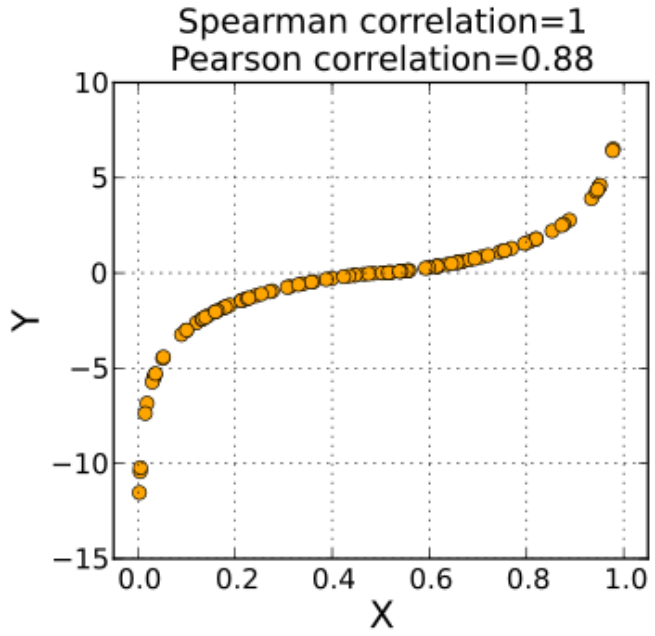

Figure 3.6: How the Spearmen measure works, compared to Pearson

to the recommendation algorithm.

3.1.2 Content-Based Design

3.2 GUI Design

One thing to have in mind when it comes to design, is colours, and it is very difficult to choose which colours to use when designing an interactive system. For example, Microsoft use blue in their Windows operating systems as an background colour, because blue means calm. And Apple also uses blue when it comes to their products. Like these examples every company has thought hard about what colours to use when it comes to designing their website. Another thing to have in mind is that the same colour can have a different meaning around the world. And example from [**?**] is that the colour blue is seen as a bad thing for those working in healthcare and in business it means reliability. In the table below one can see how the majority of the western world sees what the different colours means. Figure 3.7 shows this.

In the Figure 3.8 one can see the Login screen. We have tried to keep it simple and smooth and nothing to distract one from doing what have to be done. It is simple because we have few buttons and few interactions with the

Red	Danger, hot, fire
Yellow	Caution, slow, test
Green	Go, okay, clear, vegetation, safety
Blue	Cold, water, calm, sky
Warm colours	Action, response required, proximity
Cool colours	Status, background information, distance
Greys, white and blue	Neutrality

Figure 3.7: Colors and the interactive meanings

user. The colour we have choosen to our login screen is blue and white, which we according to the table, thinks are the right colours for the screen.

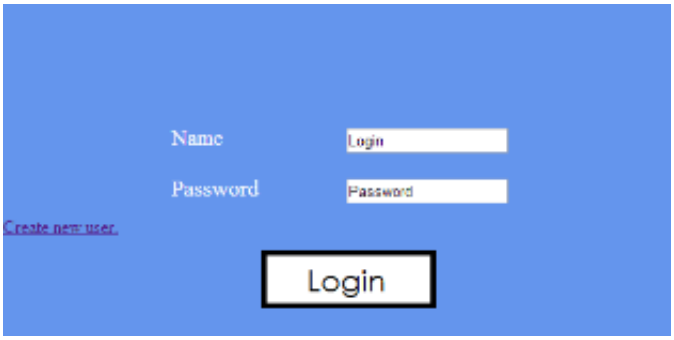

Figure 3.8: Design of the login screen

When you have passed the login screen you come to our front page. Our front page is where you can go to everything relevant for the user. Again we have gone for a simple and clean look, with only buttons that are necessary. It also has a, if the user decides, profile image and the username beside it, to indicate that the user if logged in.

If we look at the sketch we made a long time ago in the start of the project it doesn't deviate too much from the product we have now. We have the same thing with profile picture and username, but have removed the option to insert a new media into ones medalist from the top bar. We have also removed the option to chat to others through the website through the chat system we wanted to implement. Instead of a home button that looks like a house in the lower right corner, we moved it up to take a more central placement and placed it on the top of the list of menu buttons. In that way it is more convenient for the user to come to the front place. The add media from the sketch, we have moved to the button medialist, which makes more sense and is more

convenient. The central place where all the media is displayed, have stayed relatively the same, with a list that shows all the relevant media. Though, this have been collected into a single tab, for all recommendations given to the user. Again we have gone for the simple colours with blue and white. See Figure 3.9 for old design and Figure 3.10 for the current design.

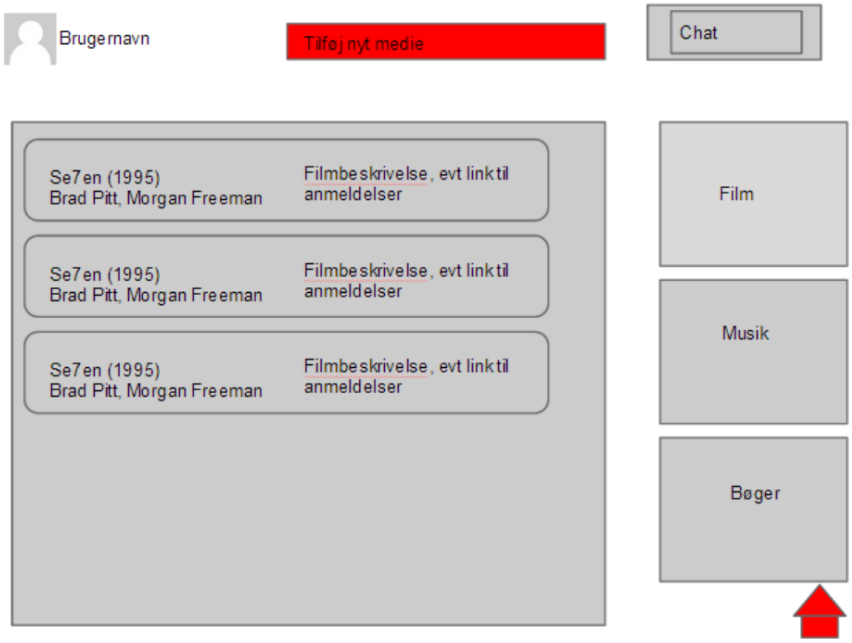

Figure 3.9: Early prorotype of the website design

3.3 Design Patterns

This section is going to be about the possible design patterns, which is available to structure our system, implement a user interface, and eventually help simplify the testing process.

MVC

MVC stands for Model-View-Controller. The MVC pattern separates the application into three main components: Model, View and Controller. The model manages the behavior and data of the application, and responds to requests about its state, usually from the view through some kind of event, or the controller. It also responds to instructions to change state, which is again issued by

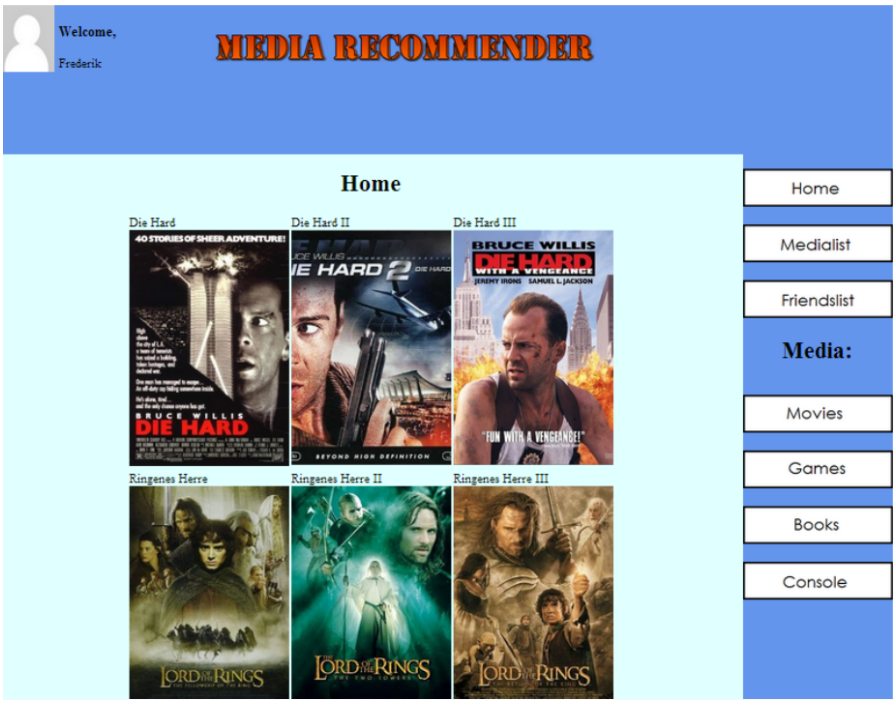

Figure 3.10: Current version of the website design

the view, but is performed through the controller. The view displays the state of the model through the graphical user interface, which as said, is received through the controllers, or indirectly by the model. [**?**]

MVVM

Another pattern which which considered for this project was the MVVM pattern, which stands for Model View Viewmodel, its three main components. Like MVC, MVVM separates the system into three different components, creating a layered structure. The difference between is that MVVM is much more strict in this separation, which limits the model component to simply containing information about objects, and the view to simply provide a user interface. [**?**]

The viewmodel is the component which implements most of the logical code, and is the binding point between the model and the view. The viewmodel implements commands, which for example is bound to a button on the view, together with restrictions, and take over the moment a user pushes said button. Beyond the button click, the view have given over full control to the viewmodel. The viewmodel also implements restrictions on the underlying

model, so the properties and logic a model normally would have is located in the viewmodel instead. [**?**]

Comparison

Both patterns provide this layered structure, which can help with testing the system later on, since it is divided into a workable bits following these patterns. For this project, the MVC pattern will be used, simply because it is the more simple of the two patterns, and still provides a good testing structure, despite being looser than the MVVM pattern. MVVM also requires more graphical design, since the view cannot have any notable code-behind. The technicalities behind graphical design is not the part of this projects goals, and is further reason for why MVC was chosen. The MVVM is also better suited for regular desk application, and since the one of the goals is the produce af web application, MVC was again weighed higher than MVVM.

3.3.1 Implementation

In the Figure 3.11 above is a representation of how the MVC pattern is planned to be structured in our web application.

The model is the part of the MVC pattern which contains the general data of the systems classes, and their objects. You could say it is the database, or the class structure of the system. Besides storing class and object information, and communicating with other model members, the model component does not do much on its own, and is largely oblivious to the surrounding structure.

The controller is another part of the MVC pattern which gets information from the model and the view which does the "hard" work. It calculates depending on if it received information from the view or the model, and returns something to be shows on the view, or something to be stored in the model. It means the controller is the brain of the MVC pattern, and is the binding point between the view and the model.

The view is the part of the MVC pattern which basically is everything visually you can see and change on. What is viewed is usually the classes and objects which the model contains, which has been translated to a more understandable format in the graphical user interface, which the user then can see and manipulate. Any changes the user performs goes through the controller, and down into the model. The view is just as "stupid" as the model, as it cannot do much on its own, and depends on the controller, and indirect events.

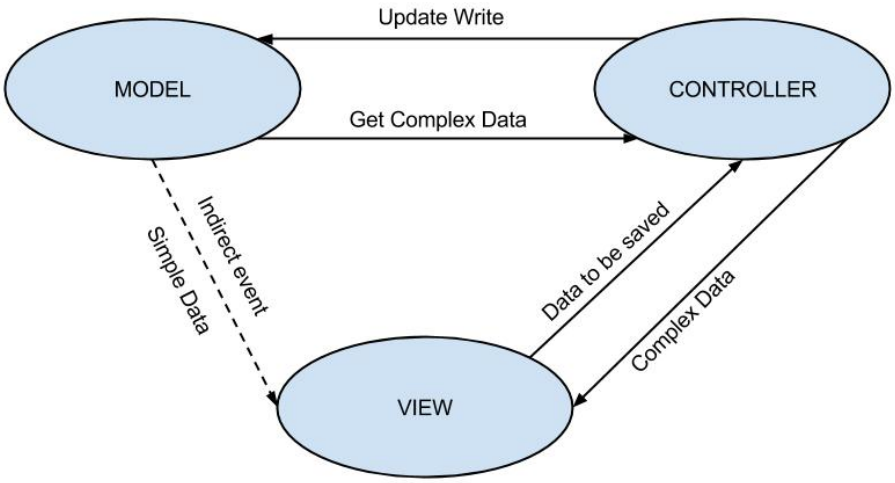

Figure 3.11: How the MVC is planned to be implemented in this project

The whole point of using the MVC pattern is to create some kind of layered system, dividing the responsibilities into several components. The MVC pattern can be done in various ways, for many different purposes, but this specific pattern is what we found the most appealing for this system. It places the largest need for testing onto the controllers, which contains most of the logical information processing, and the recommendation algorithm. While the graphical user interface is not completely interchangeable like it would be in a MVVM pattern, it is still easier to experiment and further develop with than without the MVC pattern.

3.4 Summary

Chapter 4

Program

This chapter is going to cover the system which was created for this project, based on the previous analysis and design chapters. The contains details regarding the implementation of the recommendation algorithm, the final version of the graphical user interface, API's, and testing.

4.1 Algorithm Implementation

This section is going to again look at the parts which makes up this projects recommendations algorithm. This time, the previous theory is going to be applied and explained in the context of this system, and how it is utilized and implemented in our system.

4.1.1 Collaborative Algorithm

The collaborative side of the algorithm starts by receiving a single user, and afterwards get all different users put into a collection. With the IndexUsers method, a dictionary is created, which contains pairs of a user from the collection, together with a coefficient, based on the linear similarity between them and the main user. This coefficient is based on the Pearson correlation coefficient which has previously been described in Section 3.1.1.

Pearson

The two sets which is needed for each Pearson calculation, is the ratings of media items, which both the main user and secondary user have in common in their medialists. The Pearson calculation will then return a coefficient based on

these two collections of ratings, indicating how similar they are in their ratings, and general rating habits. Following this, users with negative or no correlation will be removed, and the remaining is sorted by their given coefficient, going from highest to lowest. See figure 3.4 for the Pearson calculation.

x	y	xy	x^2	y^2
2	3	6	4	9
4	5	20	16	25
6	5	30	36	25
8	7	56	64	49
20	20	112	120	108

Table 4.1: Pearson Example

In the above example is two sets of ratings, x and y, which is then put together to create 5 sums, which also can be seen above. These sums is then used as the parameters for the Pearson calculation, together with the number of pairs, which is 4. In this example Pearson returns 0,949, which is a very good coefficient. See Listing 2 for the part of the algorithm which finds these ratings.

```
1  private double CompareUserPair(User mainUser, User user)
   {
3    List<int> mainRating = new List<int>();
     List<int> userRating = new List<int>();
5    int x = 0, y = 0, xy = 0, x2 = 0, y2 = 0;

7    foreach (Media media in user.MediaList.Keys)
     {
9      if (mainUser.MediaList.ContainsKey(media))
       {
11       mainRating.Add(mainUser.MediaList[media]);
         userRating.Add(user.MediaList[media]);
13     }
     }
15
     if (mainRating.Count() < 3)
17   {
       return 0;
19   }
```

```
21    for (int i = 0; i < mainRating.Count; i++)
      {
23      x += mainRating[i];
        y += userRating[i];
25      xy += mainRating[i] * userRating[i];
        x2 += (int)Math.Pow(mainRating[i], 2);
27      y2 += (int)Math.Pow(userRating[i], 2);
      }
29
      return Pearson(x, y, xy, x2, y2, mainRating.Count());
31  }
```

Listing 4.1: The CompareUserPair method

The initial loop runs through the main users medialist, and if the secondary user have the same media in theirs, their ratings will be added to two lists. If this loop couldn't find more than two media matches, the method will return 0. If you only have to sets of ratings, Pearson will always return either -1, 0, or 1, which is not a precise coefficient for how similar they, and may result in wrong recommendations. As seen in Section 3.1.1, Pearson tries to make a line of best fit between two sets of data, and if there is only two points in their graph representation, the line will always be a perfect fit.

The parameters for the Pearsons calculation is then determined through another loop, if there was a enough rating pairs to work with. The result of Pearson will then be returned. Another problem with Pearson happens when all ratings extracted from one user is identical. For example, if every element in y from Table 4.1 is 5. If this happens, Pearson cannot create a proper coefficient, and will return NaN, not a number. These users who is assigned NaN is later discarded. This may mean users with good coefficient can end up being ignored.

Recommendation Extraction

After every user have been assigned a coefficient, the next part of the algorithm begins. This part extracts media from the rated users medialists, and returns these media as recommendations. See Figure 4.1 for an example of this process.

In the example, user A is the user which the algorithm is trying to generate recommendations for. User B and C have both been assigned a coefficient, where B's coefficient is higher. The algorithm will begin adding media from

B's medialist he have rated higher than a certain threshold, to the collection which will be returned as the generated recommendations.

Once the algorithm begins looking through user C, there have already been added several media to the recommendation list. Users with higher coefficients will always take precedence, and so does their media in the recommendation list. But, this may change in two scenarios: If the same media appears in an other users medialist, or if a user is friends with the main user. In the example, media 4 appears again, which boosts it up the recommendation list by one. Because C is friends with A, media 4 is boosted yet again by one. All other media is also boosted by one because of this friendship relation, effectively giving media from user C an advantage. Through this extraction the returned media follows three priorities: first by highest rated user, second by if the same media have already been added, and third by whether or not the user is a friend.

4.1.2 Content-Based Algorithm

The content-based part of the algorithm starts like the collaborative part by generating coefficients, but this time for media items instead of users. This is done by comparing vectors which each media and user has, that indicates whether or not they have a certain property for media, and preferences for users. These properties is currently genre and which associated people is connected to the vector. For the media, the vector is simply 0 and 1's, indicating whether or not they have something, while the user vector can be higher, lower, can decimals, which is altered by personal preferences. An associated person can be an actor for a movie, or an author for a book.

Vectors

These vectors is dynamically updated, as new associated people enters the system. The genre part of the vector is static, and corresponds to the predefined genres available in the system. When a new associated person is being created in the system, the users vectors is updated to accommodate this. This means that the user will always have a vector with the same length as the media with the most recently added associated person. See Figure 4.2 for an example.

In the example above, each media is added to the collection one by one, with each addition all users in the system is updated as it is required. Red numbers indicate that the certain person was not already part of the vectors, and therefore had to be added to all the user vectors, and the certain media

itself. Green numbers indicate the certain person already have a spot in the vectors, and therefore did not have to update all the user vectors.

Users can change their vectors by adding media to their medialists. This process will convert the rating which the user gave into a number, using the method seen in Listing 4.2.

```
private double ConvertRating(double rating)
{
  return (rating / 5) - 1;
}
```

Listing 4.2: The CompareUserPair method of the recommendation algorithm

The user can give a rating in the range of 0 til 10. The ConvertRating function will then return a number between -1 and 1, where -1 is 0 and 1 is 10. This number will then be added to the user vector in each corresponding spot where the added media has a 1. For an example, see Table 4.2.

Media	**Rating**	**0**	**1**	**2**	**3**	**4**	**5**
Media 3	7	0	0.4	0	0.4	0	0.4
Media 2	2	0	0.4	-0.6	-0.2	-0.6	0.4

Table 4.2: User vector alteration upon adding media to their medialist

In the above example, a user adds media 3 and 2 which can be seen in Figure 4.2, and gives them a rating of 7 and 2, respectively. The 7 is converted into 0.4, and is added to the user vector corresponding to the media vector. The same happens with the other media, except it is a negative number. This is the result of a user giving a media item a low rating. The user vector should then be a representation of what he likes and dislikes.

Vector Comparison

By comparing the vectors, you can generate a coefficient for how well a certain media suits a user's taste. If the media hits a lot of the positive spots in user vector, it will receive a high coefficient. Of course, it can also receive a low, or even a negative coefficient, if it hits negative spots in the user vector. See Table 4.3 for an example using the previous user and media examples.

The comparison is done using the cosine distance as described in Section 3.1.2, and returns -0.062 for the first media item and 0.247 for the second media item. During the content-based algorithm, every media which the user haven't

User	0	0.4	-0.6	-0.2	-0.6	0.4
Media 1	1	1	1	0	0	0
Media 4	1	1	0	0	0	1

Table 4.3: Content Example

already consumed, is assigned a coefficient through this method. The algorithm will then sort, and choose the media with the highest coefficients, and returns them as its recommended media. See Listing 4.3 for the part of the algorithm which calculates this coefficient.

```
private double CompareVectorRepresentations(List<double>
    userRepresentation, List<bool> mediaRepresentation,
   double userLength, double mediaLength, Media media)
{
  double commonPoints = 0;

  for (int i = 0; i < mediaRepresentation.Count; i++)
  {
    if (userRepresentation[i] != 0 &&
   mediaRepresentation[i])
    {
      commonPoints += userRepresentation[i];
    }
  }

  if (commonPoints == 0)
    return 0;

  return FindCosineDistance(commonPoints, userLength,
   mediaLength);
}
```

Listing 4.3: The CompareVectorRepresentations method

The media vector is actually a list of booleans, rather than a list a doubles, like the user vector is. The loop will run for the length of the media vector, and will ignore anything past that point in the user vector, since it cannot hit any more common points. If any common points were found, the function will return the result of the cosine distance calculation. The lengths were determined prior to the call of the compare functions, and were passed as parameters.

4.2 GUI

4.3 API

In order to get the media data which is needed for the application to run at an acceptable precision level it was needed to access several open-source (that is databases that allows access without requiring payment) media databases. The interface used to communicate with these databases is called an API (Application programming interface). This allows users access to their data in an easy and controlled way. The term API is used in several different contexts[REF1]. The databases which was needed needed to access uses an API integrated with HTTP, so to access different data there had to be sent HTTP-requests. When the database receives these requests and recognises it, it sends the data requested back. Most commonly the data is returned in either XML or JSON (JavaScript Object Notation).

We were unable to find any single database that offered all three of the featured media (those being; movies, video games and books) in this system, so we had to use three different. This was not a big problem since once the code to create and send a HTTP-request was done interpreting the answer was the only additional code needed. The following three databases were used:

- https://www.themoviedb.org/
- http://isbndb.com/
- http://thegamesdb.net/

TheMovieDb and isbndb both required an developer key to access their API. In both instances there was no problem to get the key. You simply had to register a user with the respective services.

Right these are simple used to get data for which we can use in our prototype, but it could also be a way to continually update our systems collection of media, so it stays up to date, and available to users.

4.4 Testing

4.5 Persistence

4.6 Summary

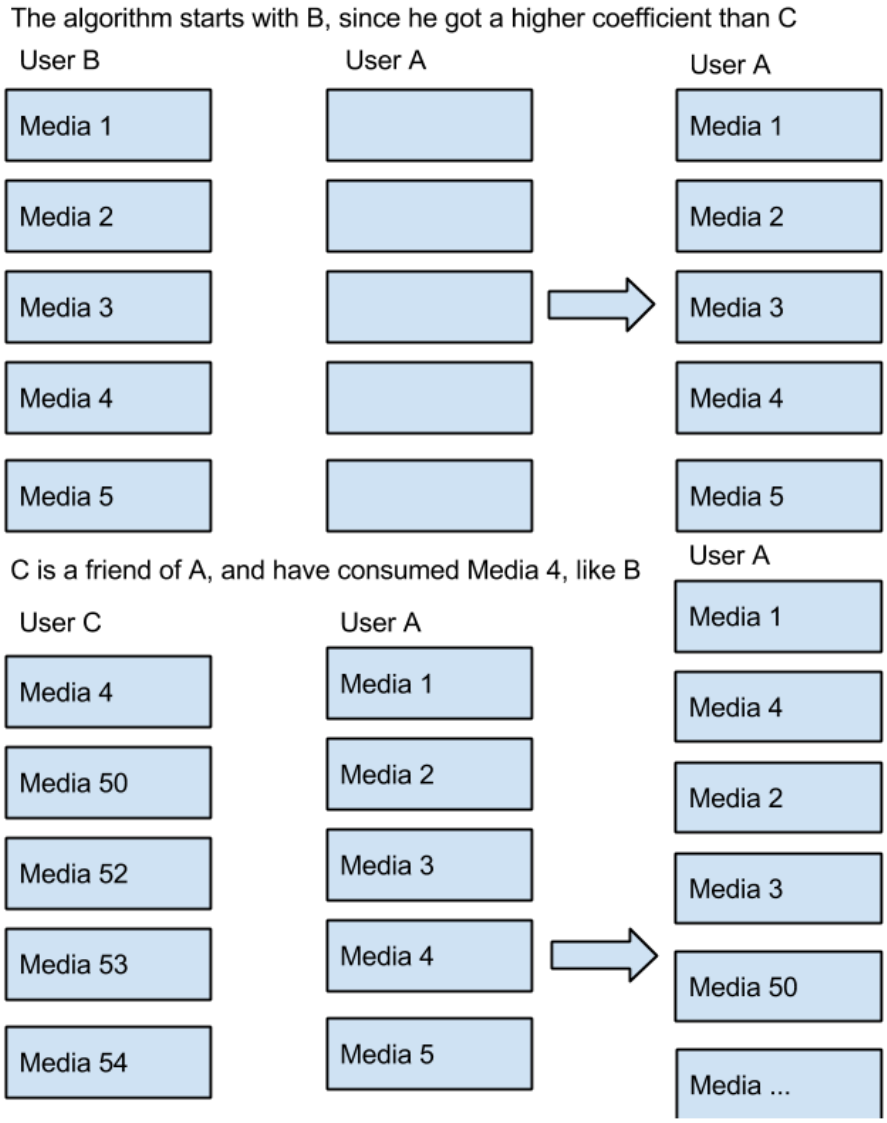

Figure 4.1: Example showing the collaborative selection process

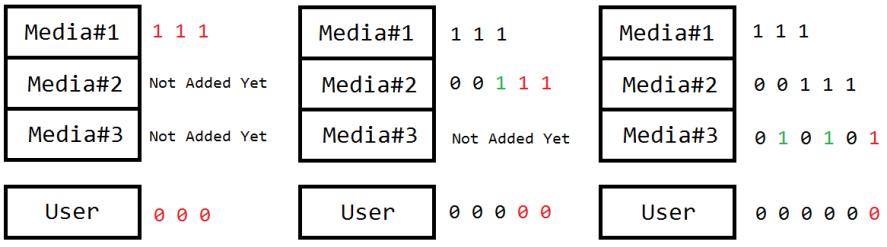

Figure 4.2: Vector updates as new media is added

Chapter 5

Conclusion

5.1 Perspective

5.2 Further Work

Chapter 6

Academic Report

This chapter is the academic report, which is going to focus on the courses which ran parallel with the project, and how the project have utilized the theories and methods that were presented. The chapter is going to focus on the system development and algorithm design courses.

6.1 System Development

This section is going to cover various subjects and activities regarding object-oriented analysis and design, applied in the context of this program. It is based on the systemdefinition 2.9.1. All the theory and methods applied during this section is based on the system development course, which we had running parallel with the project. The methods, theories, and figure composition is based on the textbook for the system development course, Object-Oriented Analysis & Design [**?**].

6.1.1 Problem Domain-Analysis

This is an analysis performed in relation to object-oriented analysis and design, and will focus on the problem-domain of this project. The problem-domain is the space which a system supervises and represents, in the form of an IT-solution. This analysis is done to describe the reality of the problem-domain, and to find connections between different entities, and their state.

Classes & Events

The problem-domain for this project is the media interested people, and the media which they wish to consume. It is all about the people involved in consuming different kinds of media, and recommending to other like-minded people, which will be called friends from now on. These people and media is the only which has to be represented in the IT-solution, and is therefore the classes for this problem-domain.

Events is the instant actions which can be initiated by, or affect, the different classes inside the problem-domain. The events for this problem-domain revolve around the consumption of media, the recommendation of media to other people, and people creating and breaking connections with their friends. See Table 6.1.

	Media Interested	**Media**
See Media	X	X
Recommend Media	X	X
"Get" Friend	X	
"Remove" Friend	X	

Table 6.1: Event Table

See Figure 6.1 for a class diagram, which shows connections between different classes, and the cardinality between them.

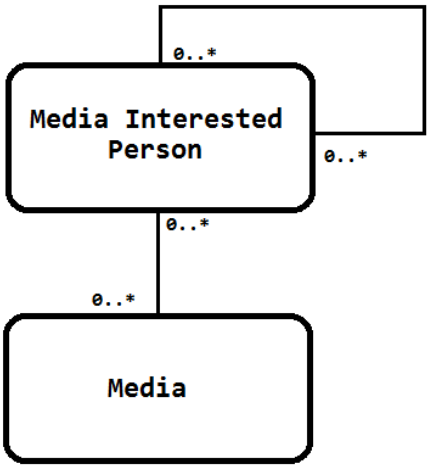

Figure 6.1: Classes in the problem-domain

The class diagram shows that the association between media interested people and media, and how multiple people can consume the same piece of media, and how various media can be consumed by a single person. An association from media interested people to itself represents the friends, which these people can make with other like-minded people. The association structure shows that there is no dependencies between these entities, and can exist without each other, which is the correct representation of this problem domains.

Event Traces

Next which has to be looked at is the event traces which every object, an instance of class, go through, as it is created. The event trace depict the events which the object of a class can be perform or be affected by during its existence, how it changes states inside the problem domain, and it may even leave the problem-domain completely. See figure 6.2.

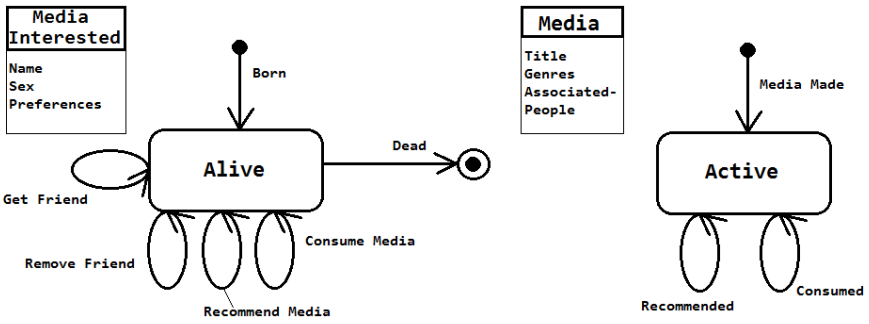

Figure 6.2: Event traces for the classes

These event traces shows the independence which these classes has in the problem-domain, much like the class diagram showed. It shows how all the events are iterative, and be be performed, as long as the object of the class exists. Also shown is various variables which is relevant in the context of a recommendation system, and the class itself. The course table can now be rewritten, to accommodate for the iterative nature of all events in this problem-domain. See table 6.2. The multiplication symbol indicates an event can happen multiple times for the instance of a class.

	Media Interested	Media
See Media	*	*
Recommend Media	*	*
"Get" Friend	*	
"Remove" Friend	*	

Table 6.2: The updated Event Table

6.1.2 Application-Domain Analysis

The application-domain is how various actors interact with the system, and through that controls, documents, and observes the problem-domain. Actors can be any exterior entity which is connected to the usage of the system, like people, but also includes external systems. These actors have various patterns of use available to be them, which lets them perform actions with the system.

Actors & Use Cases

For this system there is only two kind of actors: The 'user', and what has been named call the 'updater'.

The user is the general media interested person who uses the system to find media which he might be interested in, which is his purpose in the context of the system. He has a medialist, where he rates and indicates media, which he has consumed. He also have a friendlies where he makes connections with other people he know, or through their common interest. The unique characteristic of a user is their medialist and preferences, which can alter the recommendations process, generating different results.

The updater is an external system. Its purpose is to update the media which the system has available to it, with information coming from various sources. This is done to keep the system updated for newly released media, and made available for the users to add to their medialists. The characteristic of an updater is the media type which it keeps track of, and the sources which it extracts these data from.

The patterns of use is the various actions which is the actors can perform in the application-domain. The patterns of use for this is: See medialist, see friendlist, indicate/rate media, remove media, add friend, remove friend, update media collection. These have been made into a table, with actors, indicating which use case belongs to which actor. See Table 6.3.

Among these you could also include a use case called 'Generate recommen-

dations'. This depends on how this use is actually activated, as it may not be the most sensible choice to have this available to all users. The more sensible choice would be to activate it in certain intervals, to avoid a user abusing the action, possible slowing or crashing the system from overwork. In this it will be excluded.

	User	Updater
See Medialist	X	
See Friendlies	X	
Indicate/Rate Media	X	
Remove Media	X	
Add Friend	X	
Remove Friend	X	
Update Media Collection		X

Table 6.3: Use Case table

The following two figures shows patterns of use, indicate/rate media, and update media collections. These two were singled out for being the more unique patterns of use among those available, and the more crucial among them. See the first Figure 6.3 for the indicate/rate media use case, and Figure 6.4 for the update media collection use case.

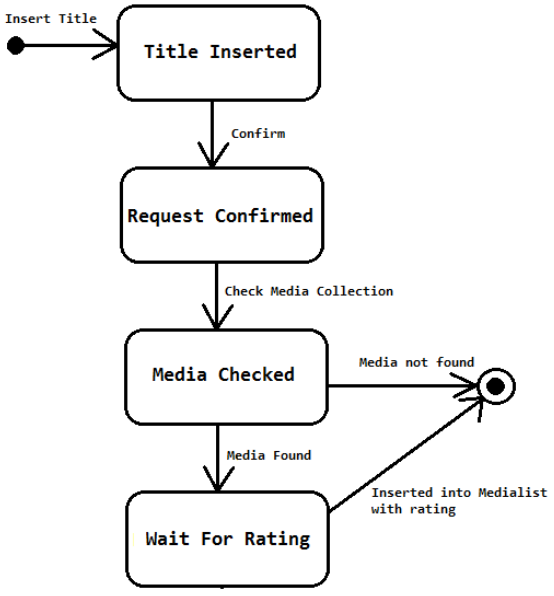

Figure 6.3: Diagram of the Indicate/Rate media use case

The indicate/rate media use case is initiated by the user, when the user in questions wants to add a piece of media, together with a rating, to their medialist. The user inserts the title of a media, or some other defining characteristic, which then have to be confirmed before the use case begins to check whether or not the piece of media exists in the systems collections. If the media was not in the collection, the use case will stop. If they media was, it will wait for the user to indicate his rating for the media, and will then finish the use case by saving it in the users medialist. Involved objects include the media interested person, and the media objects.

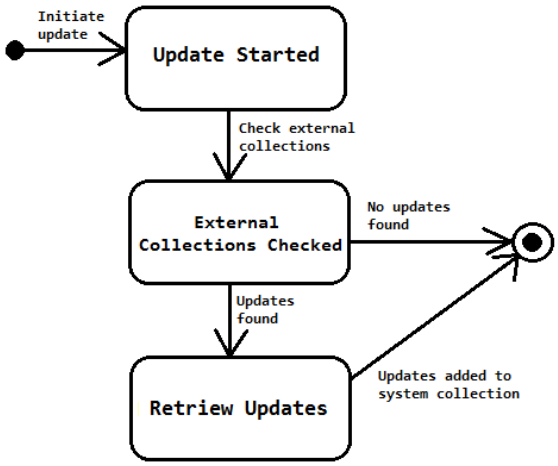

Figure 6.4: Diagram of the Update Media Collection use case

The update media collection is either initiated in a certain interval, or when new media content is detected available for the system. This case shows the former possibility. This use case starts by checking external collections upon being initiated, and will either end if no new media content was detected, or will begin retrieving said media content. After retrieving the media updated from the external collections, the use case will end by adding the media updates to the systems collections. Involved objects are several media objects.

Functions

Functions is the methods which is stands for most of the systems processing of data, and makes it possible for actors to communicate with the model of the system, through this function layer. Functions can include any of the patterns of use which has already been described, but there can also be additional functions, depending on the context of the system. For this system, there is

a function called 'Generate Recommendations', which was excluded from the patterns of use on this case, but is still very crucial for the system.

Functions are divided into four types: Update, read, calculate, and signal.

- Update Functions
  - Indicate/Rate media, remove media, add friend, remove friend, update media collection.
- Read Functions
  - See media, see friends, search for
- Signal & Calculate Functions
  - Generate recommendations

In this context, the generate recommendations function can easily go as both a signal and calculate function, as it has properties of both kinds. It processes data from the model, and generates recommendations based on this data, and then makes it viewable for the user, through the 'See media' function. The function itself is activated by a signal, like the user logging into their account, or a planned interval, generating the recommendations outside of the users consent.

These functions is then put into a table, indicating their function type, and determine the complexity of every function. See Table 6.4.

Function	**Complexity**	**Type**
Indicate/Rate Media	Medium	Update
Remove Media	Medium	Update
Add Friend	Simple	Update
Remove Friend	Simple	Update
Update Media Collection	Complex	Update
See Media	Simple	Read
See friends	Simple	Read
Search For	Medium	Read
Generate Recommendations	Very Complex	Signal/Calculation

Table 6.4: Functions with their complexity and type

6.1.3 Architectural Design

First of all, there is going to be set various criteria for the system, to indicate what general properties is the most crucial. With this, and what have previously been done, the component structure of the system is going to be defined, with the model component and function component.

Criteria

Based on the context of the system, and the system definition, criteria is going to set. This is done to determine which properties is the most important for the system, for the forthcoming development of the system, and limiting the work effort to what is the most crucial. See Table 6.5.

	Very	**Important**	**Less**	**Irrelevant**	**Easily Fulfilled**
Usable			X		
Secure				X	
Efficient			X		
Correct		X			
Reliable	X				
Maintainable			X		
Testable	X				
Flexible		X			
Comprehensive			X		
Reuseable				X	
Portable				X	
Interoperable		X			

Table 6.5: Criteria for the system

The properties indicated as very important, is 'Reliable' and 'Testable'. Reliable is the property that the system can perform its functions with precision, and reliably every time. This is important because the system is a recommendations system, which requires various algorithms to compare media and users, to generate these recommendations. This functionality should therefore work correctly. Testable is the property that makes it possible for the system to be tested, and ensuring that the systems runs as intended. This is crucial because, with time, the algorithm is most likely going to require various tweaks, and additions, to enhance the performance of the algorithm.

The other three important properties, correct, flexible, and interoperable, is also significant. Correct as an extension to the reliable property, again signifying that it is crucial the algorithm works as intended. Flexible, for the same reasons mentioned above, which is to make it possible to tweak to algorithm and the system to enhance performance. Interoperable, since the system is dependent on external sources to feed it new media data, so the system is up to date with newly released media content.

Components

Components is the parts which makes the system, and in this section, these sections is going to be defined, connected, and represented in a diagram. The components for this system is divided into four parts: User interface, functions, model, and external sources or servers. See Figure 6.5.

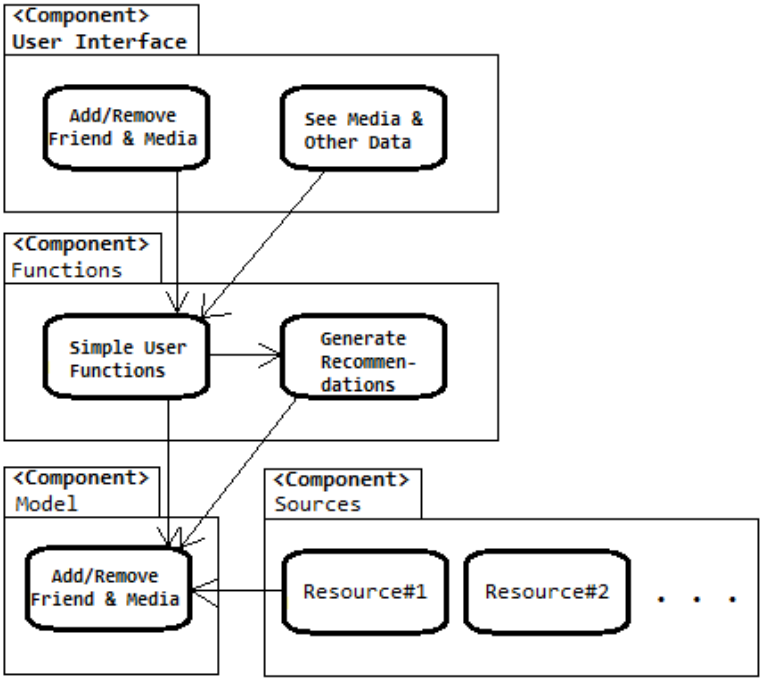

Figure 6.5: The components of the system, divided in layers

The user interface component contains the components which makes it possible for the user using the system, to get access and view data in the system. It has its add and remove component, for both media and friends, and its see component, which retrieves data from system, making it viewable for the user.

Both of these is connected to a component in the function components, which here is called 'Simple user functions'. This component contains the procedures that retrieves data from the model, and updates the model with data received from the user interface. This component is also connected to another component in the functions layer, called 'Generate Recommendations'. As previously explained, this procedure is indirectly connected to the user, activated through certain actions performed by the user, or by interval. Actions like the user logging into their account, or changing their preferences. Both of these components is connected to the model, which all concrete data regarding the users, all the media, and various other notable classes and objects. Right now it is represented by a component called DB, but will be updated to the forthcoming model component.

Using a server pattern, the last part of this component architecture is represented: External sources or servers. These are external servers, which has previously been described, and works together with the update media collections function. It is also through this function that these external sources is connected to the model of the component architecture.

6.1.4 Component Design

In the previous section, the overall structure of the components were looked at. In this part, the model component and function component is going to looked at more thorough, based on previous sections.

Model Component

The model component was the part which only had what was called the DB component, in the previous figure. What is included here is actually the original class structure, which is now going to be inserted into the component structure, after it has been updated. This update is based on the event table which was created in the problem area analysis, and with implementation in mind.

Following the theory, the class diagram has been updated to become the model component, taking into consideration private and shared events, and whether or they're iterative or singular. For example, the add/remove friend events become classes for themselves, and is a part of the class which it had a singular iterative connection with. The other add/remove for the media had a shared iterative connection, so it's more open how this is going to be showed on the new class diagram. In the Figure 6.6 you can see it as a new layer between

the media interested person and the media, and is called the medialist.

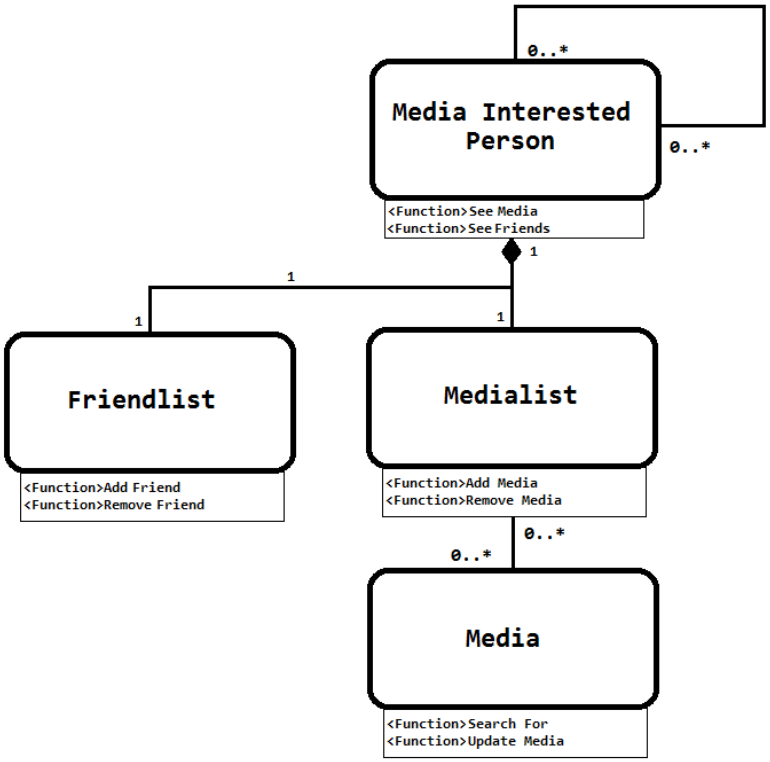

Figure 6.6: The model component with associated functions

Function Component

The function component is going to be made by going back to the functions which was defined in a previous section, and adding them to the updated component diagram. See Figure 6.7. Their placement is dependent on how many classes have a relations to the specific function. If a function only affects one type of class, then the function can be added directly to the class inside the model komponent. If it affects several, then it should be a separate component inside the function component, pointing the classes in the model component on which it has effect. This can also be seen in Figure 6.6.

For example, the 'Generate Recommendations', works not only on media in the system, but also on users, which it generates media recommendations for. It was already defined in the function component prior to this, this just cements its position there. There is things like add/remove media and friends,

which is limited to only one kind of class, and is therefore attached to that class in the model component.

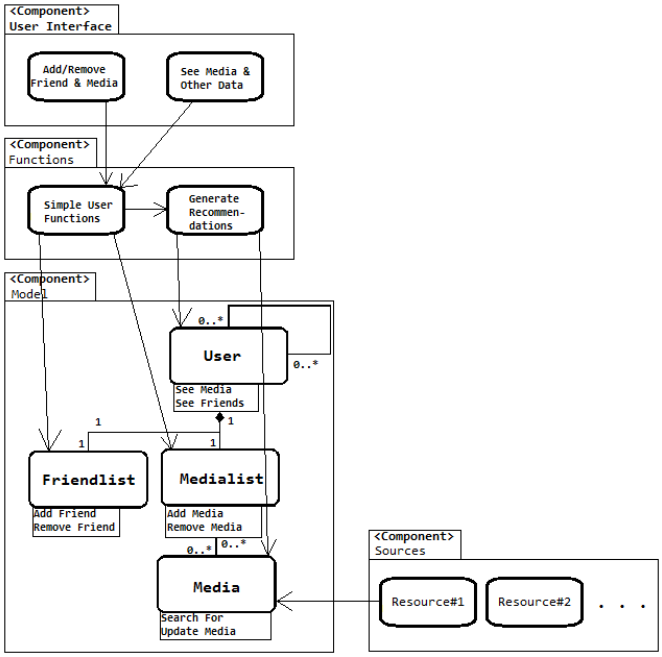

Figure 6.7: The full component structure

6.2 AlgorithmAnalysis

This section is going to cover some of the areas which was taught doing the Algorithm and Data Structures, which ran parallel with the project. The section is based on the actual system which was created during the project, using theory taught during the course applied to algorithms to determine their running time, and analysis of sorting algorithms which were used to implement the algorithm. The theory is based on the textbook for the algorithm and data structures course, Introduction To Algorithms [**?**].

6.2.1 Pseudo Code

Before the development of the recommendations algorithm, several pseudo code examples was created, to try and formalize how the collaborative side the algorithm should work, implementing the theory described in Section 3.1.1. In

Algorithm 1 and 2 you can see the two methods which does the bulk of the collaborative work.

IndexUsers(User U, Set of users A)
Data: User U, and all other users Set A
Result: a dictionary D of Set A with a coefficient
Initialization:
double k
Dictionary(User, double) D
foreach User in A **do**
  if User from A != U **then**
    Compare the user pair
    k <- CompareUserPair(U, user from A)
    D <- Add (User from A, k) to the dictionary
  end
end
Return D

Algorithm 1: The IndexUsers method

CompareUserPair(User U, User A)
Data: Main user U, and user A
Result: a coefficient for how similar user U and A are
Initialization:
Set X of integers
Set Y of integers
foreach Media in user U's medialist **do**
  if User A also have Media in their medialist **then**
    X <- Add U's rating of this media
    Y <- Add A's rating of this media
  end
end
Apply the Pearson measure to the two sets Return Pearson($\sum X$, $\sum Y$, $\sum XY$, $\sum X^2$, $\sum Y^2$, Size of X)

Algorithm 2: The CompareUserPair method

The content-based side of the algorithm works in a similar way, with the difference being the type of data which it works on. The collaborative part compares a users watched media with the watched media this user has in common with every other user in the system. The content-based part compares

a vector representation of the user with every vector representations of media, which is of similar length.

6.2.2 Asymptotic Running Time

The general running of this of the algorithm can be analysed using asymptotic notation, which can define both the worst and average-case running time of the algorithm. The running time of the loops in Algorithms 1 2 can be quite erratic, as it depends on how many media a specific user have added to their medialists. The if-structure within the second loop is also an erratic number of times during the algorithm, so it is hard to pinpoint the exact average-case running time.

The running time is described using Big-O notation, with expressions indicating variables inside the algorithm. Big-O notation uses the following symbols:

- $O(n)$, an upper bound, indicating the highest running time possible.
- $\Omega(n)$, a lower bound, indicating the lowest running time possible.
- $\Theta(n)$, a tight bound, indicating both of the previous two running-times.

Fortunately, The worst-case running time is relatively easy to find. The worst-case would occur if every user have the exact same media indicated on their medialists. The worst-case running time is then all users in the system, except the user which the algorithm is generating recommendations for, times the subset of media which is available from all the users indicated media in their medialists. The upper bound of the algorithm can then be defined as:

$$m(n-1) = mn - m = O(mn)$$

Where n is all users, and m is the available media. As mentioned above, the average-case running time is harder to precisely find. What can be derived from the algorithm is that no matter what, all the users in the system will be processed, so the lower bound of the algorithm can be defined as $\Omega(n)$.

The content-based part is similar, but is much more stable in its running time. The user is compared to all media in the system by comparing their vector representations, where the running time is dictated by the size of the medias vector. The algorithm will for each media go through the size of their vector representations, and because of the way the media vectors is constructed,

as desribed in Section [REF], the algorithm will run for every media times the longest media vector divided in half.

$$\frac{mn}{2} = \Theta(mn)$$

Where m is all media, and n is the size of the longest media vector representation. The running-time is tightly bound, and will never deviate from this running time.

6.2.3 Sorting

The recommendation algorithm which was created together with this project also utilizes a sorting algorithm to some degree, which is the OrderBy method made available by the .NET framework. OrderBy can be invoked through query expressions or lambda expressions on collections, and take as parameters elements, or properties from those elements, and a logical predicate for how the sorting should be done. **[?]**

Quicksort

The actual sorting algorithm which OrderBy implements is a stable QuickSort algorithm, which maintain the relative order of elements which is equal in accordance with the predicate. OrderBy is used during the recommendations algorithm, both in the collaborative and content-based side of the algorithm, on the coefficient generated through both processes. The algorithm actually used the OrderByDescending version, which does the same thing, except in reserve order. What is important is the coefficients, which should be sorted so they highest coefficients is placed in the front.

Quicksort has a worst-case running time of $O(n^2)$, which is quite slow, but it is also very rare that quicksort encounters its worst-case scenario. Its average-case is much more likely, increasing its likelihood with the size of the collections quicksort is applied to, and has a running time of $O(nlgn)$.

The most important part of the quicksort algorithm is the partition part, which divides the collection into two parts, based on a pivot element from the collection, which depends on the implementation of the algorithm. See Algorithm 3 for the pseudo code.

It shows the partition method which takes the set A, together with references to the lowest and the highest elements of the set p and r, respectively. Partition returns another reference q that points to the pivot chosen by the algorithm. The state of set A now is that all elements less than the element q

```
Quicksort( A, p, r )
Initialization:
int *q
if p < r then
    q <- Partition( A, p, r )
    Quicksort( A, p, q − 1 )
    Quicksort( A, q + 1, r )
end
```

Algorithm 3: The Quicksort algorithm

points to is now on one side of q, while the elements which is higher is on the other side. Quicksort then runs itself recursively with the two new sets divided by q, where q is now sorted. The worst-case running time occurs if the chosen pivot element is also the highest element in the set every time quicksort is run. See Algorithm 4 for the partition pseudo code.

```
Partition( A, p ,r )
Initialization:
int x
int i <- p-1
for j = p to r-1 do
    if A[ j ] >= x then
        i <- i+1
        Exchange A[ i ] with A[ j ]
    end
end
Exchange A[ i+1 ] with A[ r ]
return i+1
```

Algorithm 4: The Partition function for Quicksort

If the chosen pivot element is always the highest, the returned reference q will always be the last element in the set. This means that the recursive calls following the partition will run in respectively $n-1$ and 0 time. If this happens throughout the whole course of the algorithm, the running time will be an arithmetic series**[?]**, which is $O(n^2)$.

$$n + (n-1) + (n-2) + ... + 2 + 1 + 0 = \frac{n(n+1)}{2} = \frac{n^2+n}{2} = O(n^2)$$

The average-case for quicksort, which is also the best-case, is whenever partition split the set so neither of the two subsets generates is empty. In this case the running time will always be $O(nlgn)$. This means that the recommendations algorithm will most likely always sort at this efficient running time, since the worst-case is rare.