RECMIX

A Value Based Recommender for Music

Group:	**Student ID:**
Kenzo Boudier	5333024
Lucas Fatas	ID
Nathaniel DeLeeuw	ID
Daniel Puente Barajas	ID
Diego Verone	ID

Date: 4 June 2022

Course: CSE2000 Software Project

Teaching Assistant: Bianca Cosma

Table of Contents

NEW:

Table of Contents

OLD:

[Preface]

This report was written by a group of five enthusiastic computer science students at Delft University of Technology. For the final quarter of our second year, we were given the task to create a web application. The website is an experiment to test the efficiency of value based recommender systems and is to be made public at the end of the project.

During the composition of this report, we assumed the reader to know some basic knowledge of the following: the application Spotify, and basic computational concepts in Java and Python. An index is at the end of the report to aid readers with any technical vocabulary

Readers that are curious about the ….

We would like to show our gratitude to our T.A. Bianca Cosme for her constructive advice and reliability throughout the project. Additionally we would like to thank our technical writing teacher INSERT NAME for his aid and teaching us to enhance our writing. Furthermore, we want to thank our client NAME and her supervisor NAME for being open to our opinion and very understanding during the entirety of the project

Delft, 06 June 2022
Boudier Kenzo
Lucas Fatas
Nathaniel DeLeeuw
Daniel Puente Barajas

Diego Vieron

1 Introduction to our Project

This chapter will introduce our software project and give some background information on it. Section 1.1 will give a proper presentation of the project and state the aim of this report. Next, in section 1.2, we will discuss the history of recommendation systems as it is a dominant part of this project. In the following section (1.3) we will summarize the project.

1.1 Presentation of the Project

Exploring the potential of using value based recommendation algorithms is a possible key to create a new generation of more performance recommendation systems. Recommender systems are intelligent systems that recommend to a customer items based on his taste, interest and personality. If such a system is not performant, this is disadvantageous for both the clients and provider as the client cannot find any items he could like and the provider loses revenu. PhD student Sandy Manolios is in the process of determining the importance of values in a recommendation system .

The objective of this report is to see the process and steps taken to create a values based and a personality based recommendation system for music. Sandy Manolios asked our group to create an online website that matches people's musical taste in order to determine the importance of values in recommendation systems.In order to achieve the desired results, participants will take a test to determine their values and be matched to a user with similar values. Then the participant will rate the music of the similar user. A crucial factor in this experiment is to see whether matched users like or not the music of the other user. The limitation of this experiment is the sample size and the different music tastes.

The report will be presented in the following structure. Chapter 2 will furnish an extensive explanation on the matching process employed on the website. The different tools employed during this research will be located in chapter 3. The outcome of this experience will be in chapter 4. Chapter 5 will build on the outcome found in chapter 4 and break it down in order to see a more detailed result. Chapter 6 will restate the important factors and conclude this document

1.2 History of Recommender Systems

1.3 Summary

2 Problem Analysis

The objective of this chapter is to analyze the problem laid out for us, the people affected by it and the consequences it will have. In section 1.1 we will study the context of the problem, as well as an explanation of the psychological methods we will use to investigate human connections. Next, we will discuss who will be affected by our application and how they will be affected in section 1.2. In the following section 1.3 we will develop a list of project statements, dividing the task into smaller and more achievable goals, and after this we will discuss our goals in the next section 1.4. Next, we will take a look at the inspiration we took from existing apps in section 1.5, and possible use cases of our app by the different users that we will handle in section 1.6. Finally, we will show how our data will be structured in section 1.7.

2.1 Context of the problem

People can usually be classified into different personalities and values. We intend to explore those classifications and check if there is any correlation with the music preferences in it. In the following section we are going to explore humans according to both their personality and values. Since these two concepts can be abstract and easily confused, we will provide some explanation for them, as well as an overview of the classification methods we will use, in this section:

2.1.1 Personality-based recommendations

Personality is generally defined as the group of all the characteristics and traits that make one person different from another. Such traits are, for example, imagination, anxiety, or dutifulness.
These traits can be divided into the six Hexaco categories of personality, according to modern psychology. We will use this division to quantify each participant's personality:

- Honesty-Humility: in this category people who avoid manipulating others, do not break rules, and lead a humble life have a high score.
- Emotionality: related to fear of physical dangers, anxiety at life, need for emotional support and empathy.
- Extraversion: people who lead high extraversion scores are usually linked to social competence, enjoyment within social gatherings and interactions, leadership within a group, enthusiasm, energy and self-esteem.
- Agreeableness: agreeable people are forgiving, lenient in judging others, can compromise and cooperate and are mild-tempered.
- Conscientiousness: this personality category is the one that covers time and physical organization, organized work, accuracy and perfection in tasks, and careful decision-making.
- Openness to Experience: people with a high score in this category enjoy the beauty of art and nature, like to research different topics, are imaginative, and become interested in different ideas and people.

These 6 dimensions include all the important traits of human nature, and are used as the defining factors. They are universally accepted and recognized by psychologists all around the globe to determine human personality, and it is the metric we will use for our personality evaluation.
Note that these dimensions range from positive and negative values to the specified characteristic, meaning that someone can be either very extraverted, meaning he/she is very social and open, or not at all extraverted, meaning he/she is shyer and more introverted. So this classification gives us insight in both ways of the spectrum for each of the outlined categories.
Note as well that there are a huge number of human traits, and all this division is able to classify all these

traits into one of the specified categories.
We will use this division of the human psyche as the first part of this project: classify each participant by assigning a value for each of the categories, match participants with similar personalities and suggest music from this match.

2.1.2 Value-based recommendation

Values act as a guiding principle in our life. They are strongly associated with our morals and judgments. They are deeply immersed in the decision-making process and prepare individuals to act in a certain way depending on the consequences of those actions. There are several values that condition how we interact with our environment, but for this project we are going to take a look at Schwartz's Theory of basic values. This theory establishes 10 main values to take into account when defining a person, that are derived from the three main needs for human existence: biological needs, social interaction, and group survival:

- Universalism: motivational goal of understanding and protecting the welfare of humans and nature.
- Benevolence: protecting and enhancing the welfare of human beings whom the individual is in contact with.
- Tradition: acceptance of ideologies and beliefs that one's tradition provides.
- Restrain: restriction of one's harmful impulses
- Safety: underlying motivational goal of harmony and safety of society, oneself and other
- Power: It involves status, prestige and power over other individuals and resources.
- Achievement: succeeding by showing competence according to society's standards
- Hedonism: gratifying one's own impulses.
- Stimulation: seeking excitement and thrill in life.
- Self-direction: defining and exploring one's own way.

These values are organized in a circular way according to the compatibility of such values:

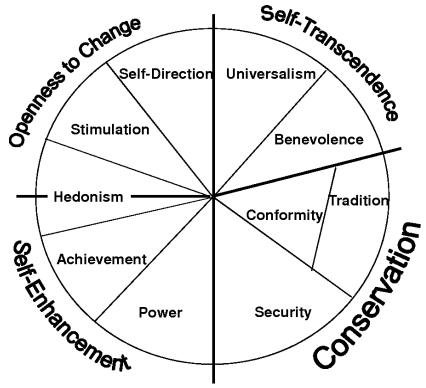


Figure 1: Schwartz's Theory of basic values

We will use the specified model for our project as well. After gathering data about participants' values, we will match participants with similar values and recommend songs based on that.

2.1.3 Random recommendation

Lastly, we will implement a random recommender in order to have a control group, which will be used as a

benchmark or a point of comparison against which the other recommendations will be compared to assess the validity of the thesis.

In conclusion, we will implement these three kinds of recommendations in order to see if people with similar personalities have a more similar taste in music and check if a recommender system that uses personality and values as input for recommendation would work. We will do that by implementing the experiment in two batches: in the first batch we will just gather the personality, values and music of each participant (by making use of their Spotify account) and in the second batch the participants will as well submit their personality and get song recommendations based on that, according to the three ways described above.

We can acquire the personality and values of each participant with two simple questionnaires, and we will retrieve the participant's music taste by making use of the Spotify API and fetching the top songs of the participants. The second batch participants will then get the recommendations and provide feedback on how good such recommendations were, by answering meaningful questions and rating them, and providing optional open feedback and ratings for each song.

2.2 Stakeholders

In this chapter we will discuss the stakeholders of the project. A stakeholder is a person or organization who either influences a system's requirements or is impacted by that system. We take these stakeholders into account at the time of the requirement elicitation, so that we make sure that our project is satisfactory to the people affected by it.

There are both direct and indirect stakeholders involved in this project. The difference lies in the kind of effect that our application will have. Direct stakeholders are involved with the day-to-day activities of a project, and directly affected by the final project. Indirect stakeholders pay attention to the finished project outcome rather than the process of completing it, and are not necessarily affected by the product development.

We can then see that the direct stakeholders will be: our client, the researcher, Sandy Manolios; the Teaching Assistant assigned to our group, Bianca Cosma; the coach for the project, Gosia Migut; and us as the developers. All these people influence the daily progress of the project and are involved in the whole development process, thus they can all be classified as such.

As indirect stakeholders we can see that TU Delft will be affected by it, since one of its Ph.D. students' thesis will be influenced by this and it potentially could prove useful in the study of personalities and Cognitive Science. We also have to take into account the current privacy and internet safety laws in order not to negatively impact the participants of our website, so we have to take the participants of our webpage as passive stakeholders, since they will be influenced by the final product; and the Dutch and EU government, given that they will influence the way we handle the product.
If after the thesis publication, personality-values recommendation starts being implemented as a recommender system by big corporations, music artists should also be considered as passive stakeholders, since the number of listeners they receive on certain songs may either increase or decrease based on the recommendations.

2.3 Problem Statement

This section aims to divide the problem provided to us into more specific and achievable goals, which will provide an overview of the issues that will be tackled in the development of our project:

2.3.1 Gather Data

Our group will have to conceive a website from scratch. After researching for websites, we realized that there is no similar web application available for us to build upon. This website would calculate the personality and value of a given participant based on a questionnaire. This data would be then later used for matching participants and their music taste. The challenge is that our client must be able to easily retrieve the data and manipulate it.
The questions from which the questionnaire is composed are entirely provided by Sandy Manolios and we will display them exactly how she sent them in order not to compromise the participant analysis process.

2.3.2 Matching Participants

The website should be able to match participants in three different ways; randomly which will serve as the control group, based on personality to see the correlation between personality and taste in music and lastly based on a participant's value. The matching process for personality and value will employ the nearest neighbor method to decide on a pair of individuals

2.3.3 Recommended Songs Ratings

The greatest objective of our website is the retrieval of participants' opinions on a set of songs recommended through similar participants personalities and values . The client needs the data from these answers for the thesis

2.3.4 Satisfying Stakeholders

During this project our main stakeholder will be our client, since a product that satisfies her needs is what we're working for. Although as we are collecting data we do need to conform ourselves to the TU Delft, Dutch and European regulation. Furthermore, we must also take into account that this project may change how recommender systems behave, impacting users and artists that use applications that implement this recommendation system.
Our group has to find a way to encompass everyone's needs and satisfy them.

2.4 Project Goals

After all the analysis conducted in the previous sections, where we have analyzed our responsibilities and the problem to be solved, we have distilled our final project objective into 4 distinct sub goals specified below:

2.4.1 Making a scalable frontend

We want our application to work both on computers and phones/tablets. This will allow participants to take part in the questionnaire on every device without any constraint.
To accomplish this goal we're going to use the Tailwind library, which has built-in support for scalable styling and will allow our application to be handled by multiple different devices.

2.4.2 Using Spotify API

Through the use of Spotify's API we can allow participants to listen to a 30 seconds snippet of the song they're being asked about. Allowing participants to hear 30 seconds of the song they're being questioned to evaluate allows for a much better experience during the questionnaire. This will also allow for better results

from which the client will be able to receive better information. The Spotify API also allows us to fetch the participants' listening history, essential in the recommendation process.

2.4.3 Easily Retrieve the Data

After having collected all the necessary data from the participants that answered the questionnaires, we want to create an easy-to-use platform such that the client can directly work on the collected information from there, without having to deal with stored data manually herself.
This process will facilitate the client approach to the data and the way it is handled and examined.

2.4.4 Monitor the Experiment

We need to create allow the researcher monitor the experiment through the use of parameter modification (the similarity metric between users, etc) and an overview of the whole process, including control over the research by showing the size of the answers and providing a way to switch to phase 2 of the experiment with the second batch.

2.5 Inspiration from Existing Products

We will now take a look at how we got started in our application in this chapter. We did that by taking a look at similar products and taking notes and inspiration from them on how to organize our app and how to create a good User Interface.

Our web application is partly a personality test/form and partly a music recommendation system where we ask for feedback. This chapter wants to show the kind of inspiration that we took from already existing services that we looked at in order to get a better idea about the implementation of our project. Since our app has such a clear distinction of requirements, as it is both a questionnaire and a recommendation system, the inspiration sessions were divided in such both parts, where we took a look at products first as a quiz and then as a feedback website.

For the first part, there exists a profusion of personality test websites available on the internet. There are also plenty of websites where we can create questionnaires. The most known one is Google Forms, but open source alternatives, like Budibase or Cryptpad, also exist. They helped us gain some inspiration regarding User Interface and how to structure the questions, showing us what ways are good to implement the front-end part of our project to make the app appealing. For the second part, such websites also helped us with ideas on how to handle the feedback after getting the recommendations, as well on how to structure the songs and feedback fields in order to make it appealing to the user.

2.6 Use cases

This section discusses examples on how the users will use our application. There are 3 different types of users that will make use of our web application: participant of batch 1, participant of batch 2 and the researcher. By Using use cases, we are pointing out their journey through our site. Based on these use cases, we can easily create functional requirements for the MoSCoW method.

Participant of batch 1:

- The participant wants to access our site thanks to a link provided in the Prolific site.
- The participant needs to log in into their Spotify account.
- The participant needs to be able to answer questions according to personality and values, split

into multiple pages.
- Our system needs to calculate the values and the personality of the participant based on the answers and store it with his/her Spotify top songs.

Participant of batch 2:
- The participant wants to access our site thanks to a link provided in the Prolific site.
- The participant needs to log in into their Spotify account.
- The participant needs to be able to answer questions according to personality and values, split into multiple pages.
- Our system needs to calculate the values and the personality of the participant based on the answers.
- Our system matches the participant to participants of batch one with the most similar values, most similar personality and one random participant and retrieve from the database their songs.
- The participant of batch 2 listens to the songs of the matched participants of batch 1 and gives their opinion on it.

Researcher:
- The researcher wants to log in to the dashboard if she wants to change the parameters of the experiment.
- The researcher wants to retrieve the values, personality and songs of the first batch (in a CSV file).
- The researcher wants to adjust the parameters of the study (Examples: similarity metric, playlist sizes).
- The researcher wants to see the progress of the study.
- The researcher wants to retrieve the ratings of the recommended songs of the second batch participants (in a CSV file).

2.7 The Data Structure

Retrieving and processing data plays an integral role in our project which is why clearly identifying the different data structures and how they are related is crucial. This chapter will focus on data representation and discussion on how to store it.

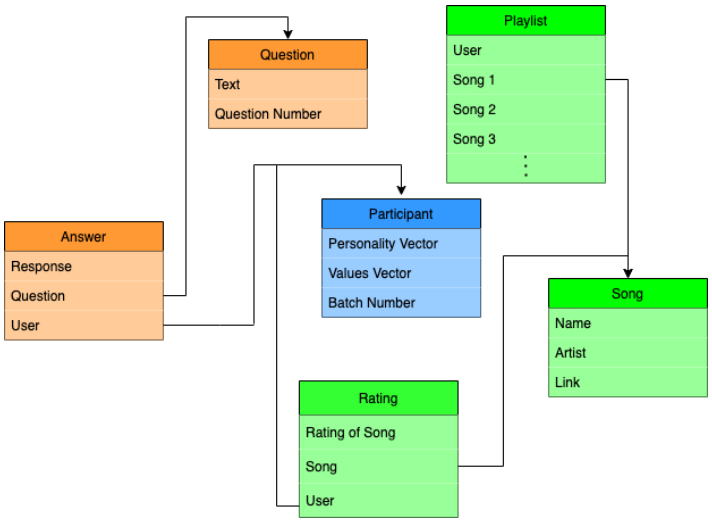


Figure 2: High level overview of data structures

1. **Participant:** The participants will have a personality vector and value vector based on their answers to the questionnaire. They are divided into two batches and will also have Spotify music data associated with it: Batch 1 participants will have its listening history attached to the vectors, and batch 2 participants will have the recommendations provided to them.
2. **Question:** Question contains a text with the question and a number to identify it. There will be three types of questions: those related to the personality part, those related to the value part. They will be provided by our client in order to effectively calculate the vectors and provide a faithful representation of both personality models.
3. **Answer:** An answer stores the response a participant gives to a question. Each answer to each question will be stored individually, and it will be structured ranging from one to five, in order to convey different levels of agreement when answering the questions -1 meaning lack of agreement and 5 meaning total agreement-.
4. **Song:** The songs retrieved from the participant's Spotify will have a name, artist and link stored.
5. **Playlist:** A collection of songs, represented in a list form. Each participant from batch 1 will have a playlist representing the listening history, and each participant from batch 2 will have three playlists attached, representing the three different kinds of recommendations provided to him.
6. **Rating:** The participant's rating of a song that was recommended to him. Each rating for each song will be stored individually.

3 Requirements

This chapter will discuss the division of the task provided to us into a list of requirements, following the MoSCoW method, that will help us in the development of our project and will divide the work into subproblems that will be solved through several sprints. Section 3.1 will discuss how we found those requirements and which methods we made use of, and section 3.2 will show the list of requirements that was composed after such elicitation methods were used.

3.1 Requirement Elicitation

The first step in our requirement elicitation in this project was a one to one interview with our client, our only direct stakeholder (2.1.1). Then the following section (2.1.2) will talk about our brainstorming session that followed the interview. The last section (2.1.3) will explain the MoSCoW method used to prioritize the requirements of the web application.

3.1.1 Interview with our Principal Stakeholder

We had a first meeting with our stakeholder, Sandy Manolios, and her mentor Cynthia Liem before the project began. It was an introduction to the problem given to us and an overview of her PhD Thesis. It helped us to have a general overview over the project and necessary context about the subject of her thesis.

Thursday 21st April 2022 we had our first official interview, where, based on the knowledge we had from the previous meeting, we were able to ask questions about uncertainties we had and also more technical parts of the project. We then constructed a first draft of the requirements.

3.1.2 Brainstorming Session with the Software Group

We created the first draft of the requirements in a brainstorming session between ourselves. After

researching together similar products to identify features that our product will need we added to and refined our first draft of the requirements list. During this session we were in communication with our TA to get feedback on our requirements and with our client to clarify certain aspects of the project.

3.1.3 MoSCoW Method

The purpose of the MoSCoW method is to prioritize requirements. Requirements are either one of the following: functional or non-functional:

- Functional requirements are requirements that explain how the application is supposed to behave and which features it is supposed to have. These requirements are decided with the client, Sandy Monlios in our case.
- Non-functional requirements are requirements linked to the performance of the application. These requirements are quality constraints to ensure a good product. They are important but not as important as functional requirements

While interviewing our client, we made sure to ask about the importance and priority of every requirement, to divide them easily into the categories of the MoSCoW method. While programming, we will base the sprints of the scrum methodology on these priorities:

- The must-haves are the base requirements of our project. Without the must-haves the base goal of the project cannot be achieved and that is why they are going to be implemented as first.
- The should-haves are in general to improve the participant experience while being on the site, with modification possibilities and design choices.
- The could-haves are mostly requirements to ease the interactions between our client and the web application. Facilitating the viewing and retrieving of data is one of these aspects.
- The wont-haves are requirements that will not be implemented in the period of our project, but can possibly be implemented later on.

3.2 Requirement List

Below we have listed out the requirements according to the MoSCoW method of the project. We decided to use the MoSCoW method as it is great for high level requirements and easily allows for requirements to be changed or added [6]. These two aspects of the MoSCoW method are both critical to this project. The non-functional requirements will be discussed (3.2.1) followed by the functional requirements(3.2.2). The “Must Have” requirements (3.2.2.1) will go over the most important features of our application. The following section (3.2.2.2) is the “Should have” requirements that are important but not critical for the application to work. Then we will show the “Could Have” requirements (3.2.2.3). Lastly, we will discuss some of the requirements that this project will not have (3.2.2.4).

3.2.1 Non-Functional Requirements

- Data should be stored and handled according to the GDPR.
- The website must be compatible with Chrome, Firefox and Safari
- The login credentials of the researcher should be safely stored, in order to access the dashboard.

3.2.2 Functional Requirements

3.2.2.1 Must Have

- The participant must be able to access our software through a website.
- The participant must be able to read and accept a consent form related to sharing their Spotify data.
- The participant must be able to answer and modify the questions from the questionnaire
- The participant must be matched with other participants based on the responses to the questionnaires.

- The participant's answers to the questionnaire must be converted to a vector representing personality and a vector representing values.
- The participant must have the personality vector, value vector and top songs stored in a persistent way.
- The participant must be able to rate the playlist of recommended songs and have the option to rate the individual songs.
- The participant must be able to have the option to give open feedback about the recommended music
- The participant must be able to connect their Spotify account so that their top songs can be retrieved
- The researcher must be able to download participant data into a CSV file.
- The researcher must have a dashboard to adjust parameters for the study (Examples: similarity metric, current batch)

3.2.2.2 Should Have

- The participant should be able to go back at any moment to the previous page
- The participant should get the music recommendations in 30 second snippets
- The researcher should be able to see the progress of the study in the dashboard
- The researcher should be able to retrieve all the answers to the questionnaires

3.2.2.3 Could Have

- The participant should be able to access and use the website from their phone
- The participant could be able to like or add recommendations to his account
- The researcher could be able to see if a participant added a recommended song to his playlist
- The researcher could have an interface to sort and filter data of participants in the dashboard

3.2.2.4 Won't Have

- The project could be structured in a way that it will be reusable for other kinds of research in the future.
- A change password button on the dashboard

4 Design

The design chapter introduces a high-level overview of the structure, languages, libraries and technologies used to develop the final application. In section we'll discuss the architecture used to define the structure of the project. Section 3.1.1 discusses the back-end server, while the web client and database are discussed in 3.1.2 and 3.1.3 respectively. Afterwards we'll talk about everything that concerns the hosting in 3.2. Finally, we'll discuss the different testing methodologies that were used in 3.3.1, the front-end testing in 3.3.2 and the back-end testing in 3.3.3

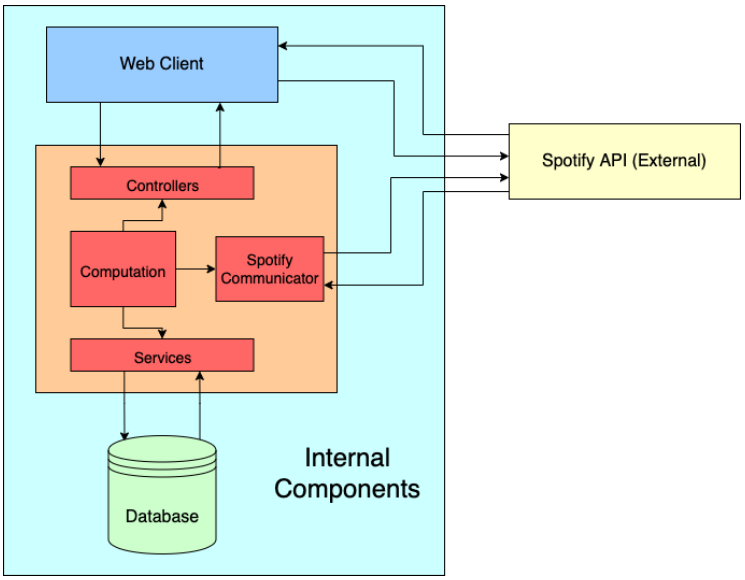
4.1 Architecture

After analyzing all the relevant requirements for the application, we concluded that the most adequate architecture would have been to divide the project into front-end and back-end.

Since the users will interact with the platform through a browser, we figured that treating it as a web-application would benefit the development process.

4.1.1 Back-End Server

The Back-End Server is implemented using the Python programming language and the Flask web framework. The back-end server covers the following responsibilities. Create all the endpoints that will be accessed from the front-end to exchange information. Process the information received and if necessary, send a response back to the web-client. Interact with the database by either storing or retrieving the processed information. Manage the Spotify API to retrieve users' tokens after login, fetching their top songs and preparing the data in order to send it to the client.



4.1.2 Web Client

The web client is the actual application accessed by the final user. The front-end will be entirely written using the JavaScript programming language. The main frameworks used to develop it are React.js and Tailwind for the styling.

The front-end will be split up into two macro sections: first batch and second batch.
The first batch will include initially a Spotify login, after which, if successful, the participant will reach the questions section. In there he/she will be able to answer questions. After the last questions the participant will submit all the questions that will be received in the backend and stored in the database with the associated participant id. The second batch consists of participants answering the same questions as the first batch, and then, based on the given answers, specific songs will be suggested to the participant which will have to rate them based on their personal preference.

4.1.3 Database

The data will be managed using a MySQL Database. All the communication with the database is handled by the back-end server, this provides a layer of security by which the user can't directly access sensitive information. The choice of a relational database instead of a NoSQL one was made since the data that is stored in it is structured and we didn't need any of the document-oriented functionalities that NoSQL databases provide.

4.2 Hosting

Since this application will be accessed through a browser, we needed to think about the hosting part of all services, front-end, back-end and database.
Initially we considered multiple alternatives like AWS (Amazon Web Services) and the Google Cloud platform. After a discussion with the project supervisor, we decided to host everything on the university's servers, since it would've been easy to access it both for us and for Sandy after we deployed the application. The hosting supports an SSL certificate and it is able to run all the services of the application.
It is worth mentioning that hosting everything on the same server simplified both the development and deployment process.

4.3 Testing

To ensure that our software is working as intended, extensive testing will be required for all aspects of the software. Small mistakes in our software could have detrimental effects on the study. An area of the software where this is especially true is when matching the participants. If the matching software mismatched participants, this would not be visible while using the software but would ruin the study. To ensure this does not happen, we will need a varying set of tests for the software.

4.3.1 Testing typologies used

To make sure that our application does not contain any errors or bugs, we are using different types of testing. On both the back-end and the front-end we are implementing unit and integration tests. These tests ensure that all the individual components are working and also the interactions between them.
On top of the previous tests, we are also doing API-testing on the back-end to make sure that the communication with the Spotify API goes as intended.
At last, to test the general working of our application, we also make use of production testing and user testing.

4.3.2 Front-end

For the front-end testing, we use the Jest testing framework. Jest provides a complete, well documented API for testing front-end code. It is especially useful since the use of frameworks like React can introduce many problems and bugs even with front-end development.
Jest includes useful functionalities like code coverage and mocking, these tools allow us to create a test-suite that is as complete as possible in order to eliminate most of the bugs before deployment.

4.3.3 Back-end

The testing tool being used for the back-end is Pytest. Ensuring that testing is done correctly and thoroughly on the back-end is crucial. The computation of the results of the questionnaires and the matching process both occur on the back-end. A small error here could severely affect the results of the study. Extensive unit testing is performed on all the different components of these computations. We also incorporate integration and database testing into our testing suite.

5 Ethics & Values

In this part we will discuss the potential ethics issues that our Web application may entail and the values it may impede. Section 4.1 will discuss the ethics categories that are relevant and the ones that are not to this project. Then the following section (4.2) will go in depth on the ethical problems of the application. Lastly section 4.3 will explain the solutions found to mitigate the problems mentioned in the previous section

5.1 The five ethical categories

The following section will briefly explain the 5 ethical problems given by TuDelft Write. Additionally, the relevance of each category in the scope of this project will be discussed.

5.1.1 Privacy and Data Security

Privacy and data security deals with the protection of user sensitive data. Sensitive data is information that could be held hostage for a ransom or abused in any way. The application designed does not store or require sensitive data therefore, this ethics problem is not relevant. One potential problem is data alteration and/or destruction this is why the data will be secured from these threats.

5.1.2 Shaping users

The ethics category Shaping users handles all the effects a machine/application can have on a user. This ethical problem is present in our project as users will see their personlity and values according to a test they take. The results of the tests could lead to a reinforcement of a faulty perception of a user's personality and values that could have severe effects on identity and self-beliefs.

5.1.3 Bias and Discrimination

Bias and Discrimination covers all the problems with the bias a machine/application can contain. This ethical problem is the main concern of this project as the data can be biased due to sampling. This ethical problem will be seen in more depth in section 4.2.

5.1.4 AI and Justification

AI and Justifications is the problem of who is accountable for bad decisions made by AIs. This problem is absent in our project. There are 3 reasons why AI and Justification is not present in this project:

- The lack of an AI in the project
- The matching algorithm being a “white-box” and therefore modifiable in case of unjust results
- No “decisions” are being made after the matching process.

5.1.5 Designing Ethically

The final ethics category Designing Ethically is based on the concept of doing “good” by design. Good meaning respecting values such as liberty. This ethical problem is one our project takes into account as the application will be used by a variety of users therefore the project needs to account human values, norms and morals. However we will not go in more depth as it is not one of the main ethical problems in our project.

5.2 Ethics problems

This section will discuss the ethical problems in the project and the consequences they could entail.

- The main ethical issues for this project are the impact of the application on users and the bias and discrimination that this application could cause. The application can impact a user’s vision of themselves and their representation. This in consequence could impact how they act, experience the world. This issue can be considered an open and closed door problem as it can recomfort a user in their beliefs or on the contrary shatter their beliefs. Depending on the results they get from the test, users, especially ones of younger age, can take the results too seriously that could have severe effects on identity and self-beliefs.
- The second issue is the bias and discrimination that could occur when running the application. The bias is mainly pre-existing and social as the users taking the test will be users using prolific creating a biased dataset due to this social bias. Additionally there will be some pre-existing individual bias from the client as our one client has absolute input into the design of the system impacting the design and format of the website.

In conclusion the group recognised two important ethical problems, the impact on users and the bias our application could have. The next section will cover the solutions found to counteract these concerns

5.3 The solutions

In this final section the reader will see the measures taken to prevent any ethical problems:

- The solution to prevent impacting users is having a form at the beginning of the website. This form will tell them what they are about to do and the potential risks it may entail. Additionally a button to show the results so users that want to see their results can see it by pressing the button and those who do not can just skip the graph. By doing these changes it minimizes the impact the application can have on a user.
- To deal with the second problem our group came with the following solution: to ask the TA, Tu coach and the client supervisor for their opinion, by doing so the client still does have the final word on the application however the impact on the project is no longer absolute.
- The solution to the bias and discrimination problem is a bit more complicated, since we have no control over who takes the tests in our app and we can only make sure that the computations are correct regardless of who takes the test. It’s the responsibility of the Prolific-based selection process to find a set of participants diverse enough to mitigate the negative impacts of such social bias.

\

6 Implementation

7 Product discussion and future recommendations

8 Conclusion