# RECMIX

## A Value Based Recommender for Music

**By:**

Kenzo Boudier, Lucas Fatas, Nathaniël De Leeuw,
Daniel Puente Barajas, Diego Viero

**4 June 2022**

TA - Bianca Cosma
Coach - Gosia Migut
Client - Sandy Manolios
Client Supervisor - Cynthia Liem

# [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 if people of similar values like similar music. The objective of our project is for it to be made public so we can run the experiment for a PhD student .

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. Even then, most of the technical terms are thoroughly explained throughout the report.

Readers that are curious about the organization and design of the project should read Chapter 4. For the readers that are interested in the implementation process, Chapter 5 and 7 are sections that cover these topics .

We would like to show our gratitude to our T.A. Bianca Cosma and our coach Gosia Migut for her constructive advice and reliability throughout the project. Additionally we would like to thank our technical writing teacher Sjaak Baars for his aid and teaching us ways to enhance our writing. Furthermore, we want to thank our client Sandy Monolios and her supervisor Cynthia Liem 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 Viero

# [Summary]

Everyone has values and personalities that differ from person to person. In the past years the world has found many ways to exploit the personality of individuals. Such an example is recommender systems: they recommend an object to a user based on users with similar personality and other factors. The aim of our experiment is to investigate whether values [4] in a person could influence and improve these systems. We have been assigned this task by PhD. candidate Sandy Manolios, who is investigating the relationship between these values and recommender systems.

This report aims to document the steps taken during the past 9 weeks to implement a web application that matches users based on their values and see whether the users like the music of their matched partners. These steps are in chronological order to show the thought process as well. The first step is where the problem and the goals of the website are mentioned. Then the following two steps will cover the structure of the application and have a concrete list of what needs to be implemented in the website. Afterwards, the next step will discuss the ethical problems that the project entails. The final step will give insight on elements that can be improved upon. This leads to the conclusion where the reader can see that we managed to implement a working application. However, there is still much room for improvement.

# Table of Contents

# 1 Introduction to our project

People have varying personality traits and values that define who they are as a person. Could these different personality traits and values give us an insight into the music taste of people? This is a question that PhD Candidate Sandy Manolios is trying to gain insight into for her Thesis. Sandy wants to conduct a study to examine this relationship. Studying the correlation between these attributes and a person's taste could prove to be very useful for future technologies such as recommendation systems.

The objective of this report is to outline the steps we have taken to develop a software to facilitate Sandy Manolios's study. A website needs to be created to gather the personality traits, values and top songs of participants. This data will be used to recommend to participants songs from other participants with similar personalities and values. They will then be asked to rate the songs and answer a few questions about them. All this data needs to be stored and be accessed through an interface so that Sandy Manolios can retrieve the data. This interface also needs to allow her to adjust the parameters of the study. 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 contain the analysis of the problem. The report will cover the requirements for the software in chapter 3. This will be followed by two chapters regarding the design (Chapter 4) of the website and an outline of the implementation (Chapter 5) respectively. Chapter 6 will be the discussion and the ethical implications of the software. The report will end with a Chapter 8 consisting of the conclusion

# 2 Problem analysis

The objective of this chapter is to analyze the problem and challenges laid out for us, the people affected by it and the consequences our product will have in the world. In section 2.1 we will study the context of the problem, as well as an explanation of the psychological methods we will use to investigate human connections. In the following section 2.2 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 2.3. Finally, we will take a look at the inspiration we took from existing apps in section 2.4.

## 2.1 Context of the problem

This subsection will focus on the topics and concepts outside of Computer Science that we will have to deal with in order to perform a successful experiment. More specifically, we will focus on the psychology aspect of the app and the different human classification methods that we will use to conduct the research: HEXACO based personality classification (section 2.1.1), PVQ based value classification (section 2.1.2) and random choice of users (section 2.1.3).

People can be classified according to different psychological scales, based on their nature. We intend to explore those classifications and check if there is any correlation with the music taste of similarly classified people. Since these two concepts can be abstract and easily confused, we will provide an overview of the psychological methods we will use:

### 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. These traits can be divided into the six HEXACO [1] categories of personality, according to modern psychology. We will use this division to quantify each participant's personality, as shown on figure 1:

- Honesty-Humility: people who do not break rules and lead a humble life.
- Emotionality: related to fear of physical dangers, need for emotional support and empathy.
- Extraversion: people who lead high extraversion scores are usually socially competent, and enjoy social gatherings and interactions, being the leader, and have high 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.


Figure 1: HEXACO 6 personality traits
(https://en.wikipedia.org/wiki/HEXACO_model_of_personality_structure#/media/File:HEXACO_1.png)

These 6 dimensions include all the important traits of human nature, and are used as the defining factors for it. 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 an immense number of human traits and factors that define people, 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 [5]. 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 and grouped according to the compatibility of such values as shown on Figure 2:

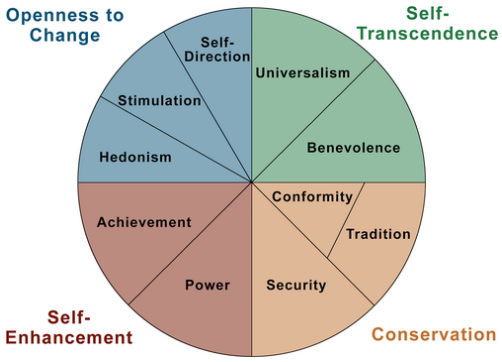



Figure 2: Schwartz's Theory of basic values
(https://onlinelibrary.wiley.com/doi/full/10.1002/per.2294)

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 that will match users randomly to others, and 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. This will be the control aspect of the experiment: we want to assess if the other kinds of recommendations will receive better feedback on average than the random one.

We can acquire the personality and values of each participant with two simple questionnaires. The participants' music taste will be retrieved using the Spotify API [3] and retrieving their top songs. The participants of the second batch (which will start when the researcher believes that enough data has been collected in the first batch) will get these songs recommended to them, after which
they will provide feedback on the quality of these recommendations. This feedback will be organized into mandatory and optional. Mandatory feedback will consist of a list of questions and a rating score from 1 to 5 assessing the quality of the recommendation. Optional feedback consists of participants providing open feedback about each playlist as a whole and rate from 1 to 5 each song in each playlist.

## 2.2 Problem statement

This section aims to divide the problem provided to us into more specific and achievable statements to solve. This will provide an overview of the issues that will be tackled in the development of our project: Gathering user data in a sensible way in subsection 2.3.1, correctly matching the participants according to their personalities and values in section 2.3.2. Section 2.3.3 will focus on tackling the feedback part of this experiment, and finally we will study the stakeholder satisfaction in section 2.3.4.

### 2.2.1 Gather data

Our group will have to conceive a website from scratch. There is no similar web application available for us to build upon. Our website will calculate the personality and value scores of a given participant based on the questionnaires. This data is then later used for matching participants and suggest songs of other users. The challenge is that our client, Sandy Manolios, must be able to easily retrieve the data and manipulate it.
The questions from which the questionnaire is composed, as well as the feedback questions and the instructive text to conduct the experiment 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.
We have to gather the participants' data, format them in a sensible way, and store this information for the research process. Thus, we have to let the participants know about this through a consent page, and conform to the GDPR guidelines [3].

### 2.2.2 Matching participants

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

### 2.2.3 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 researcher needs the data from these answers for her thesis. This feedback will have a mandatory component, which will be the main focus for the research; and an optional component, which will serve as a way to enrich the experiment and let the users provide deeper insight into the recommendations.

### 2.2.4 Satisfying stakeholders

The stakeholders will be more deeply discussed in section 3.1. During this project our main stakeholder will be our client, Sandy Manolios, and her supervisor, Cynthia Liem, since a product that satisfies her needs is what we're working for. We will receive feedback throughout the weeks with tips and suggestions on how to improve the app.

As we are collecting data we do need to conform ourselves to the TU Delft, Dutch and European data storage 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.

## 2.3 Project goals

After all the problem statements outlined in the previous section 2.3, where we have analyzed our responsibilities and the problems to solve, we have distilled our objectives into 4 distinct goals that will help us solve the objectives that we are set to fulfill in this project: create a user friendly app in section 2.4.2, use the spotify API in section 2.4.2, easily retrieve the data in section 2.4.3 and monitor the experiment in section 2.4.4.

### 2.3.1 Create a user friendly application

Our task is to create an app that will ask participants to log into their Spotify account (where the music will be retrieved), fill personality and value questionnaires, calculate the scores for each feature, and finally store this data along with the Spotify information of their top 5 songs. After enough data has been gathered, the researcher will

set the application to the second batch*. Posterior to answering the questionnaires, new participants will be recommended songs based on the existing data, and they will provide feedback on these recommendations. This feedback and all the data will be available to download through the researcher dashboard, so that Sandy can perform all the analysis and either prove or disprove her thesis.

### 2.3.2 Use the spotify API

Through the use of Spotify's API [3] we can retrieve participants' top songs and allow them to listen to 30 seconds snippets of the songs they are being recommended. This allows for better and more informative results from which the client will be able to receive better information. We decided to use snippets instead of the whole song since participants will have to listen to a total of 15 songs, and the 30-second snippets is a fair compromise. The Spotify API also allows us to fetch the participants' listening history, which becomes essential in the recommendation process.

### 2.3.3 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 process such that the client can extract the data from our app and directly work on the collected information, without having to deal with stored data manually herself. After discussing with our client different alternatives, we have decided that this retrieval will be done through CSV files. This process will facilitate the client's approach to the data and the way it is handled and examined.

### 2.3.4 Monitor the experiment

One requirement from the researcher perspective was to allow her to monitor the experiment at any time. We tackled this issue through the use of a dashboard which allows for an overview of the whole process, including control over the research, showing the number of participants that completed the experiment and the distance metric currently used for the matching. The dashboard can be accessed exclusively from the researcher.

## 2.4 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 consists of a personality and values questionnaire and a feedback section based on recommended songs. This chapter wants to show how we were inspired from already existing services in order to get a better idea about the implementation of our project.

For the questionnaire, there exists a profusion of personality test websites available on the internet. There are also plenty of websites where we can create questionnaires, Google Forms is an example. Although, open source alternatives, like Budibase or Cryptpad, are also an alternative. The products mentioned helped us gain some inspiration regarding User Interface and how to structure the questions in order to make them as user-friendly as possible. For the feedback 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.

# 3 Requirements

This chapter will discuss the division of each task into a list of requirements. This is handled with the help of the MoSCoW method, which will help us in the development of our project. Section 3.1 mentions the stakeholders that we had to take into account when eliciting the requirements. Section 3.2 discusses how we found those requirements and which methods we used. Finally, section 3.3 shows the final list of requirements that was conceived through the aforementioned process.

## 3.1 Stakeholders

This chapter examines 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. Stakeholders are taken into account at the time of the requirements elicitation, so that the final application satisfies 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 are directly affected by the final product.
- Indirect stakeholders pay attention to the finished product rather than the process of completing it. They are not necessarily affected by its development.

From an in-depth analysis, we defined the following direct stakeholders: our client (and 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.

As indirect stakeholders we can recognize TU Delft, since one of its Ph.D. students' thesis will be influenced by our product and it potentially could prove useful in the study of music recommender systems. 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 web application as passive stakeholders, since they will be influenced by the final product. Finally the Dutch and EU government, given that they will influence the way we handle the product. If at some point personality and/or values are implemented as part of recommender systems, music artists and people within the music world should also be considered as passive stakeholders.

## 3.2 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). The following section (2.1.2) mentions our brainstorming session that followed the interview. The last section (2.1.3) explains the MoSCoW methodused to prioritize the requirements of the web application.

### 3.2.1 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.

### 3.2.2 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.

### 3.2.3 Brainstorming session with the software group

This section discusses examples on how the users will use our application. We used these examples and stories to aid in the elicitation of the requirements. 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).

### 3.2.4 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.

## 3.3 Requirement list

Below we have listed out the requirements according to the MoSCoW method of the project. We decided to use the MoSCoW method[6] as it is great for high level requirements and easily allows for requirements to be changed or added. 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).

### 3.3.1 Functional requirements

In the following chapter, we list the final requirement of our project. All the following requirements are achieved, unless said otherwise.

#### 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 connect their Spotify account so that their top songs can be retrieved after login.
- The participant must see an introductory text for each questionnaire to have a clear overview of the process.
- The participant must be able to answer and modify the questions from 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 of batch 2 must be matched with other participants of batch 1 based on the responses to the questionnaires and be shown the top songs of the matched users, called recommendations.

- The participant must see his playlist recommendations all together and one by one in different pages.
- The participant of batch 2 must see an explanatory text for the recommendations.
- The participant must be able to rate the list of recommended songs and have the option to rate the individual songs on a scale of 1 to 5.
- The participant must be able to have the option to give open feedback about the recommended music for each recommendation.
- The participant must provide feedback according to a list of questions for each recommendation.
- The researcher must be able to download participant data and the similarity matrix into a CSV file through the dashboard.
- The researcher must be able to recognize the parameters of the experiment (batch number, similarity matrix and date and time) through the name of the CSV file.
- The researcher must be able to see the information inside the CSV file in a clear and structured way based on what was agreed upon.
- The researcher must be able to switch between batch 1 and 2 through the dashboard
- The researcher must have a dashboard to monitor the experiment and adjust parameters for the study (Examples: similarity metric, current batch).
- The participant must be able to visualize and export his personality and value scores after he is done with the experiment.
- 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

#### Should have

- 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
- The researcher should be able to change the final redirect link easily through a file

#### Could have

- The participant could be able to like or add recommendations to his account (not achieved)
- The project will handle more batches in the future. (not achieved)
- The researcher shouldn't be able to change parameters like questions during the process of one batch

#### 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

### 3.3.2 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.

# 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 4.1 we'll discuss the architecture used to define the structure of the

project. Afterwards we'll talk about everything that concerns the hosting in 4.2. Finally, we'll discuss the different testing methodologies that were used in 4.3.1.

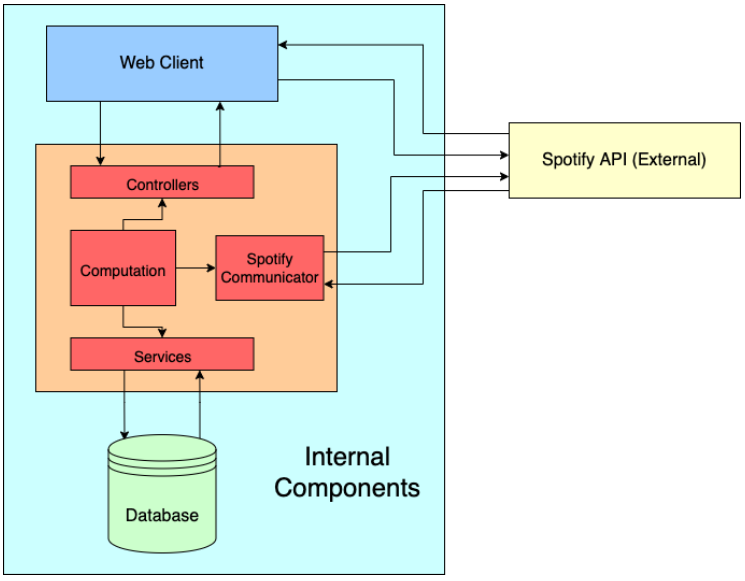
## 4.1 Architecture

This section will discuss the architecture of our application and how our project will be structured. Subsection 4.1.1 discusses the back-end server, while the web client and database are discussed in 4.1.2 and 4.1.3 respectively. Finally, the Data Structure is looked at in subsection 4.1.4

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 back-end 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.

### 4.1.4 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.

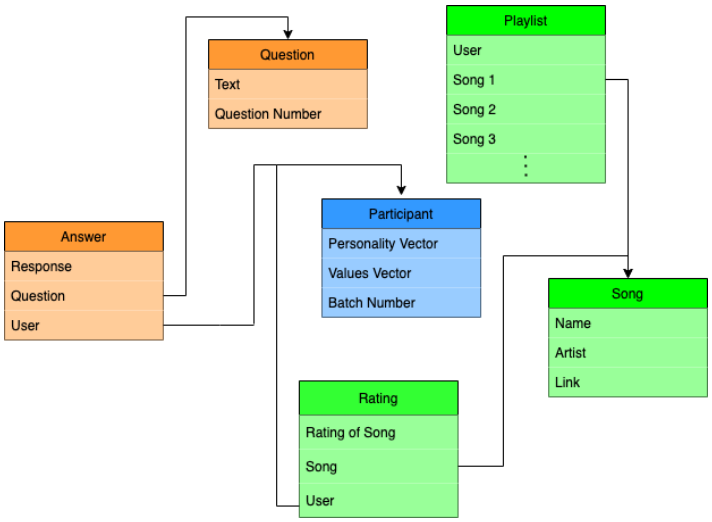



Figure 2: High level overview of data structures

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. **Song Rating:** The participant's rating of a song that was recommended to him. Each rating for each song will be stored individually.
7. **Playlist Rating:** The participant's rating of a recommendation. That playlist also contains open feedback and the score for the questions asked by the client.
8. **Match:** Stores the participant's id along with the id of the three users he has been matched with.

## 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

We will now discuss testing and how we assessed that our app worked properly and did what we asked for. We show the different ways we tested our app in subsection 4.3.1. 4.3.2 and 4.3.3 refer to the specifics of front-end testing and back-end testing, respectively.

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 Implementation

The implementation section provides a deep overview of how the application was developed. 5.1 discusses everything regarding the Scrum methodology and how it benefited the teamwork in the group. 5.2 mentions the tools utilized for intra and extra-group communication. Afterwards, 5.3 covers how we assessed weekly meetings with both the client and teaching assistant and improvements made based on the feedback. Finally 5.4 examines how the application was implemented with specific attention to how the final user perceives each part.

## 5.1 Project management

We decided that using the Scrum with one-week sprints would be the most beneficial methodology to use for our project.

### 5.1.1 Why use scrum

While considering the different methodologies, we first decided that we would use an Agile methodology. The key reasons for this were adaptability and that working software is frequently delivered. In early meetings with the client, there seemed to be certain aspects of the project that were still uncertain to them, so by using an Agile methodology we had more flexibility to adapt to changes [9]. Since we have had weekly meetings planned with our client, the incremental and interactive nature of agile methodologies has allowed us to frequently share and get feedback on working software.

Before settling for Scrum, another agile methodology that we considered is Kanban because of its visual and interactive methods which would have enabled the team to have a cohesive workflow at all times. But we decided that Scrum would be ideal for this project as we are all familiar with it , and it is better in small highly collaborative teams [10]

### 5.1.2 Scrum roles

For the Scrum roles, we decided that Lucas Fatas would be the project owner. He focused mainly on managing the scrum backlog and ensuring that our software is satisfying the requirements. Diego Viero was the Scrum master and organized our sprint meetings, managed blockers and ensured we worked transparently.

### 5.1.3 Weekly sprints

We decided to work with weekly sprints since the timeframe for our project is short. Our meetings with the client are every Thursday afternoon, so we decided to have our sprint reviews Thursday mornings before the meetings and our sprint plannings after the meetings. This allowed us to get feedback quickly on the work we did during the sprint and allowed us to adapt to any changes requested by the client before our sprint planning sessions.

## 5.2 Tools

Throughout the project we make use of different tools and technologies that helped us with:communication,

development, version control and more. Here is an overview of these tools and technologies:

### 5.2.1 Gitlab

We used a repository in Gitlab to share our code in a collaborative way and keep a clear overview of the version control of the code. Furthermore, we also used it for its usefulness in code quality checks and issue coordination. We assigned an issue for each requirement, feature and bug to be implemented or fixed in order to have an overview of the sprints and what each of us worked on.

Our issue boards contained a list of all these issues and also held information about the current state of each issue in the current sprint (if the issue is to be done or is being implemented, tested, or reviewed), as well as some helpful information on how to solve said issue, which could help the developer in charge take care of it. This gave us a clear overview of our progress across sprints and a straightforward way of dividing work and checking who was doing what.

When starting a sprint, we took a look at the issue board and chose which requirements to assign to the upcoming sprint. For every requirement, we created sub-issues (even if there is only one) that needed to be implemented in order to solve the requirement. These traversex the Development boards (backlog → doing → done → completed). At the end of each sprint, we closed the requirement issues after making sure that all its sub-issues were completed. Every issue was worked on in a separate branch which was then split into back-end and front-end. We then created merge requests for the issue once it was completed. We used a protected development branch where all other branches were merged into. Merging into dev was done by merge requests, each of which were reviewed and approved by at least two other developers who have not worked on the issue.

### 5.2.2 Communication

- Mattermost. We used Mattermost for communication with our TA and to keep up-to-date on the latest announcements concerning the project.
- Microsoft Teams. We used Microsoft Teams for communication with our client and to schedule weekly meetings with her, which we used to ask for feedback and solve some questions about the project that arose during the development. We used the calendar embedded in the application to organize these meetings.
- Webmail: This platform was used to maintain communication with our coach and organize the meetings with her.
- Discord. We used Discord to send each other links and useful information to each other and hosted meetings when it is not possible to meet in person.
- Google Drive. We used a shared Google Drive folder as the means to handle files about the project in an organized way. More specifically, we tackled each assignment with a shared Google Docs so that we can all work on it at the same time. We also used it to work on the documentation of our software, process and meetings.

## 5.3 Client and TA meetings

We had weekly meetings with both main stakeholders, the researcher and the university represented by our TA. In the meetings with the researcher we showed our progress and the final version of the web application. Based on her feedback we were making slight changes and improvements for the next meeting. Most of the feedback of our researcher was about the front-end and the way information was shown and retrieved.
The weekly meetings with our TA were more focused on the progress rather than the product. Thanks to these meetings, our efficiency increased week after week. We learned to use the tools methodologically. The GitLab and the repository are a good representation of our advancements, where we can clearly see a difference between the

first and last weeks.

## 5.4 Application overview

The section is separated into subsections that represent the different parts of the application. Each one elaborates on how the application was implemented and how it behaves from the final user's perspective.

### 5.4.1 Spotify data

Retrieving the participant's Spotify songs is a crucial part of the software. To be able to do this the participant first is shown a consent form and required to accept it before moving on. Once the participant has consented, they are redirected to a Spotify page. Here they will log in to their account and give permission for Spotify to give us an access code to allow us to collect their top songs. This access code gets sent to our back-end where we will retrieve the participants' top five most listened to songs in the past six months. After the participant has completed this section they will be redirected to the beginning of the questionnaire

### 5.4.2 Questionnaire

The questionnaire component consists of two separate sets of questions. One consists of 40 questions from the PVQ [4] test used to determine the user's values, the other one is composed of 60 questions from the HEXACO [2] test used to calculate the user's personality. Their order is randomized for each person participating in the experiment. After logging into their Spotify account the participant will be prompted with an introduction of the first random questionnaire. After reading the introduction, the participant can proceed to the actual questions. These are separated into different pages and in order to advance, they must answer all questions in the current page. Once done with the first questionnaire, the participant is prompted with the introduction to the second questionnaire, after which he repeats the same procedure as before. Finally, when submitting the second questionnaire, the participant is shown two graphs that display the results of both questionnaires.
At any point in the questionnaire, the participant has the opportunity to change answers already given and go back to previous pages.

All answers are stored in session storage in order to prevent potential errors that could occur if the participant mistakenly refreshes the page. Everything is kept locally until the second questionnaire submission, where answers are parsed from session storage and sent to the back-end. Once received, a function computes the result of the experiment and sends them to the front-end, which displays them using the graphs mentioned above.

Once on the results page, if the experiment is currently in the first batch*, the participant will be shown a thanks page containing the prolific code used to conclude the experiment. Otherwise they are shown the introduction to the recommender section.

### 5.4.3 Recommender

The Recommender component is exclusively for batch 2 participants and later. On the first page the participant is shown in a random order the 3 following playlists, a random, a value and a personality playlist. The personality and the value playlist are the top songs of the most similar participant of the previous batch in those specific domains.

The similarity metric used for the matching is defined in the dashboard by the researcher.
The participant is asked to listen to all the given songs and rate the playlist based on his taste. We are using a 5-star rating to represent their taste. The same method is used for rating the songs, but this is optional.
Only after the participant has rated all the playlist, he/she can continue to the following pages, where he/she is shown the playlist individually and is asked to answer some questions about them. When answering the question, the participant can still listen to the songs and change their previously given ratings. Optionally, the participant can also give open feedback about the individual playlists.
The participant can finally submit their answers, but a check is made if they have rated all the playlists and all the questions.
Like for the questionnaire part, all answers are stored in session storage in order to prevent potential errors that could occur if the participant mistakenly refreshes the page.

### 5.4.4 Dashboard

The researcher's dashboard is accessible only through a predetermined link and cannot be reached by standard interactions with the application. After clicking on the link, the researcher is redirected to a login page.
Once logged into the dashboard, a JWT token [3] is generated and stored locally. To perform any action in the dashboard, a call is first sent to the back-end to check whether the token is valid. In case it isn't, the researcher is sent back to the login page, otherwise the action is performed as expected. If the website is closed, the researcher is being automatically logged out.
The dashboard has 3 main functionalities.
The first is retrieving data about the experiment in CSV files. Calls are sent to the back-end asking for specific data and the data is then translated to a CSV file and downloaded at the front-end.
The second functionality is an overview of the current batch. The researcher can see the current batch number and what type of batch it is, together with the number of participants of this batch and the metric used to recommend the playlists.
The last functionality is changing the batch. A button can be clicked when it is time to go to the next batch. After choosing a specific similarity metric, a call is made to the back-end and the new batch is created. This is then also directly updated in the front-end.

# 6 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

## 6.1 The five ethical categories

The following section will briefly explain the 5 ethical problems given by TU Delft Write[7] . Additionally, the relevance of each category in the scope of this project will be discussed.

### 6.1.1 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.

### 6.1.2 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.

### 6.1.3 Bias and discrimination

Bias and Discrimination covers all the problems with the bias a machine/application could 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.

### 6.1.4 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” [8] and therefore modifiable in case of unjust results
- No “decisions” are being made after the matching process.

### 6.1.5 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.

## 6.2 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

## 6.3 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.

# 7 Product discussion and future recommendations

Here, the final product and future improvements of the product will be discussed. It is favorable to be acquainted with the requirements list (section 3.2) before reading this chapter. Section 7.1 will explain a number of noteworthy requirements/changes and how we managed to implement them. Then section 7.2 will show 3 features can be implemented to extend the application.

## 7.1 Noteworthy requirements/changes

Subsection 7.1 will discuss noteworthy requirements/changes. Noteworthy requirements/changes are requirements that stand out from the rest due to an aspect that renders them unique.

### 7.1.1 Showing participants a graph of their values and personality

A notable feature that is worth mentioning is the page showing the participant personality and values. To display this page and its features, both front, back-end and database are required. This feature is a great example of how each component interacts with one another. When participants submit their answers to determine values and personality, the front-end makes a call to the back-end with a message. The message contains all the answers and a unique id attributed to this specific user. Once the server (backend) receives the call, it computes the personality and values using fixed formulas given by the PVQ and HEXACO tests. Afterwards, the backend sends the questionnaire and the calculated values and personalities to the database. When the participant arrives on the final page, the frontend sends a request to the backend who sends a request to the database to fetch the values and the personality of the users. Then the backend sends back two lists containing the results to the front end. The front-end, after receiving them, displays them using a javascript library used to display radar graphs (lsee figure).

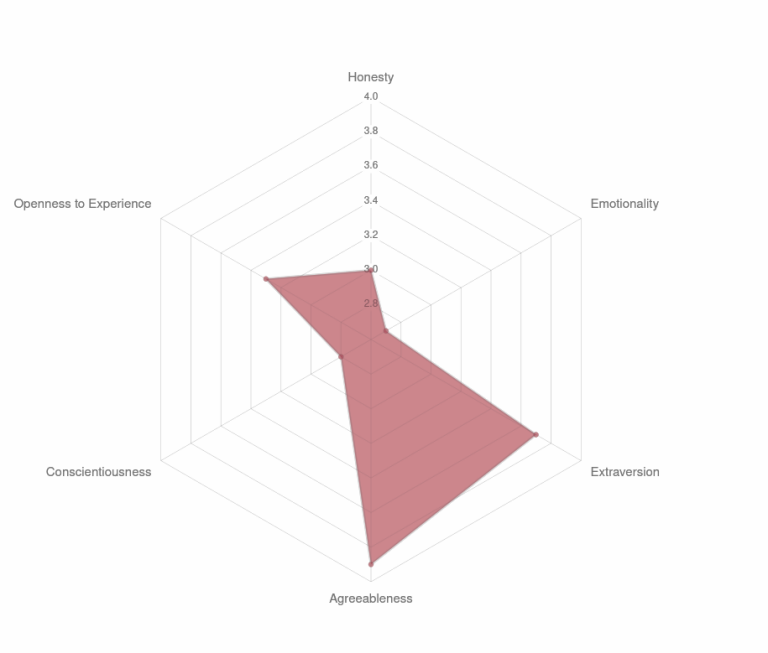



Figure: Example of graph displaying participant's results.
Graph created through: https://r-charts.com/ranking/radar-chart/

### 7.1.2 Being able to listen to 30 second music snippets

A noteworthy feature are the 30 second music snippets. These snippets come from spotify and allow participants to listen to 15 music snippets when reviewing the playlists. The reason this is a noteworthy requirement is due to the complexity of the task. Participants have to login to spotify then the application automatically gathers the top songs the participants listen to. This task is quite complex as it requires us to deal with a third party authentication.

### 7.1.3 The changes to the database

The database underwent a few different versions before the final version was agreed upon. These modifications were implemented in order to improve the usability and flexibility of our app, and to make use of a unified naming convention and professionalize this process.

The first change that was implemented was the possibility to abstract the number of questions asked about the recommendations. We are at the moment using 6 questions, and the old database had the schema QuestionFeedback with userId and 6 fields, one per question, to store this information. However, we want to allow Sandy to change the list of questions at some point, so we abstracted it and the new QuestionFeedback schema has userId, questionNumber, and answer. This allowed for a more complete version of the app and the abstraction of the number of questions. However, this made it a challenge to return the answers to these questions in a single entry in the CSV files, since information about one single final entry of the feedback was scattered through multiple columns. This was thankfully solved with the GROUP_CONCAT function of SQL, where we could retrieve all the entries in one single row.

The second change added was the abstraction of the song ratings. At first, we stored all entries of the ratings, which consisted of userId, matchedUserId, spotifyUrl and rating. However, most of these entries would be 0 since it is not mandatory to rate them to fulfill the questionnaire. Because of that, we implemented a playlistNumber field that tells us which song out of the 5 the rating was about. This was an important change since as many of the entries would be 0, our database would have a lot of unimportant data in this schema. This way only the relevant data is stored and we know that the rest of the songs are not rated and are thus 0.

Finally, we changed the naming conventions of the tables and variables to a unified camelcase convention in order to avoid possible mistakes: tables use capital letters in all the first letter of words, and variables in all but the first word. There are no spaces between words. This reduced the confusion between back-end developers when creating schemas and queries.

## 7.2 Possible ways to expand the product

This final part will discuss features that were either not thought of during the creation of the requirements or requirements that were on the requirement list but unable to be implemented . The objective of this section is 3 additional features that could be implemented and their impact on the application.

### 7.2.1 Adding a “forgot password” feature on the dashboard

One important feature that should be implemented should be an easy way to change passwords to login on the dashboard. Right now, it is impossible to change the password on the web application. This means that if Mrs. Manolios forgets the login credentials, she has to look in the database for the credentials. The only way to change the password is to go in the code and do it manually. In consequence this supposedly easy task is hard. Additionally, by touching the code, the client may inadvertently modify or delete some pieces of code leading to a failure of

some features or the entire application. This is why we think it is a necessity to implement this feature if the application is extended.

### 7.2.2 Adding multiple batches

Adding multiple batches was a feature in the requirement list however due to lack of time, it was not implemented. This feature consists of allowing the client to add extra batches or restart the entire experiment. The client would be able to create the batches or restart the experiment on the dashboard via some additional buttons. The project as it is right now only has two batches and no possibility to restart. This is very constrictive for the client as it does not allow scalability or repeatability of the experiment. The implementation of this additional feature would allow the client to overcome all the technicalities mentioned previously.

### 7.2.3 Adding a button to share the results from the two tests

The addition of a button to share the graph is a feature that does not exist on the requirement list however it would be beneficial to implement this feature. As of now the application only projects the results of the tests and there is no way to share the results. The implementation of this feature would allow participants to save and/or share their results from the value and personality test. This feature is not for the client but more for the participants as a way to thank them for participating in the experiment. Moreover, participants sharing their results promote the experience which could lead to more people taking part in the experience greatly helping Sandy Monolios for her experiment.

# 8 Conclusion

The purpose of this report was to outline the steps taken to develop a software to facilitate Sandy Manolios's study. The web application we implemented allows participants to fill out a questionnaire and be matched to similar users based on value and personality and rate the music of the matched participants. The application includes a dashboard for the researcher to interact with the study and retrieve the data.

To satisfy the aim of this report we demonstrate how we approached the project and the main problems we would encounter. We then formulated a requirement list to show the features that should be implemented in our product. How we then designed, structured and hosted the project was thoroughly explained, followed by the actual implementation of the website and the user experience. The report also tackles the ethics of the project and how they were dealt with. Lastly, the report gives insight on important features and possible ways to expand the product.

The end product meets the requirements of the software however we believe that more work will be required to complete the study and that there are possible improvements to the software. Some of the potential new features range from small changes like changing the researcher's credentials if the password is forgotten or compromised to larger changes like allowing for multiple batches. These features are not crucial to the study but would be beneficial. What we believe will be necessary is to have a developer work with Sandy Manolios when the study is being conducted to ensure maintenance of the software and server and proper function of the application.

# Reference List


[1] M. C. Ashton and K. Lee. 2009. The HEXACO–60: A Short Measure of the Major Dimensions of Personality, Journal of Personality Assessment.

[2] Hexaco.org. 2022. The HEXACO Personality Inventory. [online]

[3] General Data Protection Regulation (GDPR) 2022. [online]

[4] S. H. Schwartz, G. Melech, A. Lehrnami, S. Burgess, M. Harris, V. Owens. 2001.Extending the cross-cultural validity of the theory of basic human values with a different method of measurement, Journal of Cross-Cultural Psychology.

[5] S. H. Schwartz. 2005. Basic Human Values: An Overview, The Hebrew University of Jerusalem.

[6] S. Hatton. 2008. Choosing the Right Prioritisation Method, 19th Australian Conference on Software Engineering. 2008

[7] TU Delft Writing Center, Faculty of Technology, Policy and Management

[8] SciForce. 2022. Introduction to the White-Box AI: the Concept of Interpretability. [online]

[9] A. Cockburn. 2014. People and methodologies in software development, Faculty of Mathematics and Natural Sciences.

[10] H. Lei, F. Ganjeizadeh, P.K Jayachandran and Ozcan. 2017. A statistical analysis of the effects of Scrum and Kanban on software development projects, Robotics and Computer-Integrated Manufacturing.