

CHADMusic

Free music and karaoke app

Dragos-Daniel Bonaparte, 315261

Dan-Sebastian Ceapa, 315162

Chiril Luncasu, 315171

Matas Armonaitis, 315263



Supervisor:

Steffen Andersen (SVA), Mona Andersen (MWA),

Henrik Kronborg Pedersen (HEKP)



VIA University
College



[Number of characters]

Software Technology Engineering 2nd Semester

2.06.2022

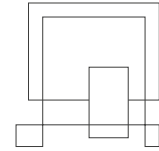
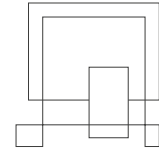


Table of content

Abstract.....	iii
1 Introduction.....	1
2 Analysis	3
2.1 Requirements	3
Functional Requirements	3
Non-Functional Requirements.....	4
2.2 Given the product backlog the use case were formulated:	5
2.2.1 Log In use case.....	7
2.2.2 Sign up use case.....	8
2.2.3 Play music use case	9
2.2.4 See lyrics use case	10
2.2.5 Search song use case.....	11
3 Design	12
4 Implementation	13
5 Test.....	24
5.1 Test Specifications	Error! Bookmark not defined.
6 Results and Discussion	25
7 Conclusions	28
8 Project future	29
9 Sources of information.....	30
10 Appendices	1



Abstract

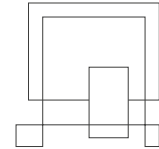
The aim of the project is to create probably the best app in terms of the existing competition when it comes to a music app, featuring a lot of features and functionality and at the same time remaining free of charge and ad-free for every user.

The team aimed for a product that will be able to deliver the existing trend features, and at the same time creating a easy to update and maintain code.

The project was constructed using Unified Process and SCRUM, applying the agile beliefs every time the team coded. As for the classes in the project that implement the back end of the program, the team followed SOLID to make everything efficient and simple.

As for the main design used by the team is MVVM which totally isolates the back end from the front-end. In terms of other used designs, the team also implemented the observer pattern, the factory method and are planning in a future release to implement the proxy pattern as well.

After 5 sprints, each of 3 days, the team managed to develop a product that not that it was satisfactory for the time, but it surpassed some of the team's expectations when talking about the features.



1 Introduction

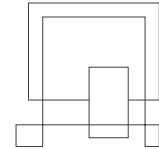
Today a lot of music apps strive to achieve a better music quality (Anon., 2022b), but the customer is annoyed by the fact that the music apps do not focus on implementing functionalities. Prior to this the customer did some research and based on the feedback of various music app users, concluded that a lot of those apps do not meet expectations when it comes to functionalities like displaying the lyrics of a song, sound design, shuffle playlist, putting the app to sleep when not used, download music/listen offline or if the app includes those features, they come at a monthly subscription fee. The current alternatives when it comes to music apps are Spotify and Tidal, both having great music quality but at some cost of course. Elaborating more on specific aspects of each of the two alternatives the customer deducted that even though Spotify is free for you to listen to music but the number of ads in the app are completely ruining the experience, on top of that you are not able to download music offline. Speaking of worse, Tidal offers great functionalities and features but this app does not even allow you to play music with ads. Instead, it requests payment from the first installation, which is not very appealing at the beginning since you do not even know how the experience will be.

Those things alone will make the average teenager to start using the offline music app that their phone comes with from the factory. The market of those music apps are teenagers, thus making it difficult for them to embrace one app or another. Usually as a teenager you are not so happy about needing to make a monthly subscription just to listen to music.

A possible music app that can be used by teenagers is YouTube music, which is easy to use but has a downside, lots of ads and very little songs with lyrics.

In summary the current problem is lack of free access to music and features in those music apps.

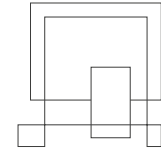
It is reasonable to infer the need for a better and free alternative that will satisfy the needs of present youth.



This app will:

- Create a more personal experience having separate users with separate playlists.
- Offer an ad free music experience
- Allow the user to sing along with the song since there is lyrics for every well-known song.
- Access to an unlimited supply of songs

Delimitation-wise, the app will not be able to add songs to playlists in this version, does not provide a shuffle option yet and the app needs optimizing when it comes to sending songs, since it takes a relatively long time to send.

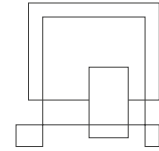


2 Analysis

Understanding the problem thoroughly, the product owner was able to infer the product backlog the customer would desire.

2.1 Requirements Functional Requirements

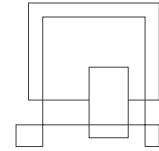
ID	Priority	Hours estimate	Task
1	Critical	72 hours	As a user I want to be able to play and listen to music in a console environment.
20	Critical	18 hours	As a user I want to be able to play and listen to music in a program window.
2	Critical	36 hours	As a user I want to be able to see the lyrics of the song that I am listening to.
3	Critical	36 hours	As a user I want to be able to pick a song from a list.
4	Critical	18 hours	As a user I want to be able to pause or resume the song.
21	Critical	32 hours	As a user I want to have multiple computers playing from the same list of songs.
5	Critical		As a user I want to be able to change the volume of the song. (Canceled by the product owner)
22	High	6 hours	As a user I want to be able to run the program by having a shortcut on my desktop.
6	High	10 hours	As a user I want to go to the next or previous song.
7	High	10 hours	As a user I want that after a song finishes, another one will start playing automatically.
8	High	36 hours	As a user I want to be able to have a liked songs playlist, for a more personal experience.
9	High	10 hours	As a user I want to be able to search for a song.



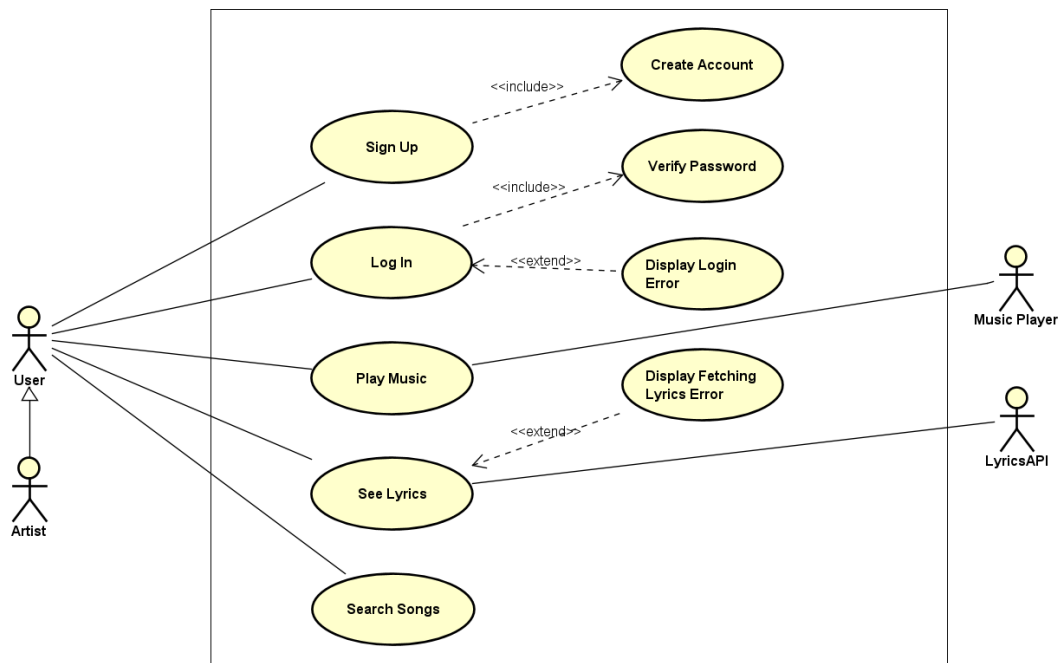
10	High		As a user I want to have the songs categorized by genres.
11	Medium		As a user I want to be able to shuffle or repeat my playlist.
12	Medium		As a user I want to get song recommendation.
23	Low	2 hours	As a user I want to know how many hours I spent listening to music.
13	Low		As a user I want to be able to change font size, color and style of the lyrics.
14	Low		As a user I want additional information about the song like the author, length, and year.
15	Low	36 hours	As a user I want to have a separate account to not merge my liked songs with another user.
16	Low	2 hours	As a user I want to be able to toggle between showing lyrics and not showing lyrics.

Non-Functional Requirements

17	Non-Functional	As a user I want to see a playlist with the most listened songs.
18	Non-Functional	As a user I want to see my activity, for example how many hours I have listened today.
19	Non-Functional	As a user I want the app to go to sleep after some inactivity.

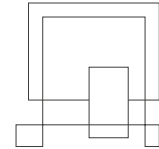


2.2 Given the product backlog the use case were formulated:



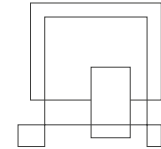
What follows is a table illustrating the relationship between the requirements and the use cases. The numbers in bold represent critical requirements.

Use cases	Covered requirements
Sign Up	15
Log In	15
Play music	1,20,3,4,6,7
See lyrics	2,16
Search songs	9
No use case in particular	21,22



The two requirements were met but not by any use case in particular, because:

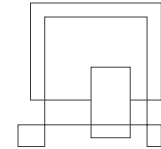
- Requirement number 21, states that we should be able to connect to a server so that maybe the customer would like to connect his computer and laptop at the same time.
- Requirement number 22, states that there should be runnable jar files on the desktop for easy execution of the program rather than running it in IntelliJ.



2.2.1 Log In use case

When a user launches the program, he has to login to the system to be able to use the whole program. To login user needs to fill in username and password text fields and press login button. Then the system checks if the user exists in the database and opens the main window of the program, if the user is not in the database, he will be asked to put in other credentials.

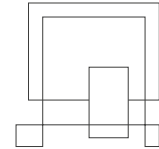
Use case section	Purpose
Use case name	Login
Scope	The Login system
Level	Logging into system
Primary actor	User
Stakeholders and interests	The user wants to login to the system to use the program
Pre-conditions	The user must have an existing account in the database
Success Guarantee	The user has an account that exists in the database
Main Success Scenario	<ol style="list-style-type: none">1. The user enters correct credentials in the needed text fields2. The system finds the user in the database3. The system opens the main window
Extensions	<ol style="list-style-type: none">1. If the user is not in the database, then the system won't open the main window
Special requirements	User must be in the database
Technology and Data variations list	The user can login to the system
Frequency of occurrence	Always
Misc.	



2.2.2 Sign up use case

When a user launches the program for the first time, he has to register to the system to be able to use the whole program. To registering user needs to fill in username, email and password text fields and press register button. Then the system checks if the user does not exists already in the database and opens the main window of the program, if the user is in the database, he will be asked to put in other credentials.

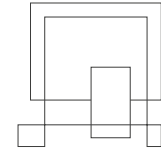
Use case section	Purpose
Use case name	Sign Up
Scope	The Register system
Level	Registering into the system
Primary actor	User
Stakeholders and interests	The user wants to register to the system to use the program
Pre-conditions	The user must have an email address
Success Guarantee	The user must not be in the database before
Main Success Scenario	<ol style="list-style-type: none">1. The user enters correct credentials in the needed text fields2. The system does not find the user in the database3. The system opens the main window
Extensions	<ol style="list-style-type: none">1. If the user is in the database, then the system won't open the main window
Special requirements	User must not be in the database
Technology and Data variations list	The user can register to the system
Frequency of occurrence	First time use
Misc.	



2.2.3 Play music use case

After the user logs in or registers to the system, the system will open the main window where the user needs to press the “All songs” button to show in the table all the songs in the database. From this point the user will need to select a song and press the play button.

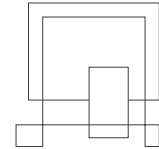
Use case section		Purpose
Use case name		Play music
Scope		The Player system
Level		Listening to music from the system
Primary actor		User
Stakeholders and interests		The user wants to listen to music
Pre-conditions		The user must select a song from the table first
Success Guarantee		The user must select a song and hit the play button
Main Success Scenario		<ol style="list-style-type: none">1. The user presses the “All songs” button2. The system retrieves the list of songs3. The user selects a song from the list4. The system sends the song to the client machine5. The user presses the play button
Extensions		
Special requirements		
Technology and Data variations list		The user can listen to music from the system
Frequency of occurrence		Very often
Misc.		



2.2.4 See lyrics use case

After the user logs in or registers to the system, the system will open the main window where the user needs to press the “All songs” button to show in the table all the songs in the database. From this point the user will need to select a song and press the play button, afterwards the user will need to press the show lyrics label.

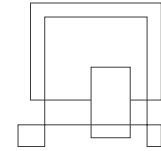
Use case section	Purpose
Use case name	See lyrics
Scope	The Lyrics API
Level	Seeing lyrics for a song
Primary actor	User
Stakeholders and interests	The user wants to see the lyrics from a song.
Pre-conditions	The user must select a song from the table first and play it.
Success Guarantee	The user must select a song and hit the play button, after which the user presses in the show lyrics label
Main Success Scenario	<ol style="list-style-type: none">1. The user presses the “All songs” button2. The system retrieves the list of songs3. The user selects a song from the list4. The system sends the song to the client machine5. The user presses the play button6. The user presses the show lyrics label7. The system opens the lyrics window once the lyrics are fetched
Extensions	<ol style="list-style-type: none">1. If the song does not have lyrics it will not show anything
Special requirements	
Technology and Data variations	The user can see lyrics of the songs the user listens to list
Frequency of occurrence	Very often
Misc.	



2.2.5 Search song use case

After the user logs in or registers to the system, the system will open the main window where the user needs to press the “All songs” button to show in the table all the songs in the database. From this point the user will need to select from the choice box what is he searching for, f.x. “title” or “artist” after which types the name of the song or artist in the search bar.

Use case section	Purpose
Use case name	Search song
Scope	The Player system
Level	Finding music in the system
Primary actor	User
Stakeholders and interests	The user wants to find music
Pre-conditions	The user must have pressed the “All songs” button
Success Guarantee	The user must have pressed the “All songs” button and search for a song
Main Success Scenario	<ol style="list-style-type: none">1. The user presses the “All songs” button2. The system retrieves the list of songs3. The user chooses what to search for4. The user types the name of the song or artist5. The system shows the song
Extensions	<ol style="list-style-type: none">1. The song may not be in the database
Special requirements	
Technology and Data variations list	The user can listen to music from the system
Frequency of occurrence	Not so often
Misc.	



3 Design

The purpose of the design section is to outline HOW the system is structured; i.e. to transform the artefacts of the analysis into a model that can be implemented. The design section is relevant for the programmer, whereas the analysis is relevant for the stakeholder.

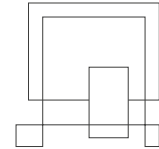
Elements that may be relevant in this section:

- Architecture: Find architecture patterns here (Leszek Maciaszek 2004, chap.9).
- Technologies: Describe technologies used, also alternative technologies. Argue for choice of technology according to the project aim.
- Design Patterns: Describe which design patterns (GoF (Gamma et al. 2002) etc.) you are using and why.
- Class Diagrams
- Interaction Diagrams
- UI design choices
- Data models, persistence, etc.

You must explain all diagrams in the report. These diagrams including descriptions are the blueprints for the implementation.

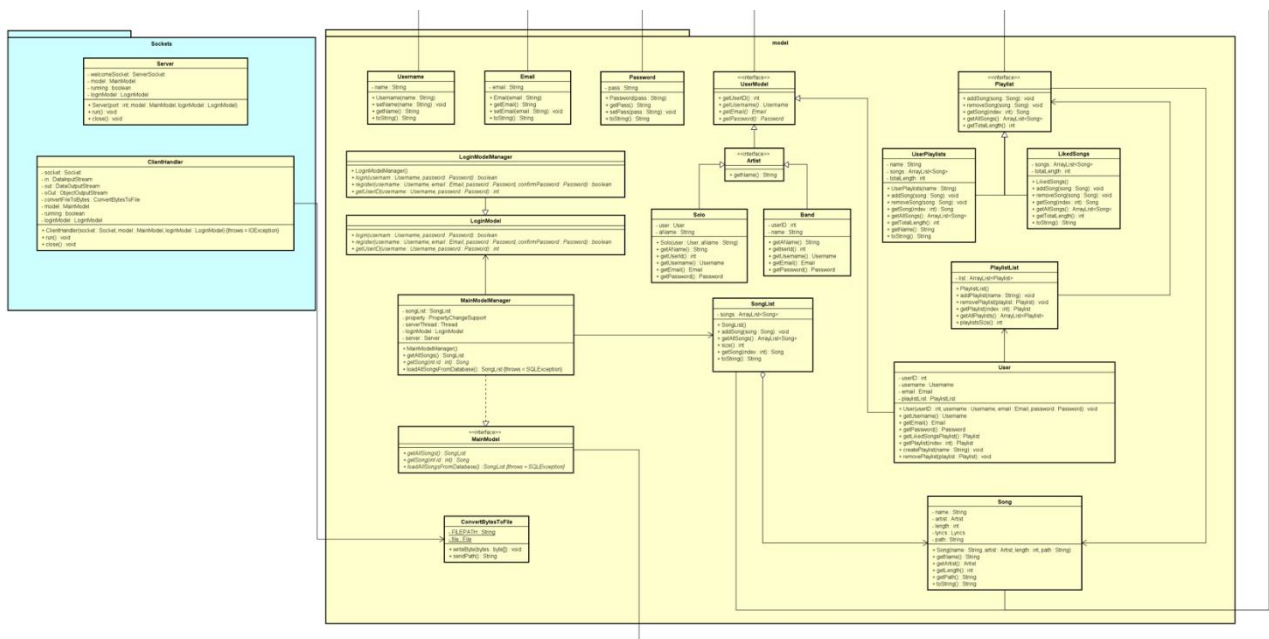
Hint: One way to figure out which objects/classes are needed in the design is to apply the General Responsibility Assignment Software Patterns/principles (GRASP) (Larman 2004, chap.17).

Hint: Consider how to design your system to make it testable.



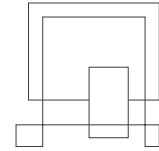
4 Implementation

Our implementation started very small and throughout the sprints it got larger and larger as the days went on. The class diagram for our system is combined, the server and the client being on the same file.



The top picture represents the server part of our program, it does not have any GUI at all since it is not needed and the product owner didn't request more than this. The central part of the Server-side is the MainModel where all the magic happens. The MainModel interacts with the sockets, the LoginModel and many other classes.

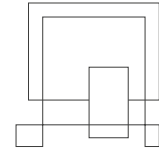
As for the server type built, the server has been built on Sockets, since there is a lot of control over them and this control was something necessary in the construction of this program.



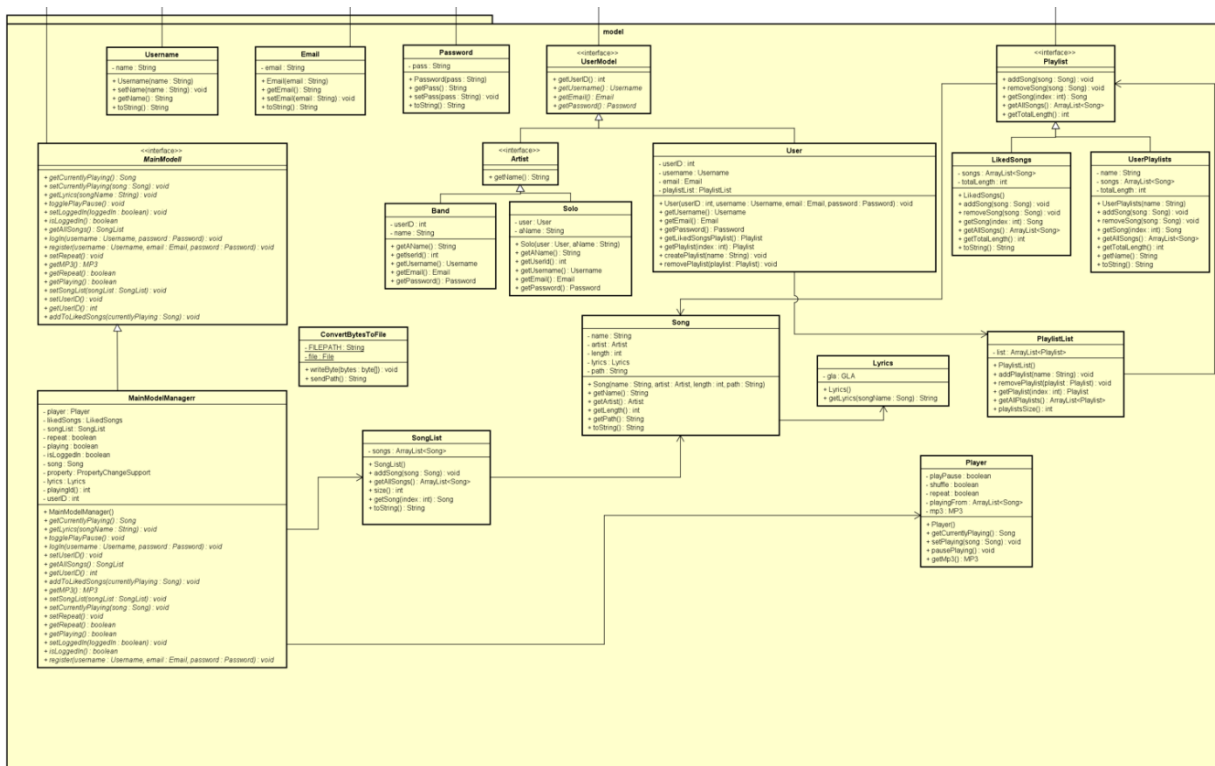
Some very interesting code can be found in the ConvertSongToFile class

```
1 usage
public byte[] convert(Song song) throws IOException {
    System.out.println("Running convert!!!!");
    file = new File(song.getPath());
    fileStream = new FileInputStream(file);
    length = file.length();
    if (length > Integer.MAX_VALUE) {
        System.out.println("TOO LARGE - ERROR");
    }
    byte[] bytes = new byte[(int)length];
    int offset = 0;
    int numRead = 0;
    while (offset < bytes.length && (numRead=fileStream.read(bytes, offset, len: bytes.length-offset)) >= 0) {
        offset += numRead;
    }
    if (offset < bytes.length) {
        throw new IOException("Could not completely read file "+ file.getName());
    }
    fileStream.close();
    return bytes;
}
```

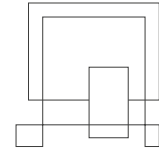
The solution of sending song files from the server to the client was to convert the song file into a bytes array and then send it to the client as a bytes array.



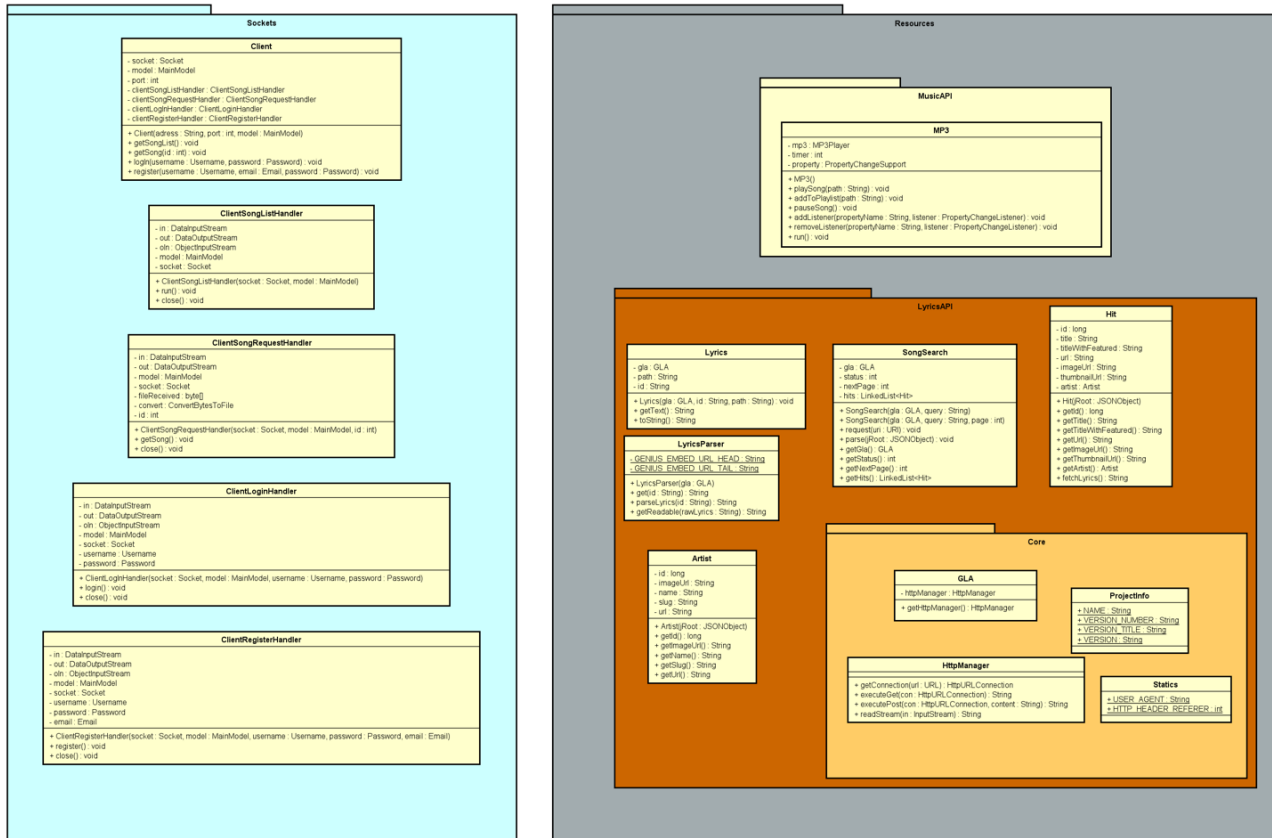
Up next there is the client side it was chosen the MVVM design pattern meaning that it should have 3 separate packages (model,viewmodel and view) but because the client needs to connect to a server there is also a sockets package and a resources package where the lyrics API relies:



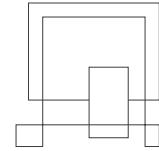
Like with the server side, on the client side everything resides on the MainModel. This is only the model package since it will be too big to put the whole diagram. The client is missing a lot of the methods from the server but has more classes than the server, fx. fetching lyrics from the internet.



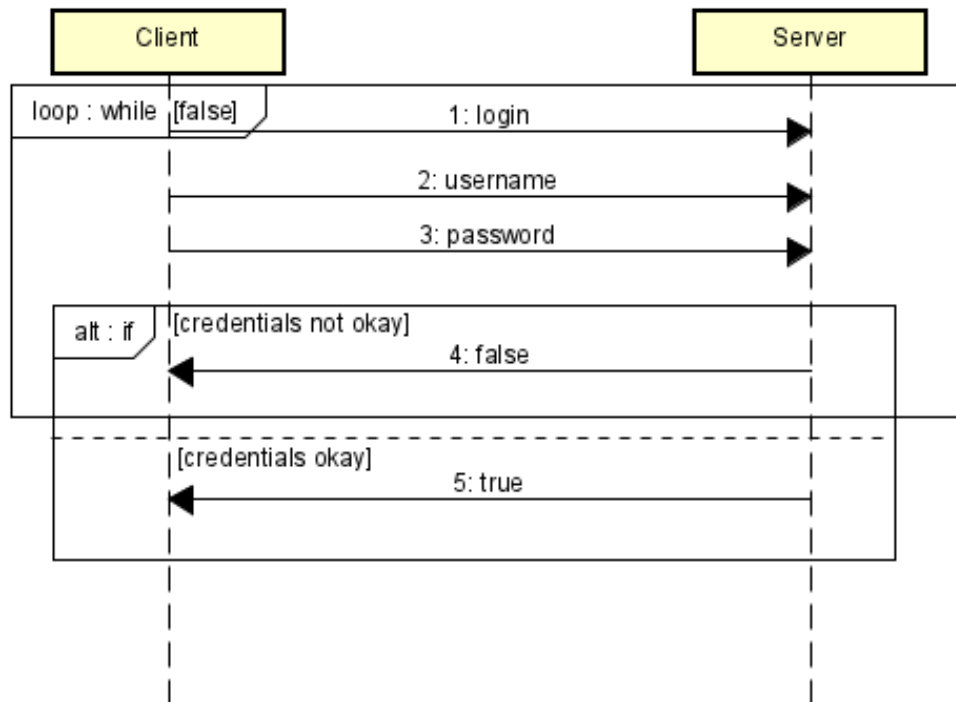
The sockets and the resources package look like this:



The socket part has specialized parts for every action that the client can take and on top of that the client does not need to stay connected continuously, so the client socket will connect to the server's socket and after the information was received the client will close the connection and it will open a new one when there is a new request.

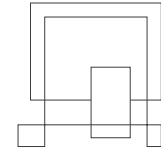


The socket protocol for logging in is the following

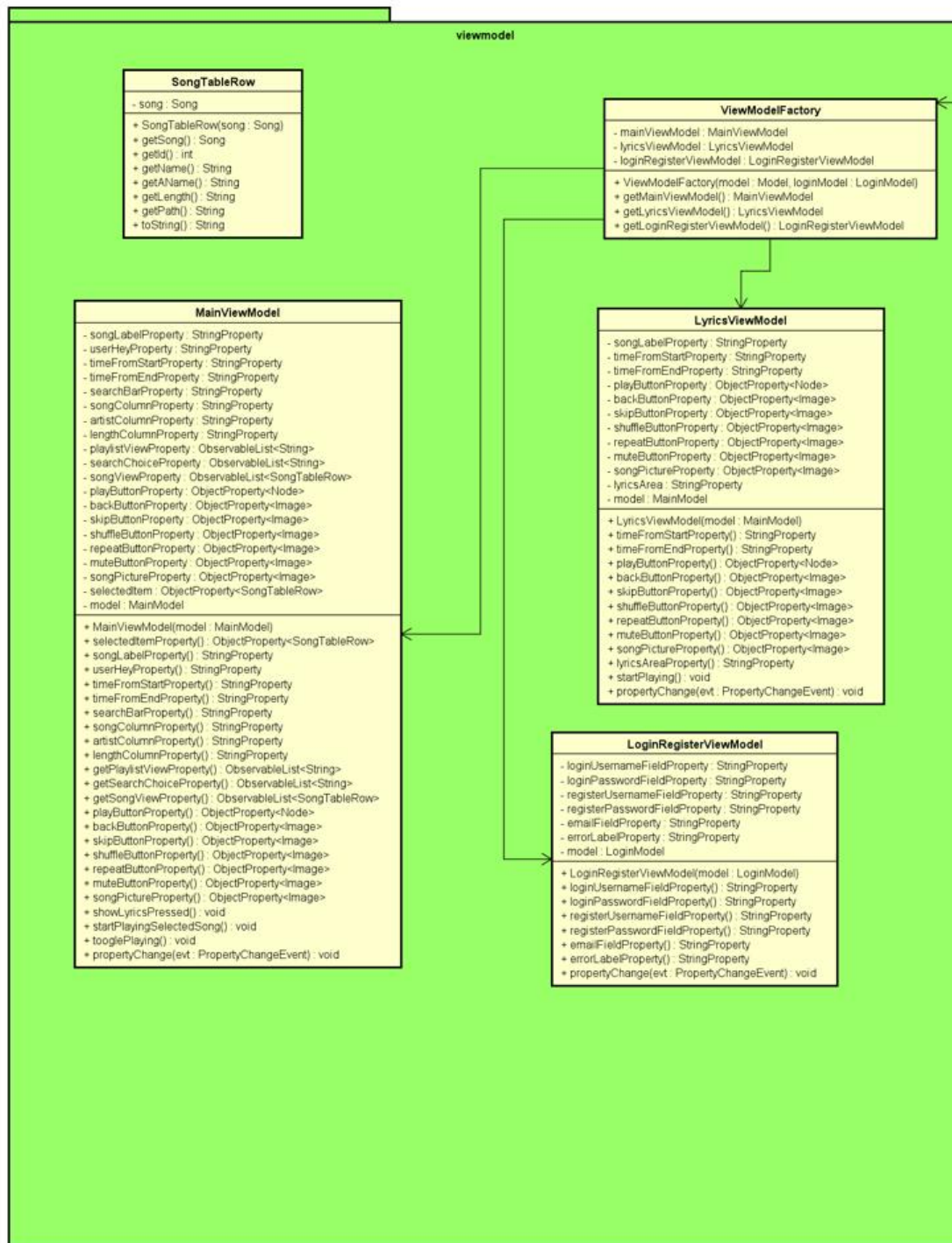


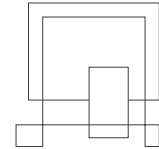
When the client model receives the answer true, it will open the next window for client to start selecting songs and using the app properly.

The resources package contains everything related to the lyrics API and the music player, methods that were implemented from a jar file.



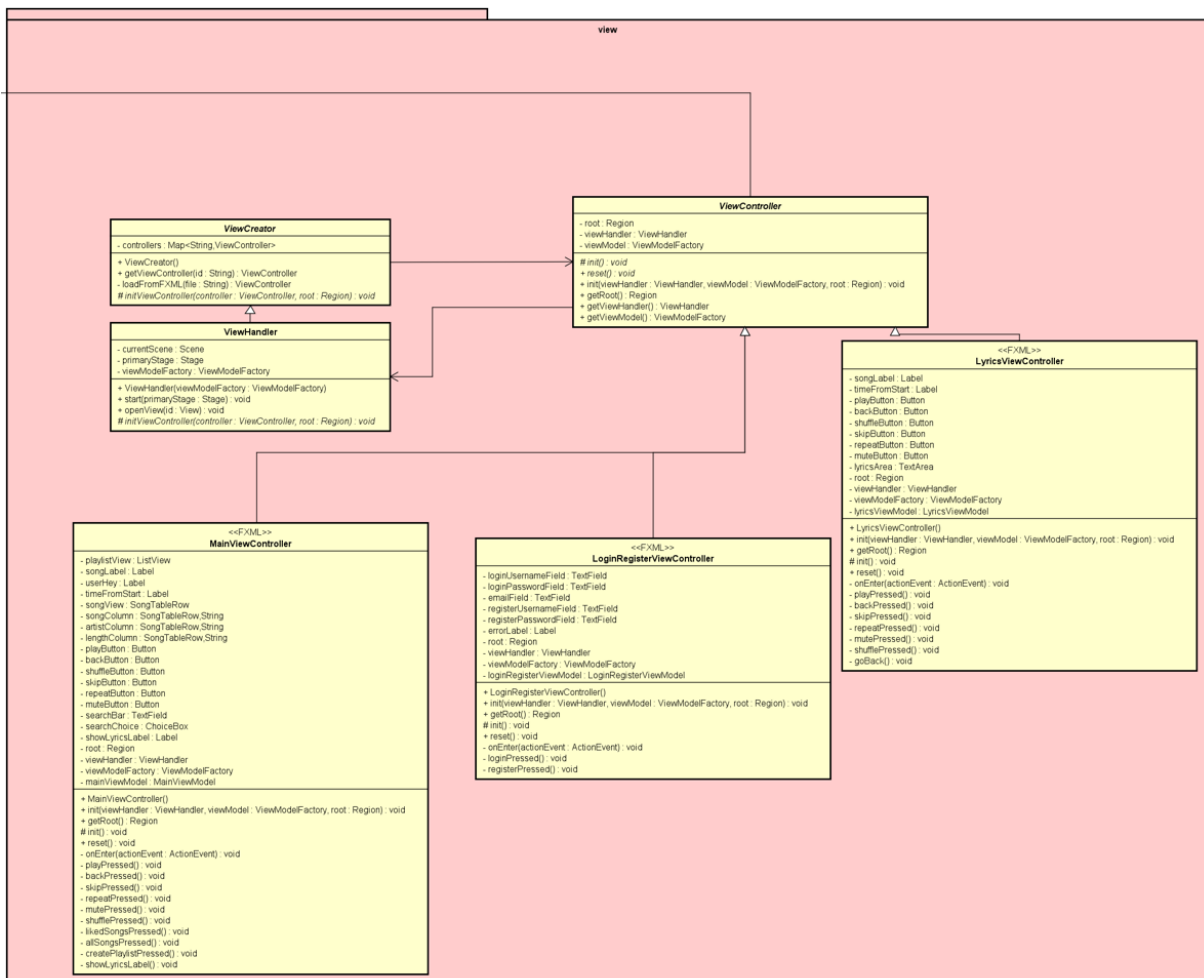
Now we have the viewmodel package:



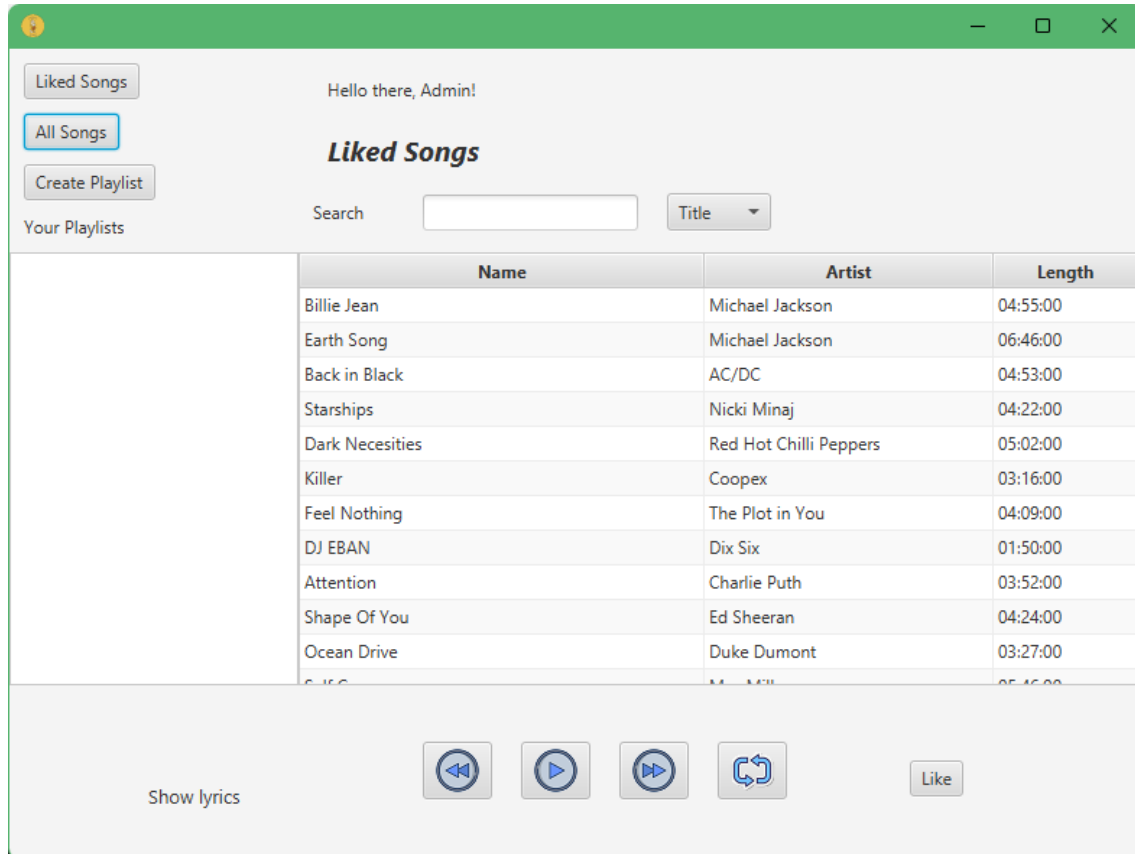
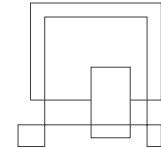


The viewmodel is not very complicated, it has indeed a lot of methods in those classes but the way it was designed in MVVM makes it very easy to navigate and make changes. The methods that do not return any properties are just calling for other methods in the model. So the complexity should not be overwhelming.

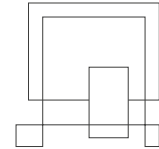
The view package was created with the factory method design pattern:



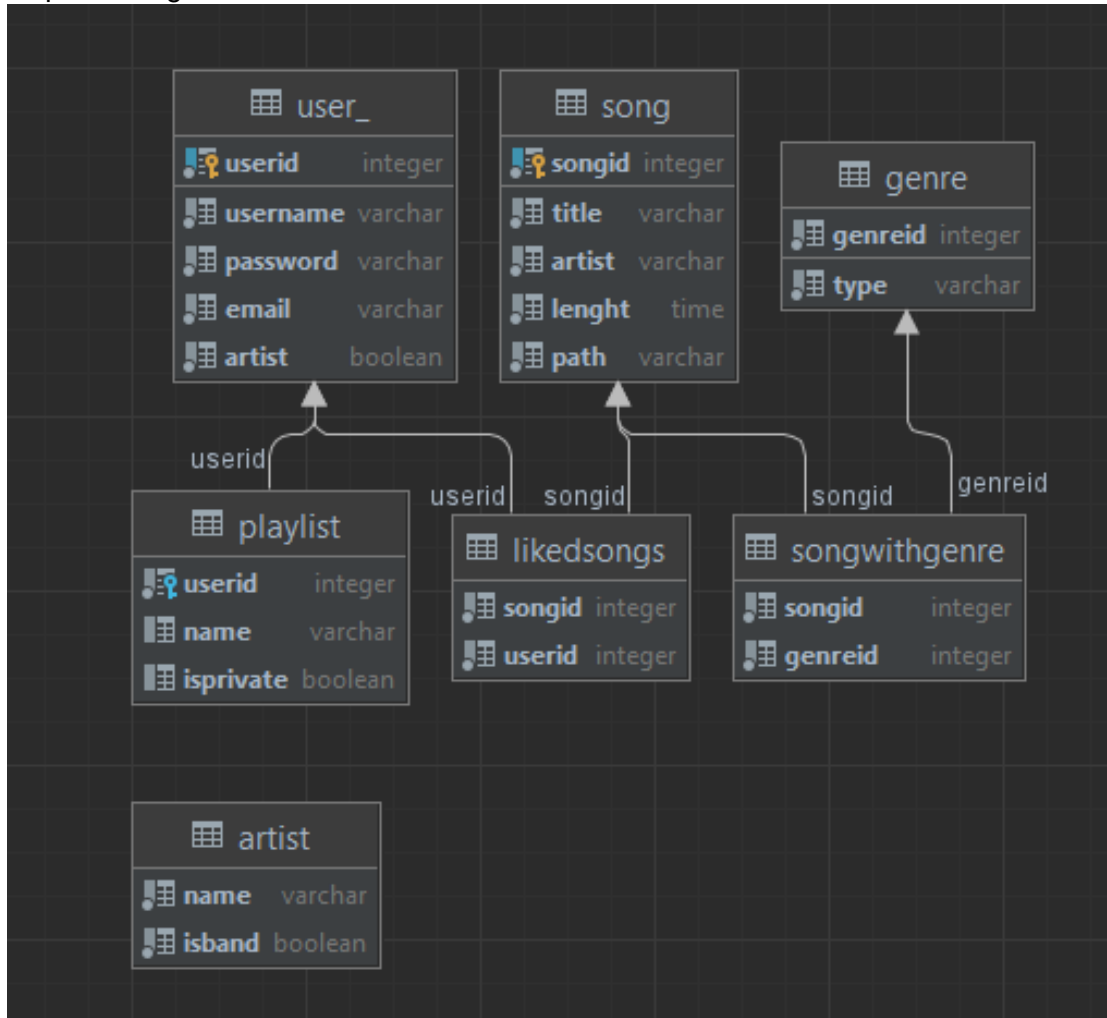
The view package was the least package that was worked on, since the factory method was done in no time, the MVVM design made it such that we code the logic in the viewmodel so, in terms of the view controllers, it was needed just to bind the properties, and afterwards just call methods on the viewmodel.



This is the interface used for the main window, it contains a lot of functionalities for the user to use. The show lyrics label is a pressable label and it will open a lyrics tab window if a song was selected first, more about this use case on the previous section.

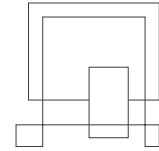


Representing the database:



There are playlist, liked songs, genre, songwithgenre and artist the table do not contain any information, the only entities that have information is the user table and the song.

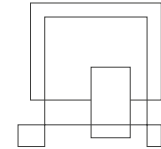
Speaking of the database the client does not interact directly with the database but instead makes a request to the server which then interacts with the database. The client does not know anything about the database.



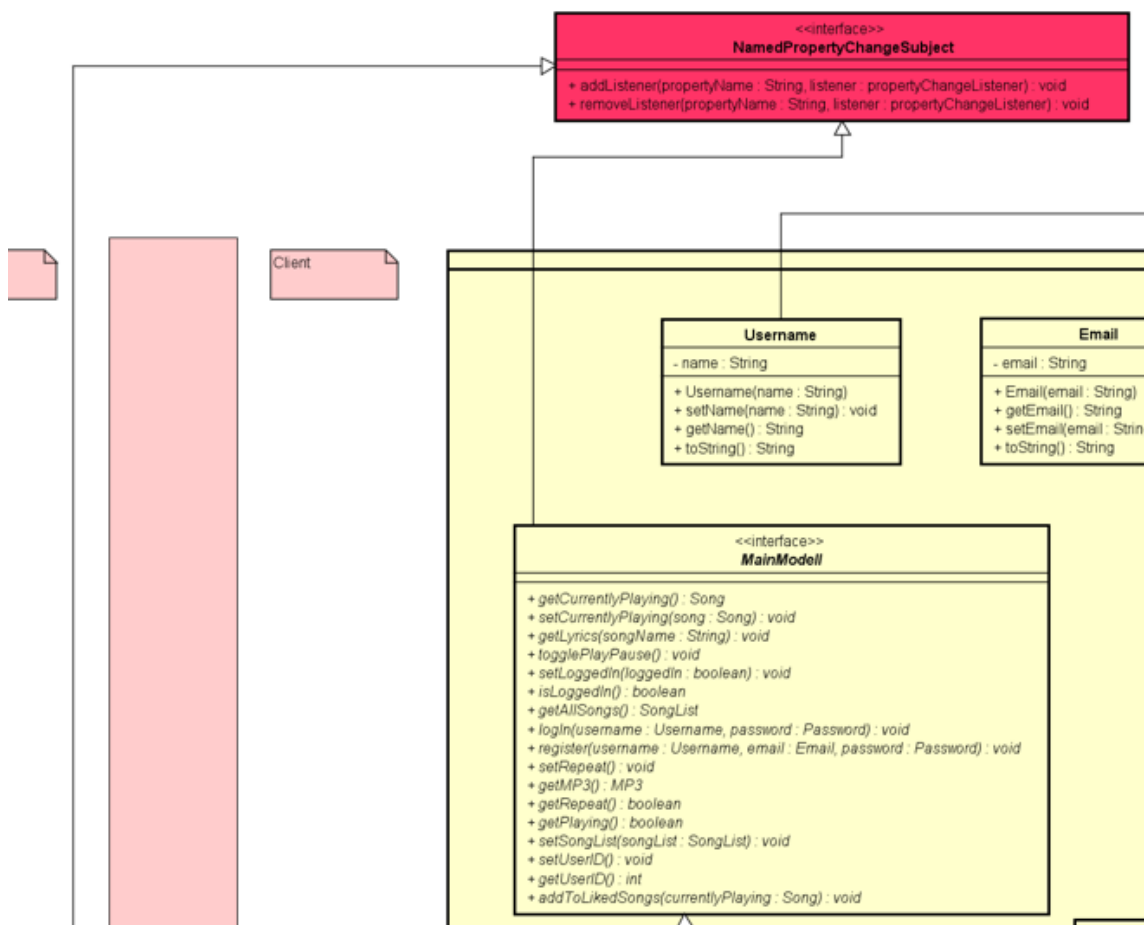
A code snippet representing the interaction of the server with the database

```
public SongList loadAllSongsFromDatabase() throws SQLException {
    DriverManager.registerDriver(new org.postgresql.Driver());
    Connection connection = DriverManager.getConnection( url: "jdbc:postgresql://localhost:5432/postgres", user: "postgres", password: "981230");
    PreparedStatement setSchema = connection.prepareStatement( sql: "set schema 'musicdata'");
    setSchema.executeUpdate();
    try {
        PreparedStatement numberOfSongs = connection.prepareStatement( sql: "SELECT count(*) FROM song");
        ResultSet count = numberOfSongs.executeQuery();
        while (count.next()) {
            for (int i = 1; i <= Integer.parseInt(count.getString( columnIndex: 1)); i++) {
                PreparedStatement statement = connection.prepareStatement( sql: "SELECT * FROM song WHERE songid=" + i + ";");
                ResultSet set = statement.executeQuery();
                while (set.next()) {
                    Solo artist = new Solo(new User( userID: 1, new Username("Michael Jackson"), new Email("michaeljackson"), new Password("password")), set.getString( columnIndex: 3));
                    String path = set.getString( columnIndex: 5);
                    System.out.println(path);
                    Song song = new Song(set.getInt( columnIndex: 1), set.getString( columnIndex: 2), artist, set.getString( columnIndex: 4), set.getString( columnIndex: 5));
                    songList.addSong(song);
                }
            }
        }
    } catch (SQLException e) {
        throw new RuntimeException(e);
    } finally {
        connection.close();
    }
    return songList;
}
```

This code will connect to the database, it will request the number of songs available, and it will request depending on the count of song all the songs that exist on the database. At the end of the method, it is returning a songList object which then is passed to the client once it has been requested.



Another design pattern that was implemented was the observer pattern which was used to display the timer in the GUI and display the song name in the label:

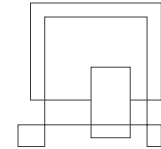


Throughout the project there were multiple JAR file used to accomplish this, and they are the following:

- jaco-mp3-player-0.9.4.jar
- MyObserver-1.4.jar
- postgresql-42.2.11.jar

As well as javaFX:

- javafx-sdk-17.0.1

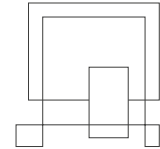


5 Test

The approach chosen for testing the implemented system was black boxing, which means that the testing was focused more on the output of the program and methods rather than the way they achieve the output.

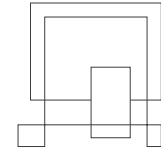
It was use JUNIT testing for both the client side and the server side thus making sure that the use cases are working as expected.

Use Case	Expected Result	Works as expected?
Sign up	The user that fills the required fields to register and are not present in the database will be added to the database and redirected to the main window.	Yes
Log In	The user that fills required fields to log in and are present in the database will be redirected to the main window.	Yes
Play music	The user that login/registers correctly, presses the “All songs” button, selects a song and plays it.	Yes
See lyrics	The user that login/registers correctly, presses the “All songs” button, selects a song and plays it the presses the “show lyrics” label will open the lyrics window.	Yes
Search song	The user that login/registers correctly, presses the “All songs” button, selects what it wants to search for and types in the search bar the name of song/artist will get the song.	Yes

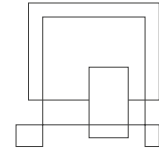


6 Results and Discussion

Product backlog				
14/23 completed				
✓	ID	Priority	Hours estimate	Task
TRUE	1	Critical	72 hours	As a user I want to be able to play and listen to music in a console environment.
TRUE	20	Critical	18 hours	As a user I want to be able to play and listen to music in a program window.
TRUE	2	Critical	36 hours	As a user I want to be able to see the lyrics of the song that I am listening to.
TRUE	3	Critical	36 hours	As a user I want to be able to pick a song from a list.
TRUE	4	Critical	18 hours	As a user I want to be able to pause or resume the song.
TRUE	21	Critical	32 hours	As a user I want to have multiple computers playing from the same list of songs.
TRUE	5	Critical		As a user I want to be able to change the volume of the song. (Canceled by the product owner)
TRUE	22	High	6 hours	As a user I want to be able to run the program by having a shortcut on my desktop.

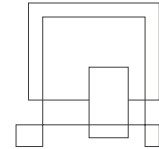


TRUE	6	High	10 hours	As a user I want to go to the next or previous song.
TRUE	7	High	10 hours	As a user I want that after a song finishes, another one will start playing automatically.
FALSE	8	High	36 hours	As a user I want to be able to have a liked songs playlist, for a more personal experience.
TRUE	9	High	10 hours	As a user I want to be able to search for a song.
TRUE	10	High		As a user I want to have the songs categorized by genres.
TRUE	11	Medium		As a user I want to be able to shuffle or repeat my playlist.
FALSE	12	Medium		As a user I want to get song recommendation.
FALSE	23	Low	2 hours	As a user I want to know how many hours I spent listening to music.
FALSE	13	Low		As a user I want to be able to change font size, color and style of the lyrics.
FALSE	14	Low		As a user I want additional information about the song like the author, length, and year.
FALSE	15	Low	36 hours	As a user I want to have a separate account to not merge my liked songs with another user.



TRUE	16	Low	2 hours	As a user I want to be able to toggle between showing lyrics and not showing lyrics.
FALSE	17	Non-Functional		As a user I want to see a playlist with the most listened songs.
FALSE	18	Non-Functional		As a user I want to see my activity, for example how many hours I have listened today.
FALSE	19	Non-Functional		As a user I want the app to go to sleep after some inactivity.

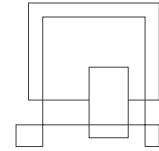
At the end of sprint 5 the development team is proud to deploy a functional and usable program.



7 Conclusions

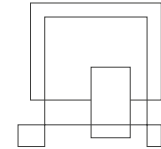
Giving the time given to the development team they managed in the course of 5 sprints to complete more than half of the requirements. The development required many hours of research when it comes to how to send songs through sockets, how to fetch lyrics, how to design the system and how to make everything work together and be satisfactory for the Product Owner as well.

The team is proud that despite the short time for development, the critical and all but one high priority requirement were met.



8 Project future

From a technical perspective the team would like to work with a protection proxy for the login function. Other than that like any other team it was suggested lots of optimisation when it comes to sending the song and to playing the song.



9 Sources of information

Note: Use the standard reference method: Harvard Anglia. A very good reference tool is Mendeley (Mendeley.com, 2016), ask VIA Library if you need help.

Banger, D., 2014. A Basic Non-Functional Requirements Checklist « Thoughts from the Systems front line.... Available at: <https://dalbanger.wordpress.com/2014/01/08/a-basic-non-functional-requirements-checklist/> [Accessed January 31, 2017].

Business Analyst Learnings, 2013. MoSCoW : Requirements Prioritization Technique — Business Analyst Learnings. , pp.1–5. Available at: <https://businessanalystlearnings.com/ba-techniques/2013/3/5/moscow-technique-requirements-prioritization> [Accessed January 31, 2017].

Dawson, C.W., 2009. Projects in Computing and Information Systems, Available at: http://www.sentimentaltoday.net/National_Academy_Press/0321263553.Addison.Wesley.Publishing.Company.Projects.in.Computing.and.Information.Systems.A.Students.Guide.Jun.2005.pdf.

Gamma, E. et al., 2002. Design Patterns – Elements of Reusable Object-Oriented Software, Available at: http://books.google.com/books?id=JPOaP7cyk6wC&pg=PA78&dq=intitle:Design+Patterns+Elements+of+Reusable+Object+Oriented+Software&hl=&cd=3&source=gbp_api%5Cnpapers2://publication/uuid/944613AA-7124-44A4-B86F-C7B2123344F3.

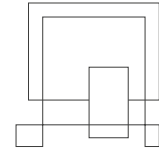
IEEE Computer Society, 2008. IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation,

Larman, C., 2004. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development,

Mendeley.com, 2016. Homepage | Mendeley. Available at: <https://www.mendeley.com/> [Accessed February 2, 2017].

YourCoach, S.M.A.R.T. goal setting | SMART | Coaching tools | YourCoach Gent. Available at: <http://www.yourcoach.be/en/coaching-tools/smart-goal-setting.php> [Accessed August 19, 2017].

Send|receive|play music files through network. Available at:(Anon., 2022g)



Java Socket Example for sending and receiving byte array. Available at:
(Anon., 2022h)

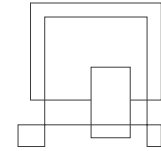
JavaFX, table view populated but text is blank/hidden/not visible, solution. Available at:
(Anon., 2022f)

Making a search bar in javafx. Available at:(Anon., 2022d)

Playing .mp3 and .wav in Java. Available at: (Anon., 2022a)

Bind TableView items with ObservableList in FXML. Available at: (Anon., 2022e)

Binding Image in Javafx. Available at: (Anon., 2022c)



10 Appendices

Appendices can be found inside of the APPENDICES folder inside of the handed-in .zip file.

They are structured by folders in the following manner:

APPENDIX FOLDER 1 – DOCUMENTS

- APPENDIX 1A – Project Description
- APPENDIX 1B – Analysis Document
- APPENDIX 1C – Sprint Backlog
- APPENDIX 1E – Chad Music Client Tutorial
- APPENDIX 1D – Chad Music Server Tutorial
- APPENDIX 1F - USER STORIES

APPENDIX FOLDER 2 – ASTAH

APPENDIX SUBFOLDER 2A – ACTIVITY DIAGRAMS

- APPENDIX 2Aa - Like a song Activity Diagram
- APPENDIX 2Ab - Login Activity Diagram
- APPENDIX 2Ac - Next Previous song Activity Diagram
- APPENDIX 2Ad - Play Pause Activity Diagram
- APPENDIX 2Ae - Register Activity Diagram
- APPENDIX 2Af - Repeat a song Activity Diagram
- APPENDIX 2Ag - Search bar Activity Diagram
- APPENDIX 2Ah - Show lyrics Activity Diagram

APPENDIX SUBFOLDER 2B – CLASS DIAGRAM

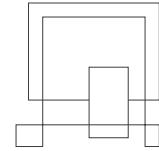
- APPENDIX 2Ba - Class Diagram CHADMUSIC

APPENDIX SUBFOLDER 2C - DOMAIN MODEL

- APPENDIX 2Ca - Domain Model

APPENDIX SUBFOLDER 2E – SEQUENCE DIAGRAMS

- APPENDIX 2Ea - Login sequence diagram
- APPENDIX 2Eb - Lyrics sequence diagram
- APPENDIX 2Ec - Playing sequence diagram
- APPENDIX 2Ed - Register sequence diagram



APPENDIX 2Ee - Searchbar sequence diagram

APPENDIX 2Ef - Server-Client login

APPENDIX 2Eg - Server-Client register

APPENDIX 2Eh - Server-Client song

APPENDIX 2Ei - Server-Client songList

APPENDIX SUBFOLDER 2F - USECASE DIAGRAMS

APPENDIX 2Fa - ChadMusic-WholeSystem

APPENDIX 2Fb - Liking a song UseCase Diagram

APPENDIX 2Fc - lyrics UseCase Diagram

APPENDIX 2Fd - pause UseCase Diagram

APPENDIX 2Fe - Register UseCase Diagram

APPENDIX 2Ff - Repeat a song UseCase Diagram

APPENDIX 2Fg - Search for a song UseCase Diagram

APPENDIX 2Fh - UseCase Diagram for login

APPENDIX FOLDER 3 – SOURCE CODE

REQUIRED-FILES

SEP-Project-INTELIJ

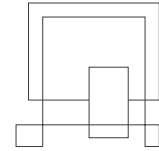
SEP-DataBase

ARTIFACTS

APPENDIX FOLDER 4 – JAVADOC

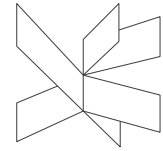
SERVER (server javadoc)

CLIENT (Client javadoc)



Appendix A Project Description

Insert the original Project Description here



NBNP

**Chiril Luncasu (315171), Dan Sebastian Ceapa (315162),
Matas Armonaitis (315263), Dragos-Daniel Bonaparte (315261)**

**Henrik Kronborg Pedersen
Steffen Vissing Andersen**

**Software Technology Engineering Semester 2
16/02/2022**

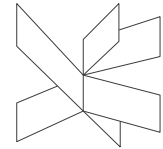
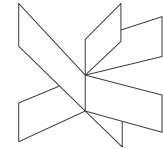


Table of content

1	Background description	1
2	Problem Statement	2
3	Definition of purpose	3
4	Delimitation.....	4
5	Methodology.....	5
6	Time schedule	6
7	Risk assessment	7
8	Sources of Information	7
	Group name: NBNP	8

Appendices (including Group Contract)

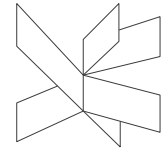


1 Background description

Today a lot of music apps strive to achieve a better music quality (Anon., 2022b), rather than work on the functionalities of the said apps. Studying the feedback of various music app users, we concluded that a lot of those apps do not meet the expectations when it comes to functionalities like display of lyrics and highlight them, sound design, shuffle playlist, putting the app to sleep when not used, download music/listen offline means of using playlist functionalities in general and if some of those features are around, they come at a cost of a monthly membership.

As a general user it is hard to find apps that have lyrics for all the songs and to be able to change them, a good example will be Spotify (Anon., 2022d) which recently added lyrics to their app, but unfortunately not for all the songs and also without a monthly membership the app is very bloated by ads and some features are disabled. Tidal, for example, comes with a lot of features but because of that, those require a monthly membership to be able to use the app. None of the mentioned apps have lyrics when in offline mode.

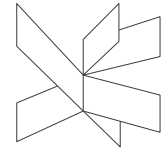
Those things don't go very well with the youth since it is the biggest consumer audience for music. Because of the very young consumers, not all of them want or have the means to pay to get specific or all the features, and some of them might find it inconvenient or hard to use. Moreover, it became more trending on social platforms to lip-sync or sing along to their favorite songs, doing karaoke and the ability of the music app to provide highlighted lyrics for the song without copyright strike might be a strong factor when choosing a daily driver app for music.



2 Problem Statement

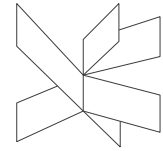
The user want to play music at a respectful quality, organize the music in playlists, and also provide every song with lyrics.

- How to make users to listen music?
- How to convince people to use?
- How to deal with copyright?



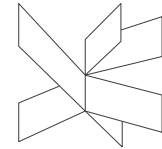
3 Definition of purpose

The purpose is to help the users get better satisfaction when people want to do a karaoke night with friends.



4 Delimitation

- We will not include choose quality of the song
- We will not include a function to change lyrics
- We will not make the music downloadable
- We will not highlight lyrics while the song plays



5 Methodology

As methodology we will use Scrum (Anon., 2022c) since it's a very optimal way of organizing workflow, team tasks, and also this will ensure that we always have a done product at hand in case the deadline is close. Our process will be based on UP methodology and Scrum. In the inception phase we will prepare basis for the project and possible architecture solution together with design trade-offs, identifying risks, and coding of initial project plan. In our elaboration phase our product owner will provide us with the needed requirements and use cases in order to reduce their impact on final product. In the construction phase the design of the system is finalized and refined and the system is built using the basis created during elaboration phase. The construction phase is divided into multiple sprints, for each sprint to result in an executable release of the system. The final sprint of construction phase releases fully completed system which is to be deployed during transition phase. And the final phase transition where the final project phase delivers the new system to its end-users.

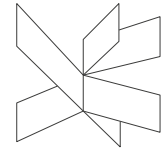
With scrum the process is broken up in smaller pieces compared to waterfall

First when we start creating the product,

- we do just enough planning to get started with (the basic functionality/features)
- we build what was planned
- we test and review and get ready to ship (to ship means that it's basically bug free in that state and can be delivered to the customer as it is)
- when this cycle is complete, we end up with a potentially shippable product

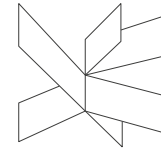
Our project team will be consisted in:

- one product owner – Dan Sebastian Ceapa
- one scrum master – Matas Armonaitis
- two developers – Dragos-Daniel Bonaparte and Chiril Luncasu



6 Time schedule





7 Risk assessment

Risks	Likelihood Scale: 1-5 5 = high risk	Severity Scale: 1-5 5 = high risk	Product of likelihood and severity	Risk mitigation e.g. Preventive- & Responsive actions	Identifiers	Responsible
Logical flaws	3	5	15	Coding in pairs, and plenty of testing.	Logical flaws – the program does not work as intended.	Dragos and Matas
Database problem	3	5	15	Having a backup database with essential information for the program	Program cannot fetch information from the database	Chiril and Dan
Copyright	4	4	16	In the program mention the creator of the song and make the music not downloadable	We get email about copyright strike	Matas

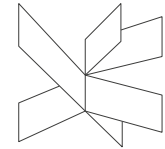
8 Sources of Information

Craig Larman Applying UML and Patterns –An Introduction to Object Oriented Analysis and Design and Iterative Development. Third Edition, 2004 Addison Wesley Professional ISBN: 0-13-148906-2 Connolly & Begg:

Database Systems: A Practical Approach to Design, Implementation, and Management Scrum: (Anon., 2022c)

Background Description resources: (Anon., 2022b, Anon., 2022d, Anon., 2022e, Anon., 2022a)

Unified Process: (Osis and Donins, 2017)



Anon. 2022a. *Anyone else think Spotify is too expensive?* : *spotify*. [online] Available at: <https://www.reddit.com/r/spotify/comments/mqkdh/anyone_else_think_spotify_is_too_expensive/> [Accessed 2 March 2022].

Anon. 2022b. *Best music streaming services 2022: free streams to hi-res audio | What Hi-Fi?* [online] Available at: <<https://www.whathifi.com/best-buys/streaming/best-musicstreaming-services>> [Accessed 2 March 2022].

Anon. 2022c. *Introduction to Scrum - 7 Minutes - YouTube*. [online] Available at: <https://www.youtube.com/watch?v=9TycLR0TqFA&ab_channel=Uzility> [Accessed 2 March 2022].

Anon. 2022d. *Spotify Revenue and Usage Statistics (2022) - Business of Apps*. [online] Available at: <<https://www.businessofapps.com/data/spotify-statistics/>> [Accessed 2 March 2022].

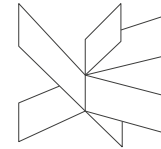
Anon. 2022e. *TikTok User Statistics (2022)*. [online] Available at: <<https://backlinko.com/tiktok-users>> [Accessed 2 March 2022].

Osis, J. and Donins, U., 2017. Software Designing With Unified Modeling Language Driven Approaches. *TopUML Modeling*, pp.53–82. <https://doi.org/10.1016/B978-0-12805476-5.00002-2>.

Appendices


Group contract

Group name: NBNP

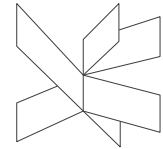




The following terms/rules have been discussed and agreed by all of our members regarding cooperation and conflicts that may occur on the way:

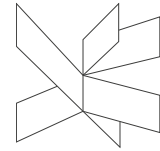
- Participation: We agree to participate in all the tasks, socialize with each other and feel good while working.
- Communication: Mainly on discord and physical meetings, alternatively we use messenger and ultimately phone calls.
- Meetings: We agree to meet on every Wednesday with some exceptions and with consent from every member we shall meet at least one more time per week.
- Conduct: We agree to be active in the group, willing to work and finish tasks before the deadline. We shall vote every idea of any members to have the satisfaction of being heard.
- Conflicts: If we get into argues, we shall end the conflict at once and stop it from escalating.
- Deadlines: Every member agreed to be active and respond before each deadline with their work.
- Consequences:
 1. If a member of the group doesn't show up to the group meeting more than 3 times without explanation, he buys drinks to everyone.
 2. If a member doesn't want to work, he gains one warning, and if he manages to get 3 warnings, the group can talk with the supervisor in order to kick him out from the group.
 3. If the member of the group is rude to other members without any valid reasons, after 3 warnings, he buys cake.

Group member's name	Student ID	Signature
Chiril Luncasu	315171	

Project Description - VIA Engineering



Matas Armonaitis	315263	
Dan Sebastian Ceapa	315162	
Dragos-Daniel Bonaparte	315261	



SEP2Y Group 2 Final Project PROCESS REPORT

Dragos Daniel Bonaparte, 315261

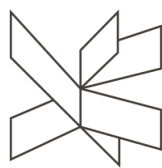
Dan-Sebastian Ceapa 315162

Matas Armonaitis, 315263

Chiril Luncasu 315171

Supervisors:

Steffen Andersen (SVA), Henrik Kronborg Pedersen(HEKP)



VIA University
College



[Number of characters]

Software Technology Engineering

2nd Semester

24.05.2022

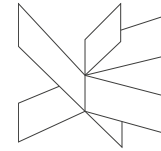
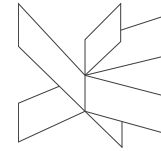


Table of contents

1	Introduction.....	1
2	Group Description.....	2
3	Project Initiation.....	8
4	Project Description.....	10
5	Project Execution	11
6	Personal Reflections.....	13
6.1	Dragos-Daniel Bonaparte	13
6.2	Chiril Luncasu	14
6.3	Matas Armonaitis	16
6.4	Dan-Sebastian Ceapa.....	17
7	Supervision.....	18
8	Conclusions	19



1 Introduction

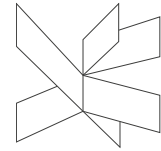
Group 2 from class Y, composed of Chiril Luncasu, Dragos Daniel Bonaparte, Dan-Sebastian Ceapa and Matas Armonaitis worked together throughout their 4 sprints in the second semester, spring 2022, for the course SEP2.

A recollection of their meetings throughout the Sprints can be assessed in the following table:

27 APRIL – 24 MAY

27/04-29/04	Met from 09:00 - 12:00
30/04-01/05	WEEKEND
02/05-06/05	Met from 09:00 - 12:00
07/05-08/05	WEEKEND
09/05-13/05	Met from 09:00-12:00
14/05-15/05	WEEKEND
16/05-20/05	Met from 09:00-12:00
21/05-22/05	WEEKEND
23/05-24/05	Met from 09:00-12:00

The Sprint Backlog for this group can be found in more detail in APPENDIX 1C Sprint Backlog. We had multiple supervisor meetings with Henrik and Steffen, and email conversations where we got help with our questions about our project, our motivation and working as a united team. The meetings we had were short 20-30 minutes meetings, that happened within the time frame 9:00 – 12:00.



2 Group Description

For our four-person group, we have some diversity regarding cultural background, experience, and people's characters in general. We have 3 students from Romania: Dragos, Chiril, and Dan, and 1 student from Lithuania: Matas. All of us have different views on the world of programming due to our prior experiences with university exercises and experience before it. When it comes to roles in the group, SCRUM for example, we tried to choose the person that fit the best!



**DRAGOS DANIEL
BONAPARTE**

Nationality: Romanian

SCRUM Roles: Developer

Fun Fact: Can repair/make anything work

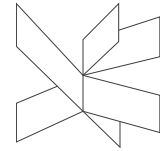


MATAS ARMONAITIS

Nationality: Lithuanian

SCRUM Roles: SCRUM master,
Developer

Fun Fact: Best bassist around



Chiril Luncasu

Nationality: Romanian

SCRUM Roles: Developer

Fun Fact: Can make you smile at any time

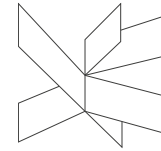


Dan – Sebastian Ceapa

Nationality: Romanian

SCRUM Roles: Product Owner,
Developer

Fun Fact: Has a lot of luck



Personal Profiles

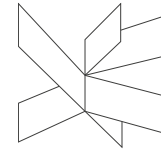
Our humble group members are very different when it comes to their character type, but when deciding our SCRUM roles, we made sure that we consider character traits and characterization of the person about themselves. We chose Matas as SCRUM master because of his friendliness, Drive to bond and nourish relationships with people, and the capability to make harsh decisions when needed, which was an easy task. But when we were looking for a Product owner we had problems, whether we would like a product owner that is very strict, or we would like a product owner which is very value base or if we wanted a product owner which had a very big drive for socialization. In the end we decided that Dan would be the best fit because of his rational thinking when it comes to his beliefs and points of view, and his drive to make a fun project which will give us an opportunity to learn a lot from.

All the members had the responsibility of being team members as well, so nobody got left out. A little lower you can see the personal profiles that can back up our claims about each of developer's personalities and traits that helped us decide their roles in the group. However, sometimes we didn't follow the personal profiles, thinking our judgement will be much more efficient. For example, at the start of group work we had issues with authority in our group, we didn't know how we should go about doing tasks. But we overcame the problem by showing some boldness and some group members tried to seek status which solved our problems. But, by just looking at our personal profiles that indicate a lack of motivation for prestige, influence, a fear of appearing weak and losing influence we would start working as a better team from the start.

Our team complements each other perfectly because we have team members that mostly focus on values, teamwork, quality, and harmony, while having members that could fill in for the missing qualities, because everyone is so diverse. Not only we had people who focus on results, goals and mostly initiate all the good ideas, but also, we had people that are motivated by variation, and focus on ideas and are just the most creative and Inspiring members of the team.

While making we didn't set any boundaries for laziness or time frames, which made our group suffer. After looking at our profiles we made sure that we put clear boundaries like penalties for not coming to meetings because in our personal profiles we didn't have a lot of people that show influence.

In a nutshell, we really think we were spot on when we chose our SCRUM roles, and we were diverse enough to show qualities that would otherwise be missing from our group. We focused on a friendly atmosphere that tried to get everyone involved and feel

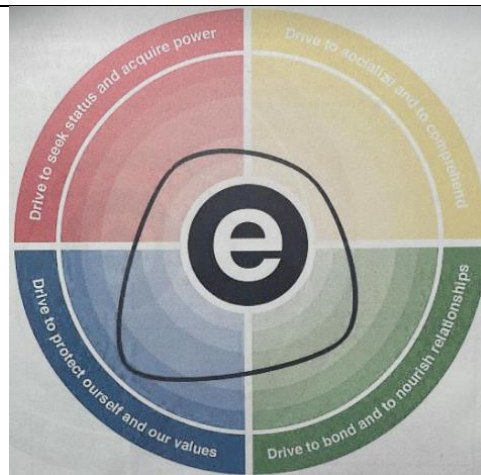


appreciated, while keeping some formal rules to make sure we have boundaries and discipline.

Matas Armonaitis



Dan-Sebastian Ceapa

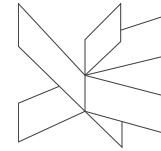


Dragos Daniel Bonaparte



Chiril Luncasu





Cultural Background

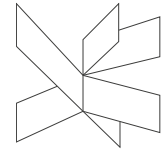
As a group we feel like our cultural backgrounds are very similar, not only because we have 3 members that have the same nationality, but because of the history our countries share, and the way our communities back home shaped us.

First, because our cultural backgrounds are very similar, we didn't really have a lot of challenges heading into the project period because we already understood how everyone communicates, understands, makes decisions and we had enough trust to get job done without any major conflicts. Another strength would be the fact that some of our group members knew a lot about the cultures of both countries, so we had people that made sure everyone was on the same page, acting as human translators if they saw that the situation wasn't clear, or someone didn't understand what they had to do properly.

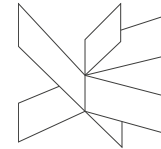
On the other hand, there are some visible minuses that we could point out as well. Because we come from the same background, we had no room for new experiences that could make our work more productive. For example, if someone came from a community that had less strict rules, or rather harsher rules, they would have a "fresh look" on the way we were working. That would greatly help us with working at our jobs in the future or next semesters, but sadly we didn't get the wanted outcome.

The most relevant dimensions in our group are surely Disagreement, Communication and Leading. Our cultures are very direct when it comes to communication. When we disagree with something we will be mostly confrontational, which makes it perfect for things we disagree with. We would list out what we don't like about it in a very open manner and then we would continue our task without having to avoid or try to mask our preferences. If we mention communication, we are mostly low-content, but when someone wants us to elaborate on a problem we can switch up to high-context communication easily. This helps a lot, because we are very direct when talking about the task, we are willing to do, about the plans we have or group activities in general. Lastly but not least, leading is a big part of the communication in our group, we are always keeping it very egalitarian, mainly through votes, and letting democracy decide what we are going to do. In our eyes every group member has their opinion, and everyone should speak up when they don't agree with a decision, but when it comes to implementing it, we try to make everyone happy by having a vote where the majority rules. Everyone is happy with it; thus, it works nicely.

A lot of times when we tried to solve a problem, we tried to be as direct as possible, confrontational at times with decision we didn't like and keeping it equal with a simple vote system that everybody agrees with.



We were already on the same page when it came to communicating, but the Culture Map let us achieve a better understanding of the fact that of the people have a different cultural background which doesn't look like our but ultimately can be understood with a little bit of effort.



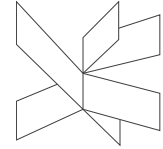
3 Project Initiation

Our group was formed out of the comradery shown towards each other. Dragos and Matas are good friends that had a very good experience with SEP1 but their group members left them to form another group with other people which they didn't take to heart, but they needed 2 other members. Because Chiril and Dan had a very bad experience with SEP1 they were looking for a new team. Dragos knew Chiril was looking for a team, so he proposed that the 4 of them unite forces for a new and hopefully better team, forming NBNP.

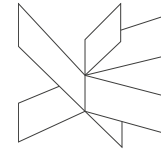
NBNP is the name of the group. It stands for "No Brain No Pain" – which is a joke that ran in the group was nominated as the name. Everyone was so active and happy to work with each other that a lot of jokes, happy moments and fun times occurred. It should be noted that we used a lot of our brain power for this project, so the name should be taken as a joke!

Our customer requested an app that would play songs in playlists that you could search up in the app and show the lyrics for the songs. The customer also required that he could create playlists and have users that had those playlists. Some features as the changing of the volume of the song in the app were reconsidered by the product owner, so we didn't have to implement them into the final version.

If we talk about "planning" for the project, we were very serious about research, and we did a lot of it even before we started doing sprints. All our team was determined to find a way to play songs and pair it up with transferring data from the database, so we were very fond of the project at the start. We started using UP at it's fullest by trying to go through Inception while modeling the best app we could while fulfilling the requirements, not forgetting about the SCRUM methodology that was given to us. We were very fond of the product owner working together with the SCRUM master to make sure we get the



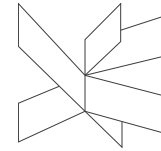
best result as soon as possible. Our burndown chart shows that we were slow in the beginning, but we were making sure that we are covering all bases at that time. Sadly, we were very demoralized because of the project state we were in, and we didn't feel like we were progressing, but after having a talk with our supervisor some group members tried inspiring the others and we started working passionately again while trying to maintain good high-content communication to end Inception very fast and easy.



4 Project Description

The group is very proud to say that our Project Description was a smooth start. We learned a lot from first semester about teamwork and how we should go about conflicts and how we should approach starting a big project. We were very happy to see that some fundamentals like Report Writing didn't change so we had an upper hand on ourselves in the first semester where we were scared because we didn't know anything.

We were set on our way to write a very good Project Description, and we finished it quite early considering that we didn't expect to finish this early. But then we had a mentor meeting where we were told that it was good, but it could have some improvements. Happily, we worked together with our supervisor to get the Project Description to a good state. Ultimately, we tried to make the product owners and our goals as clear as possible, so we had a set path, and we had a set goal in our minds. However, down the road we understood that having clear goals sometimes is very hard to work with, because we had no room for improvement on some techniques we used before, considering we weren't too fond of using the SCRUM roles, which could lead us a dark path if we didn't have a talk with our supervisor where we understood that we should be more focused on having the product owner tell us what he wants and having the SCRUM master negotiate with them.

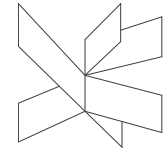


5 Project Execution

Project Execution phase was the hardest phase for the whole group. It should be mentioned that we already had burned out once due to the absence of morale in the initiation phase so we were surely pouring our souls into this to end it as soon as possible with the best quality we can provide while slowly getting into the Elaboration phase.

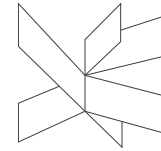
Elaboration was quite nice because we figured out a lot of the modelling by doing the project Description pretty fast as it was mentioned before, so we started working on the Analysis and Design while still keeping un with Implementation which was pretty hard because we had to design something and immediately implement it just to understand that we missed something and we had to go back and work on it. Another amazing thing that we like about Scrum is that we are allowed to go back and revisit our work to improve on it, which makes it sustainable, unlike Waterfall which had no way of going back. Sadly, we had no time for testing in Elaboration due to the slow pace we were taking, because we had to fulfill multiple critical requirements on the APPENDIX 1F user stories at the same time, which wasn't too fun, because we couldn't see any progress until later in Construction.

So, after getting through a bumpy Elaboration, we finally got into Construction where almost everything was done very surgically because we tried to make the best out of the meetings we had with our supervisors before, and the meetings between our SCRUM master and Product owner where they discussed what we can implement in time, and what their expectations should be. Our app finally started working, firstly we had some problems with it not running as it was supposed to because of some music broadcasting issues, but we achieved our goal quickly. While still modelling the database and trying to store the bare minimum of the information required, we got to a point where we could proudly say our project was finally broadcasting music from the server side to the client side on different machines, which looked very pleasing. We were still doing some



analysis and design on the way we implemented the main playlist and how we should show it to the user and how it will be played, while starting to test a lot of the stuff we did as accurately as possible, we made sure we used the ZOMBIE principles for our testing, so we have a lot of tests that will leave us with a good project without any bugs or errors(at least not any that we know of!). We were starting to deploy the program slowly but surely with multiple features added in a small span of days, just because the complexity of the project didn't allow us to add a lot of them at the start.

Finally, we got into Transition, the most fulfilling stage for all of us, which we consider is one of our best stages. We had our program working which we admired. So much that while working on the project we would play music on it rather than on other platforms, which we considered as a very big milestone. We started patching up everything, whether it came to small bugs in the final program or finishing up on the final reports (like this one!). We were in a state of bliss because we finally got through with everything and we knew we did a great job in the end. It is very important to mention that we tried hard to follow the UP Iterative Development, SCRUM roles, and approach to requirements so we had the best result by following exactly what we were told to do.



6 Personal Reflections

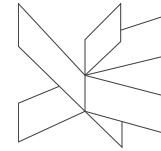
6.1 Dragos-Daniel Bonaparte

This project again showed to me that even though you come to classes daily and you do the exercises, there are still a lot of things that you need to research on before you are able to do them. The most difficult part of this project was to make the server send a song to the client. The conversion from file to bytes and back was so complicated at the beginning. In the sprint backlog this was my task and since it required a lot of research it took me a while until I was able to make it work. The MVVM design pattern has been easier and easier as the semester went by, so this was not an obstacle for me nor for my teammates.

The group contract worked like a charm again, we made sure that no one was not working but this time for us it was more complicated than usual, since 3 of us were having work twice per week in the afternoon. Fortunately, we gave up on some good sleep and woke up every morning at 8 or 8:30 and started the meeting at 9 everyday other than the weekend.

Following UP and SCRUM was a genius move since it made us work more efficient than ever. At the beginning of the project period, we started doing a project exactly like in the first semester, but after a supervisor meeting, we understood exactly what we needed to do. So, we started to code a full program every sprint, building on top of the previous sprint every single time. Our supervisor advised us to write more quality code than documenting heavily, and so we did, we gave our best efforts to achieve this app, working between work shifts, Danish classes, and personal life.

It was more than a challenge for the most of us, talking about projects with some of my classmates, everyone struggled with something, there was no team that had everything working perfectly from the beginning till end, which gave us some courage to continue, even if it meant that we might not be able to finish the program.



Looking behind at what we accomplished, I feel good about what we did, and at the same time I know that there is still so much to improve, but at the same time so much space to do so.

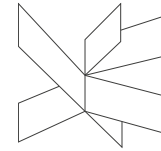
The advantage of this group is that we like to debate and laugh about a lot of stuff, we work between laughs and memes, and this atmosphere really motivates me, sometimes when the code does not work, one of us will just bring a meme out of nowhere and give you an idea or even a small smile, everything matters when stress and demotivation hits you.

The disadvantage is that we have common workdays and it was very hard in some sprints to work efficiently when all of us have a time frame of a few hours.

Everyone gave their best and everyone worked on what they knew best, and it shows in this app better than every other picture could.

6.2 Chiril Luncasu

This semester wasn't an easy task because I had a lot on my mind. I was working part time and I was a mentor, while I tried to learn and comprehend everything that was taught to us by our teachers, and I still had Danish classes that I attended. However, I can't say I was weak enough to crack under the pressure, so I just went through with it and gave it my best in every task that was assigned to me. Also, I learned a lot. I learned that time management is a very important skill I must learn to have an upper hand on the others and succeed and prove myself that I am more than just a simple student, that I can do tasks on my own. Another thing I learned is that SCRUM is much better than Waterfall, and I will most probably use it every time I will have the chance. Not only it helped me understand the project and work on it so fast and easy with an access to the tasks that I could do right there in a shared file with my group. I really love the flexibility it provides with letting you go back and work on your mistakes while keeping everything up to pace. Another thing I love is the fact that me and my teammates could fulfill tasks that they were sure they will be able to do while keeping a Sprint Backlog that would show us



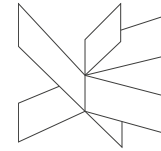
where exactly things went wrong and how far back, we are supposed to go to get to the problem.

I would lie if I'd say that the morale was at an all time high during the project, because I felt like everything I was doing wasn't enough and I started reconsidering my career choices, but after talking with our supervisors I caught my breath and was able to finally could understand that it doesn't matter how much progress we do, if we do a good job we will be victorious in the end.

Another major learned this semester is that I am not as good at programming as I thought I was. Back home I had a lot of experience, but I understood that my experience was mostly based on the practical side of programming, and I had no knowledge about theory, so when we got to the semester project, I had my first grasp on a big project that should follow theory. I was baffled and almost scarred of the fact that I sometimes didn't understand what was going on or why should we do something in a certain way if we could do it in an "easier" (at the time) way. I am very happy that I got this experience because it really humbled me, and it helped me achieve a new goal that I wanted to reach in programming. Additionally, I felt like I was learning programming from scratch again which made me recall a lot of fun and adventurous memories which I am very grateful for.

In a nutshell, I wasn't prepared for the semester project at the level I'd be happy with, and I certainly didn't understand how oblivious my logic was at the time. I had highs and lows, I went through a lot of change, and I am happy to state that I am a better programmer right now, than I was last semester, and I certainly would prefer to go through the experience of making another semester project like this!

P.S. I am not happy with the grade I got last semester and I believe I didn't sow myself in the best light, but I will try to be more accurate with my approaches and my tactics from now on! (Or in other words, I hope I get a much better grade! 😊)



6.3 Matas Armonaitis

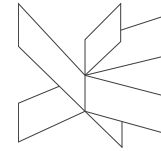
This Semester project was quite a bit different than the previous semester. In the first one there were a lot of unknowns, and everything was scary but now having experience on working in SEP it seemed like this one will be easy. Little did I know that this semester it is going to be a lot harder than the previous one. First change was that I was in a different SEP group than last semester, the only person I knew was Dragos.

Second change that was a lot different is that we started using SCRUM as our methodology. Assigning scrum roles was exciting yet scary. The third change was that we had to decide what project do we want to make. This gave us a lot of room for imagination but also there were a lot of unknowns by doing so. Even when I attended every class and did all my homework, it was still a lot of learning during the project. The music program turned out to be a lot harder to make than we first thought about it. We had problems with the server not sending songs, we had problems that the program did not work on all our computers at first.

Also, for me personally I was doing most of the testing of the program and I had a lot to learn about ZOMB+E and how to do testing in MVVM.

So, about the group work. I really enjoyed this semester working as a group. The biggest problem was that three of us had jobs in the same workplace at different days, so most of the time not all of us could be present during our meetings, but we counteracted that by working every day from 9. Yes, we did not get a lot of sleep, but this was the way that I think worked well. Everyone was doing tasks and even if they were not completed in time, they were eventually completed. We never really needed to get angry because of the amount of work we were making.

The morality of the team I think is awesome. Even in the hardest of days at least some of us would have the mood to say a joke or show a meme that would brighten our mood. Even though I was the only one not a Romanian language speaker, it did not bother me. Most of the time we spoke in English, and no one was left out at any point.



6.4 Dan-Sebastian Ceapa

This project showed me that even if we study a lot of material in class, there is a lot of research to do to do the exercise. The most difficult part of this project in my opinion was to connect the server to database. In the sprint backlog this was my task, thus it required for me to do a lot of research to make it work. The MVVM design pattern was easy going for me and my team, to do all knowledge we acquired.

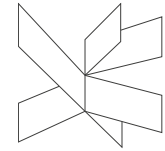
The group contract worked very well this semester, we made sure that everyone is willing to work and of course we took in consideration that 3 members from our group have work. At the beginning it was relatively hard to get in the work mood, but after some days, we got used to it. We usually started at 9:00 AM every day, beside weekends.

Following UP and SCRUM was the best idea to work more efficient. At the start of the project, we got a bit inspired by the first semester project, but after some supervisor meetings, we understood exactly what we needed to do. So, we started to code every sprint and doing our tasks. In our group we were divided by roles as we used SCRUM. For this time, I was assigned as a Product Owner, which was a big challenge for me. During the project period I understood better how SCRUM can help you so much in a team. As we didn't have a customer in order to give us requirements for the program, I was responsible to prioritize the requirements and to do the product backlog. At the end of every sprint, I looked at the requirements given to that sprint and tested how it works.

Looking behind at what we accomplished, I feel proud about what we did, but at the same time I know that there is still so much to improve.

The advantage of this group is that we could work together well, and to understand each other. We liked to have some debates on some problems in order to solve them in an efficient way. The atmosphere was very friendly, where everyone was smiling and showing some memes in order to lift up someone's mood. I can tell that it was like harmony in our group, and it really helps you to work at your maximum potential.

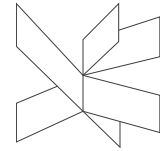
The disadvantage is that some colleagues have common workdays and it was hard in some prints to divide tasks that everyone is able to manage to finish it in time.



Everyone gave their best and everyone worked on that they know they could handle it, and this we tried to show in our app, because we really loved it and made it with passion.

7 Supervision

We can't stress enough how much we achieved just by talking to both supervisors. Whether it was a question about programming, design, or motivation they always gave their best in helping us and we are very happy we could get that help. As it was mentioned before we had a lot of mentor meetings, we talked with them over email, and we tried to be as close to the correct development path as possible. We are very thankful for the aid they gave us, and for the inspiring attitude towards how we should approach our project. Also, we would like to mention that mostly we tried to use the mentor meetings because before we had a lot of insecurities about the first semester where we had some gaps in knowledge but then we got through it, which is amazing in our opinion.



8 Conclusions

In a nutshell, the project we chose was hard, we tried our best to fulfill all the requirements, and we tried to come as close to a perfect execution as possible. However, we didn't implement everything that was expected from us, but we made sure that the other parts are perfect, and we can be proud of the parts that we implemented. Also, the best thing we did is ask for help from our supervisors, which we didn't do last semester, and we are very happy we got the courage, because the result ended up being amazing.