Qdio - Research and Analysis Document

Hugo Cliffordson[1], Alrik Kjellberg[1], Melker Veltman[1], & Oskar Wallgren[1]

Group 25

[1] *TDA 367*

Chalmers University of Technology

2018

Contents

1 Introduction

Ever since the rise of on-demand music streaming services, the world has gained the largest increase in music consumption and discovery in the history of music[1]. To fully understand the effect music has on peoples lives note that, as of 2017, the average American is spending no less than 4.5 hours a day listening to music [2]. With Spotify being the largest actor in the world for paying users [3] we saw an opportunity to further users flexibility with Spotify even more.

The functionality that Spotify is not providing is the availability to let users communicate with the device playing the music without having to manipulate this device physically. Our application Qdio, is the solution to this problem. The app enables a person playing music from Spotify to open a room that other users can connect to. Users connected to the room will then be able to search for music and add it to the music queue on the device playing the music. The use of this application will benefit everyone listening to music using Spotify as their on-demand music streaming service. The prosperity shines in the context of situations. Imagine offices, study groups, families or parties where there is a person playing music through their device or to an external speaker system. Users in these situations will then be able to contribute to the music playing using the application Qdio.

In short Qdio is an extension to the Spotify app with the purpose of enabling users to easily share their music to the queue of the person playing music through their own devices.

2 Definitions, Acronyms and Abbreviations

- Host - a user playing music from its own device and sharing it to others
- Guest - a user subscribing to a room with the possibility of adding music to its queue

3 User Stories

3.1 Epic 1

As a host, I want to create a room that shares a music queue so that others can add songs to my queue from another device

1. Open a Room
   Story ID: 1
   As a host, I want to be able to open a room so that my friends and I can have a common queue list for music.
   Acceptance criteria:

(a) A button in welcome screen with text "create a room" in the center of the screen

(b) A loading animation that shows until room is successfully created.

(c) Pressing it takes user to main interface with search and song view.

2. Pause the Music
   Story ID: 2
   As a host, I want to be able pause the music so that my device gets quiet
   Acceptance criteria:

   (a) Below the song progress bar there is a two striped pause button that stops the music

   (b) When pressing pause the button changes appearance to a play-icon

3. Fast forward Song
   Story ID: 3
   As a host, I want to be able to seek in a song so I skip or return to a part of the song
   Acceptance criteria:

   (a) Below the song information there is a seek bar with a dot representing the duration and position of the song

   (b) Pressing this dot and dragging it somewhere else on the bar changes where in song it's playing

3.2 Epic 2

As a guest, I want to be able to connect to a room so that I can add music to a shared music queue

1. Find Available Rooms
   Story ID: 4
   As a guest, I want to find a list of all available rooms so I can choose which to connect to
   Acceptance criteria:

   (a) A welcome screen shows a button "connect to a room"

   (b) Pressing this button open a view with a list of all available rooms

2. Connect to a Room
   Story ID 5:
   As a guest, I want to be able to connect to a room so that I can contribute to a shared music queue.
   Acceptance criteria:

   (a) A list that shows all available rooms with their names

   (b) Pressing an item in the list connects the user to the room and takes the user to the main interface

3.3 Shared epic

As a user, I want to be able to search for songs,

1. Search for a Song
   Story ID 6
   As a user, I want to be able to search for a song in the Spotify library so that I can find the song I want
   Acceptance criteria:
   (a) A search button in the bottom right corner opens up a keyboard and shows a search field in the top
   (b) As a user is searching a list with songs appear and reloads when adding/removing letters
   (c) Pressing search icon on keyboard hides the keyboard and shows list of search result
2. Add Song to Queue
   Story ID: 7
   As a user, I want to be able to add a song to my shared music queue so that I share my music with others
   Acceptance criteria:
   (a) From a list of songs one should be able to press an item and add it to the queue
   (b) The added song appears in the queue list
3. Search for an Artist
   Story ID: 8
   As a user, I want to be able to search for an artist so I can find a song made by an artist
   Acceptance criteria:
   (a) Typing a name of an artist in search field should show songs from that artist ordered by relevance
4. See the Queue
   Story ID: 9
   As a user, I want to be able to see the current queue list so I can keep track of upcoming songs
   Acceptance criteria:
   (a) The bottom 2/3 of the screen shows a list of all songs in the queue
   (b) A list item should contain song name and artist
   (c) If there are many songs in the queue the user should be able to scroll down the list to see hidden songs in the queue

5. Playback Status
   Story ID: 10
   As a user, I want to see the playback status of a song so that I can get information about the song and its progress
   Acceptance criteria:

   (a) The top 1/3 if the interface shows an album cover picture to the left, next to name of song, artist, and album

   (b) Below the album cover picture and song information, there is a progress bar indicating song duration stretching from left to right

4 User Interface

Since Qdio is all about being easy to use, the user interface is simple and minimalist. The application contains five different scenes in total. Three out of these five will be used if the user wants to open a room in which people can connect to. Four out of these five will be user if the user wants to connect to an existing room.

4.1 Welcome Interface

Upon opening the application, the user will interact with the welcome interface. See figure 1. This scene is built up by two large and distinct buttons. The first of these reads 'Create Room' and reacts the way the user thinks. It will create a room and navigate the user to the host interface. The second button says 'Connect To Room'. Upon clicking, it will navigate the user to the browse interface.

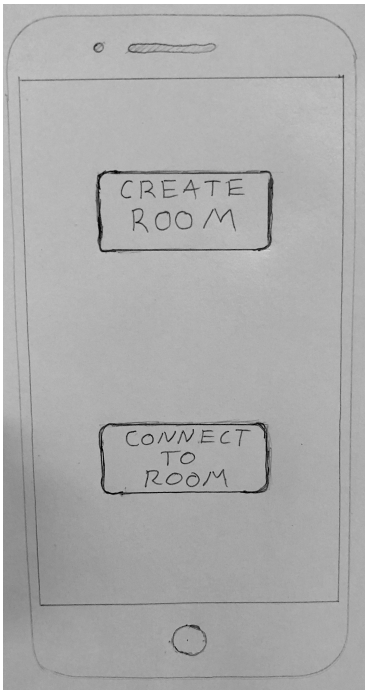

Figure 1: Welcome Screen

4.2 Room Browse Interface

After clicking 'Connect To Room' in the welcome interface the user will be lead this scene. This part of the application only shows a list of names of the available rooms that the user is able to connect to. See figure 2. Upon selecting a room, the guest main interface is shown.

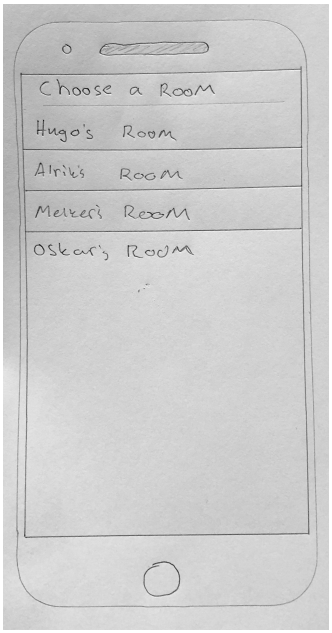

Figure 2: Room Browse Interface

4.3 Guest Main Interface

The guest main interface, see figure 3, is the main scene for users connected to a room. This interface shows the playing status and the active queue of music. In the top of this scene, the user can see the song name, artist, album and an album cover of the song that is currently playing. Beneath the song information, the user will be able to see all the songs in the queue and in which order they will play. The button with the magnifying glass icon will navigate the user to the music search interface.

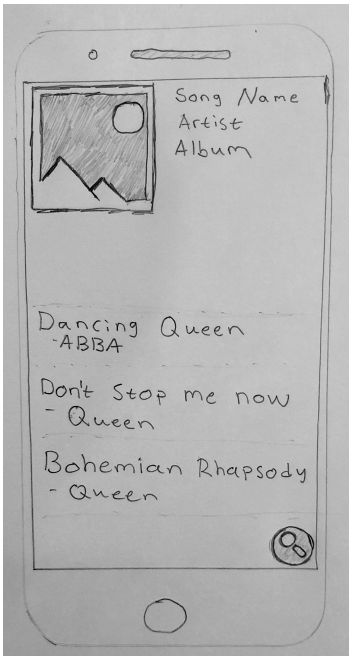

Figure 3: Guest Main Interface

4.4 Host Main Interface

The host main interface, see figure 4, is similar to the guest main interface. The difference between this scene and the guest main interface is that the host interface has controls for playing, pausing and seeking.

Figure 4: Host Main Interface

4.5 Music search Interface

After clicking the search button available in both host and guest main interface, the user will arrive at this scene. See figure 5. The interface contains a search bar and a keyboard in which users can search for songs, artists and albums. It also contains a list of the user's input results. When a song is pressed, it will be added to the queue and the user is brought back to the main interface.

Figure 5: Search Screen

5 Domain Model

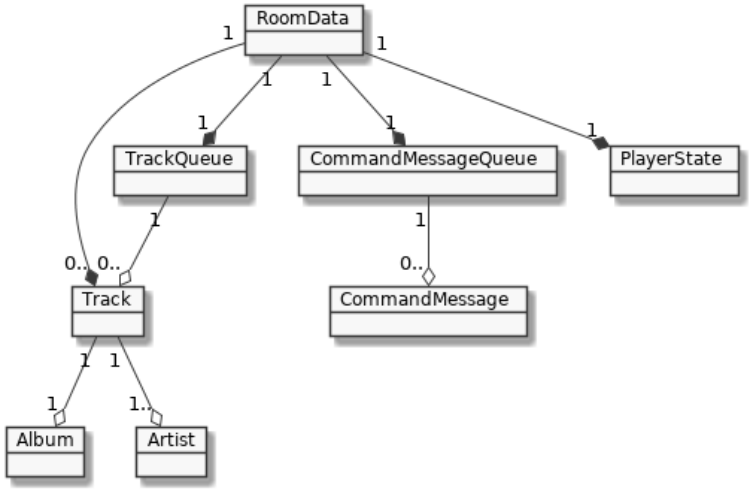

Figure 6: Domain model of Qdio

5.1 Classes

The entry point of the application is the class *RoomData* with the responsibility to delegate functionality to other classes.

The class *PlayerState* reveals what state the player is in, if it is playing, paused or stopped.

CommandMessageQueue works as a queue where all incoming messages from other devices is added and processed from. By keeping this logic separated from the communication logic, it can be separated from the main domain logic.

CommandMessage is the object that contains a message and the action to be executed. Depending on which action is set, the message is interpreted differently.

TrackQueue contains the logic for the current song queue list, which is implemented as a *First-in-First-out* queue. With the logic being separated from a ordinary Java *Queue* it would be an easy implementation to add vote functionality and a priority on songs in the same genre as the rest of the queue.

Track is the model for a song. This contains one or more *Artists* and one *Album*.

References

[1] L. B. Martens. (Mar. 2016). The increase of music consumption and discovery, [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167624516000068.

[2] Nielsen. (Nov. 13, 2017). We listen to music more than 4 1/2 hours a day, [Online]. Available: https://www.marketingcharts.com/industries/media-and-entertainment-81082.

[3] wikipedia. (Oct. 13, 2018). Comparison of on-demand music streaming services, [Online]. Available: https://en.wikipedia.org/wiki/Comparison_of_on-demand_music_streaming_services.