

# Music Manager

Andrew Marshall, Kyle Rolland

## Abstract

**The goal of this project is to create an interface that allows a user to store and manage music on their machine. Users will be able to create play lists, favorite certain songs, and sort songs by the artist, or the song name. Our goal with this project is to make a simple version of music applications like Spotify or iTunes, for people who don't like using their current music player application.**

## 1. Introduction

Music is an everyday part of life, and many people go everyday without even thinking about the luxury that we have in the modern era, being able to listen to what we want, wherever we want to. Following this, there are many many platforms that one can use to access music. We aim to reproduce one such platform, creating an interface that allows someone to add music of their choosing, organize it, and play it.

Providing people with more options is never a bad thing. Even if there are only a few things that differentiate one software from another, there are always going to be users looking for new experiences. We would like to be able to appeal to as wide an audience as possible, people who have problems with the existing flaws in other software, which are ignored or put off of being fixed by developers. Helping those people find a new place where they feel they can store their favorite songs can make a large difference, especially because music can be a very valuable, memorable resource for some people.

### 1.1. Background

Over the course of this proposal, we will use the term Metadata, and in this case, it's really just a technical term for basic music information. It pertains to things like album art, release date, track/song length, genre, artist, and the songs title.

We decided on a project like this because music is something that is easy to make connections between. Thankfully, The two of us had common ground in a genre that we used to listen to, but even if we didn't, it would be easy to come together to work on a project like this. It originally sprouted as an idea for an audio waveform visualizer, the flashing bars/lights that you'll see with music videos, that correspond to the beat of the song, or the lyrics. Neither of us have looked into what it takes to make those, and some further consideration led us to decide that it would be better to take a step back and just make a music management interface.

Andrew has run into problems with CDs on iTunes, so we hope to be able to implement a way transfer music from the manager to the CD, allowing the manager to effectively use and read CDs.

Kyle's main music application is Apple Music , and he has some issues with listening queue management, for example, shuffling a play list, and then adding it to the end of the list of what is currently being listened to, which is not possible in the application. If we could work the capability into this project, it may become an alternative listening application for him as well.

### 1.2. Impacts

There are many music managing systems on the internet, and we would like to see if we can create something worth telling others about, so that they can use as well. We want to add some functionality that gives us a chance to stand out, making it easier to reach even just a few people, and give them another place to handle the songs that they hold dear. It doesn't take much to brighten someone's day, and hopefully with this project we can manage to do that for a handful of people.

### 1.3. Challenges

It may be difficult to organize and handle the previously mentioned metadata, since there is a lot of information that goes along with it. We also may run into trouble trying to create a system to accept different file types, like .mp3 versus .mp4 versus .wav, but working a system like this out would prove very useful for attracting more users. Making a display for things like album art, and giving the user the option to upload their own art to go with songs or created play lists may also prove to be difficult, partially because of accepting images, and partially reaching back to accepting different image file types, like .jpeg and .png. Lastly, we are still sort of hoping to implement an audio waveform visualizer somewhere within the interface, and we think this will prove to be the biggest hurdle, especially if we can't find a framework to start with.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Add song to queue	Listener	Med	1
2	Add new song to manager	Listener	Hard	1
3	Add album art	Listener	Med	2
4	Play from CD	Listener	Hard	3
5	Change volume	Listener	Med	1

TABLE 1. USE CASE INDEX

## 2. Scope

Our project will be completed when we have an interface that allows the user to add songs of their choosing, modify their metadata, add those songs to play lists and edit those play lists freely and easily, search for songs within their database, and sort them by given data types, like artist, genre, and song title

We have some ideas for stretch goals. One of them would be implementing the previously mentioned audio waveform visualizer. Along with this, being able to easily export your play lists or songs to various different platforms would be very handy. Lastly, we would love to be able to implement a system that allows the user to upload images for artwork that will display when a song is playing, be it album art, or an image that will act as the cover of a given play list, allowing for more user customization.

### 2.1. Requirements

#### 2.1.1. Functional.

- Store songs that are currently being played, and are queued to play
- Keep track of previously played songs for accessibility later
- Display current song being played, as well as associated artwork
- Hold songs that are being played for storage across listening sessions
- Give access to music metadata so user can change it
- Allow for customization of user created play lists
- Prevent duplication of music in library
- Adhere to copyright laws, recognizing proper credit to musicians

#### 2.1.2. Non-Functional.

- When user updates music information, update within 5 seconds
- Ensure that music and image files are of proper formatting
- Ensure that songs and data are easily accessible and understandable to the user, through effective use of application interface

### 2.2. Use Cases

Use Case index can be seen in Table 1.

Use Case Number: 1

Use Case Name: Add song to queue

Description: User selects a song that they wish to play next. They will click on a plus button that appears when hovering over the song. This will add the song to the list, and will be played next.

- 1) User selects song that they want to play next
- 2) After selecting song, plus button appears on song bar
- 3) User presses button, adds song to queue

Termination Outcome: Selected song will be played next

(see Figure 1)

Use Case Number: 2

Use Case Name: Add music to manager

Description: Menu bar at top of screen will have a file button. User presses button, and asks user what they would like to add. User selects song, asks for file name of song that will be added. If file can be found, and is an accepted file type, it will be added to the manager.

- 1) User presses File button on menu bar
- 2) Drop down menu appears, allows user to click on Add Song...

- 3) User enters song file that they want to add
- 4) If file is appropriate formatting, song is added to manager

Termination Outcome: New song is present in management library

Alternatively:

- 1) User presses File button on menu bar
- 2) Drop down menu appears, allows user to click Add Song...
- 3) User enters song file that they want to add
- 4) Song is already in library
- 5) Song is prevented from being added
- 6) (OPTIONAL: Sends user to song in library once prevention occurs)

Termination Outcome: Duplication of song(s) prevented

(see Figure 1)

Use Case Number: 3

Use Case Name: Add album art

Description: When looking at an album or play list, displays a button that will allow the user to select an image to upload. Once selected, image will be used as art for the given album/play list.

- 1) User opens album or play list
- 2) User presses button at top corner of album art area
- 3) Menu appears that allows user to modify item
- 4) User selects Add Image...
- 5) User enters image name that they would like
- 6) If file is appropriate formatting, art is replaced

Termination Outcome: Previous album art is replaced with new art

(see Figure 3)

Use Case Number: 4

Use Case Name: Play from CD

Description: Menu bar at top of screen will have a device button, clicking on it allows user to select an outside device to play music from, or send music to (This is just a goal for right now, something to work towards).

- 1) User presses Device button on menu bar
- 2) Drop down menu appears
- 3) User selects Import from CD
- 4) Music on CD is loaded into manager

Termination Outcome: Music present on CD is now loaded into user's library

(See Figure 4)

Use Case Number: 5

Use Case Name: Change volume

Description: Volume buttons displayed beneath play section (Possibly replace with volume slider/better display). User presses + button to increment volume up by one, and - button to decrement volume by one, to an upper and lower limit respectfully.

- 1) User presses plus button next to speaker image

Termination Outcome: Volume increases by one increment

Alternatively:

- 1) User presses minus button next to speaker image

Termination Outcome: Volume decreases by one increment

(See Figure 5)

You will then need to continue to flesh out all use cases you have identified for your project.

## 2.3. Interface Mockups

## 3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

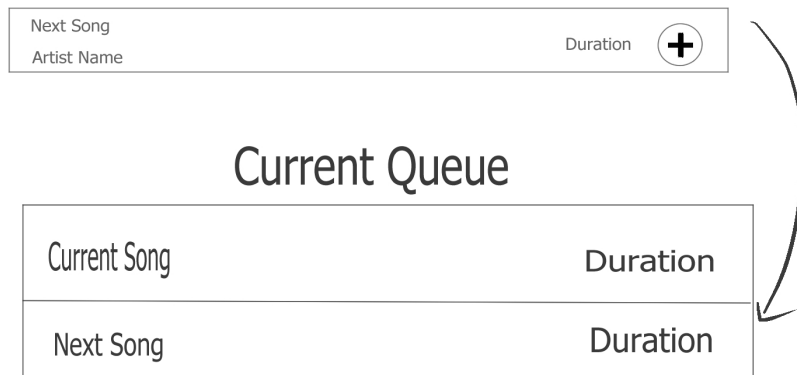


Figure 1. Use Case 1, Adding song to queue

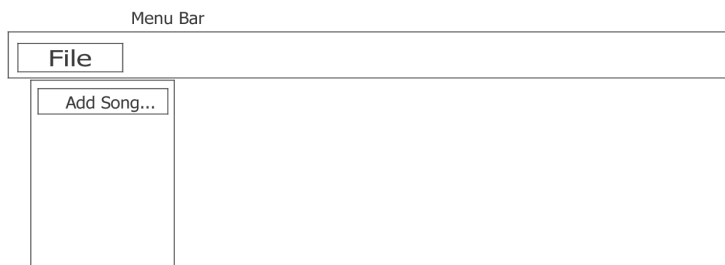


Figure 2. Use Case 2, Adding music to manager

## 4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).



Figure 3. Use Case 3, Adding album art

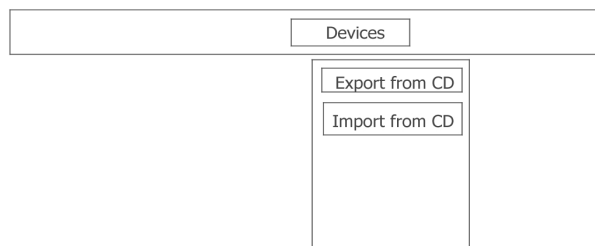


Figure 4. Use Case 4, Interacting with CD



Figure 5. Use Case 5, Volume button

#### 4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a `figure` environment, and to reference with the `ref` command. For example, see Figure 6.

#### 4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

### 5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

#### 5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

### References

- [1] H. Kopka and P. W. Daly, *A Guide to L<sup>A</sup>T<sub>E</sub>X*, 3rd ed. Harlow, England: Addison-Wesley, 1999.



Figure 6. Your figures should be in the *figure* environment, and have captions. Should also be of diagrams pertaining to your project, not random internet kittens