Butterfly

Project Document

Team Number: 3

Team Leader: Jake Wheeler

Team Members

Nate Christiansen

Nick Kapty

# Contents

## Project Drivers

1. The Purpose of the Project

2. The Client, the Customer, and Other Stakeholders

3. Users of the Product

## Project Constraints

4. Mandated Constraints

5. Naming Conventions and Definitions

6. Relevant Facts and Assumptions

## Functional Requirements

7. The Scope of the Work

8. The Scope of the Product

9. Functional and Data Requirements

## Nonfunctional Requirements

10. Look and Feel Requirements

11. Usability and Humanity Requirements

12. Performance Requirements

13. Operational and Environmental Requirements

14. Maintainability and Support Requirements

15. Security Requirements

16. Cultural and Political Requirements

17. Legal Requirements

## Project Issues

18. Open Issues

19. Off-the-Shelf Solutions

20. New Problems

21. Tasks

22. Migration to the New Product

23. Risks

24. Costs

25. User Documentation and Training

26. Waiting Room

27. Ideas for Solutions

# 1. The Purpose of the Project

## 1a. The User Business or Background of the Project Effort

### The purpose of the project is to create a music player application to play local music. Users will be able to sort by album, artist, song title, genre, as well as various other tags. Users will be able to log into social media to share what they are currently listening to.

## 1b. Goals of the Project

### To create a quality music player that will allow users to listen to music, with customizable features and social media integration.

# 2. The Client, the Customer, and Other Stakeholders

## 2a. The Client

### We are our own client, so we are using our own specifications.

## 2b. The Customer

### The customer of our product would be anyone who wishes to use our music player.

## 2c. Other Stakeholders

Our only stakeholder is Professor Fan, who is teaching us as we develop the project.

# 3. Users of the Product

## 3a. The Hands-On Users of the Product

### Any user who has the knowledge to use a computer and any passing interest in music can use our product.

## 3b. Priorities Assigned to Users

## ● Key users: The users with Twitter and the ability to dig into the customization of our player. They will use all of the additional features of the product, such as playlists, social media features, etc.

## ● Secondary users: Users who simply use the player as a music player and nothing more. They will use the product for its local music playing ability only.

## 3c. User Participation

### We will act as users for this project, testing all of the features added. We will also consider outside input from unbiased third parties.

## 3d. Maintenance Users and Service Technicians

### We will maintain and service our product after its initial release and all additional releases.

# 4. Mandated Constraints

## 4a. Solution Constraints

### We will be using Java, because Java has a rich framework, and well developed and documented APIs. Java also runs on any operating system, as it is not platform dependent. The program will also meet all of our functional and non-functional requirements.

## 4c. Partner or Collaborative Applications

### Our application will feature Twitter integration, and as such will use Twitter features and services.

## 4d. Off-the-Shelf Software

### We will be using Twitter4J, an unofficial Java library for the Twitter API.

## 4e. Anticipated Workplace Environment

### The environment targeted will be Windows machines with Java Runtime Environment installed, as well as internet connectivity.

## 4f. Schedule Constraints

### The project must be completed by December 16, 2015, in order to receive full credit for the class.

# 5. Naming Conventions and Definitions

## 5a. Definitions of All Terms, Including Acronyms, Used in the Project

### Butterfly – The name of our music player application

### Song – Individual music file that can be played

### Playlist – User selected list of songs

### Artist – Creator of a song

### Album – Collection of specific songs chosen by the artist

### Tweet – A short message that is sent to Twitter that is written by the user.

# 6. Relevant Facts and Assumptions

## 6a. Facts

### Our project will be developed in Java, and will be completed in one semester. The user will need a Twitter account to take advantage of all of the features that our application will offer.

## 6b. Assumptions

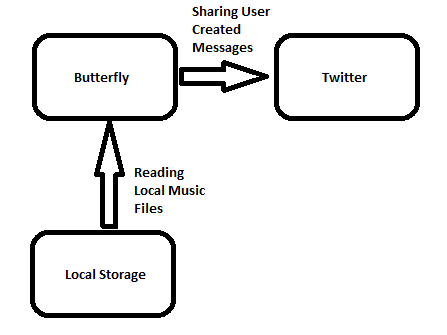
### We assume that we will be able to develop the program using NetBeans and appropriate frameworks. We also assume Twitter integration will be available.

# 7. The Scope of the Work

## 7a. The Current Situation

### Currently there are no popular music players that allow users to engage with social media.

## 7b. The Context of the Work



# 8. The Scope of the Product

## 8a. Product Boundary



# 9. Functional and Data Requirements

## 9a. Functional Requirements

|  |  |
| --- | --- |
| Use Case #1 | Play Music |
| Goal in Context | To allow the user to play music |
| Scope | The Butterfly system |
| Level | Summary |
| Primary Actor | User |
| Preconditions | Song is highlighted/selected |
| Minimal Guarantee | Nothing happens |
| Success Guarantee | The selected song begins playing |
| Trigger | Play button or double click on the song |
| Description Step |  |
| 1 | The user selects a song to play |
| 2 | The system finds the song file |
| 3 | The system begins audio output |



|  |  |
| --- | --- |
| Use Case #2 | Search Music |
| Goal in Context | To allow the user to search for a song |
| Scope | The Butterfly system |
| Level | Summary |
| Primary Actor | User |
| Preconditions | System is open and running |
| Minimal Guarantee | Tells user that song cannot be found |
| Success Guarantee | Songs matching the keywords appear to the user |
| Trigger | Typing in the search box |
| Description Step |  |
| 1 | The user types keywords into the search field |
| 2 | The system searches the keywords in the list of songs |
| 3 | The system filters matching songs and displays them to the user |



|  |  |
| --- | --- |
| Use Case #3 | Create Playlist |
| Goal in Context | To allow the user to create custom playlists |
| Scope | The Butterfly system |
| Level | Summary |
| Primary Actor | User |
| Preconditions | The user has at least one song |
| Minimal Guarantee | Nothing happens |
| Success Guarantee | The song is added to the playlist |
| Trigger | Right click, add to playlist |
| Description Step |  |
| 1 | The user right-clicks the song |
| 2 | The user selects “add to playlist” option |
| 3 | The system prompts the user to create playlist or choose playlist |
| 4 | The user selects a playlist |
| 5 | The system adds the song to the playlist |
| Extension Step |  |
| 4a | The use selects create playlist |
|  | A1: The system adds a new playlist to the list of playlists |
|  | A2: The user then selects a playlist to add the song to |



|  |  |
| --- | --- |
| Use Case #4 | Tweet Song |
| Goal in Context | To allow the user to tweet about currently chosen song |
| Scope | The Butterfly system |
| Level | Summary |
| Primary Actor | User |
| Preconditions | Song is playing, user has twitter account |
| Minimal Guarantee | Nothing happens |
| Success Guarantee | The tweet is posted by the user |
| Trigger | User selects to tweet about current song |
| Description Step |  |
| 1 | System displays window that allows user to edit tweet template |
| 2 | The user submits the tweet |
| 3 | The system posts the tweet to the user’s timeline |
| Extension Step |  |
| 1a | The user is not logged into Twitter |
|  | A1: The Login subfunction is called |



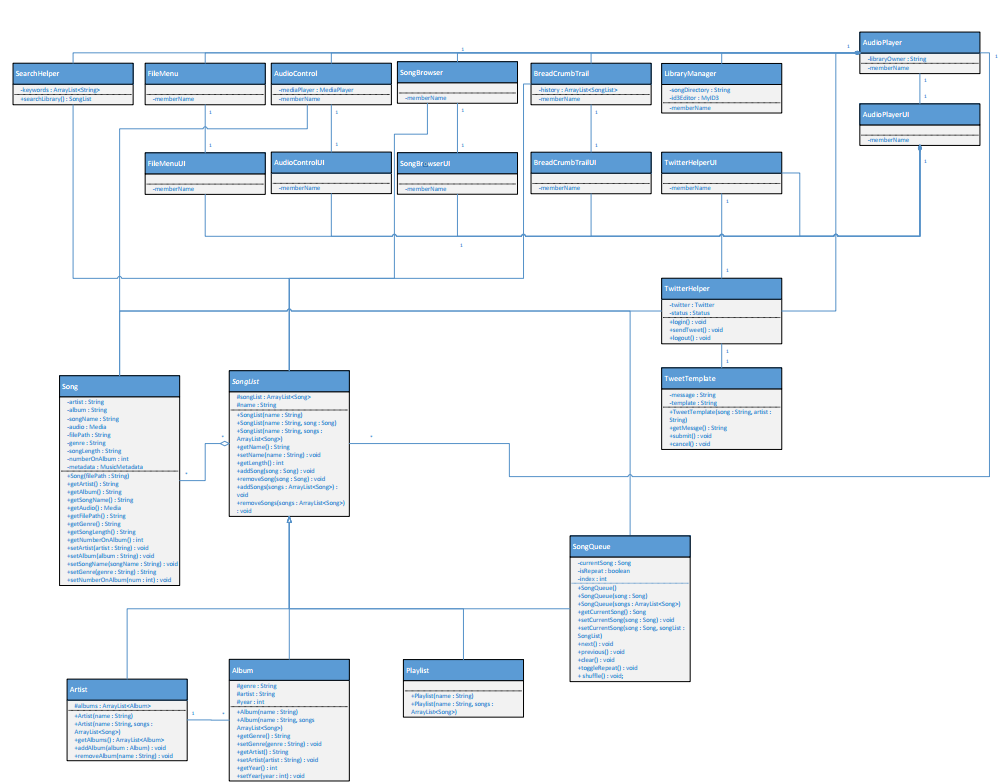
|  |  |
| --- | --- |
| Use Case #8 | Login |
| Goal in Context | To allow the user to login to Twitter |
| Scope | The Butterfly system |
| Level | Sub-function |
| Primary Actor | User |
| Preconditions | System is open and running, internet connection available |
| Minimal Guarantee | Nothing happens |
| Success Guarantee | User is logged into Twitter |
| Trigger | User attempts to login or access Twitter functionality |
| Description Step |  |
| 1 | The user enters their account username and password |
| 2 | The system logs the user into Twitter |

### 

# 10. Project Plan

### Untitled.png

# 11. Class Diagram 1 (Domain Concepts)



# 12. Class Diagram 2 (System Domain Model)



# 13. State Diagram

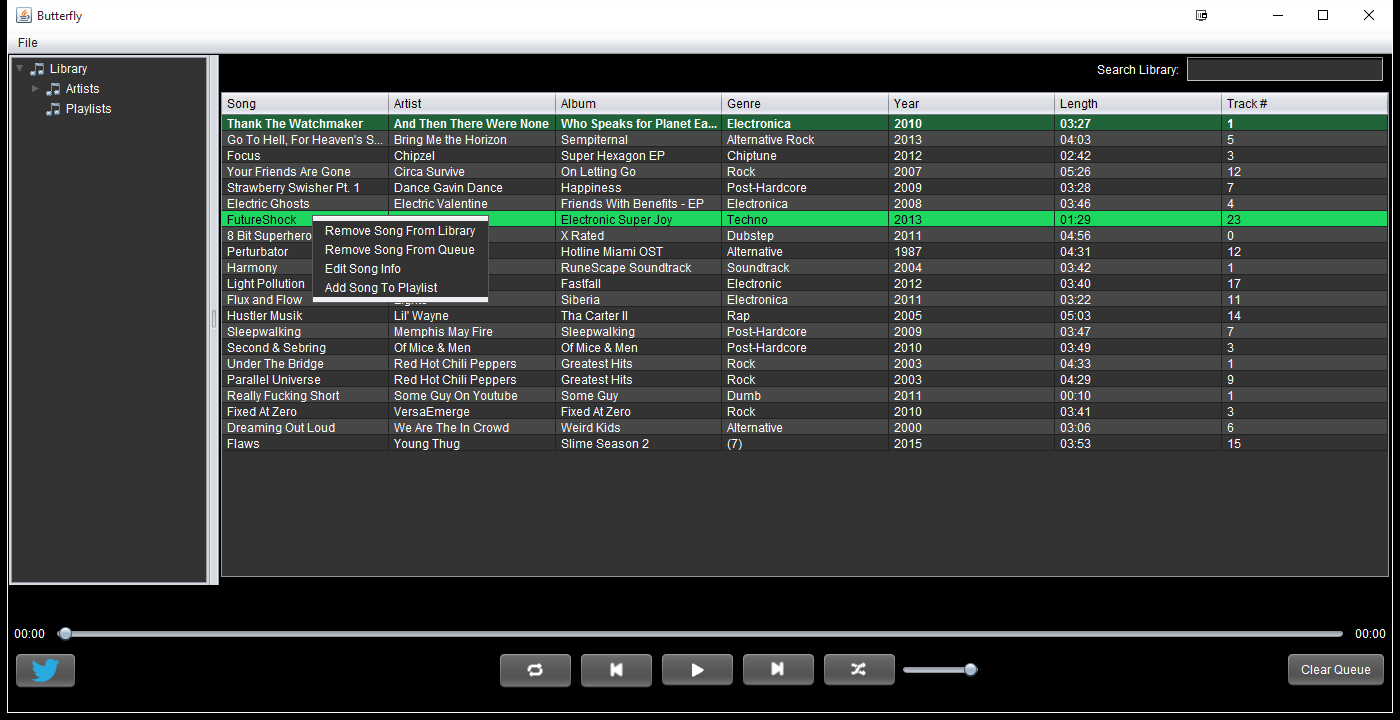
We did not create a state diagram for this project.

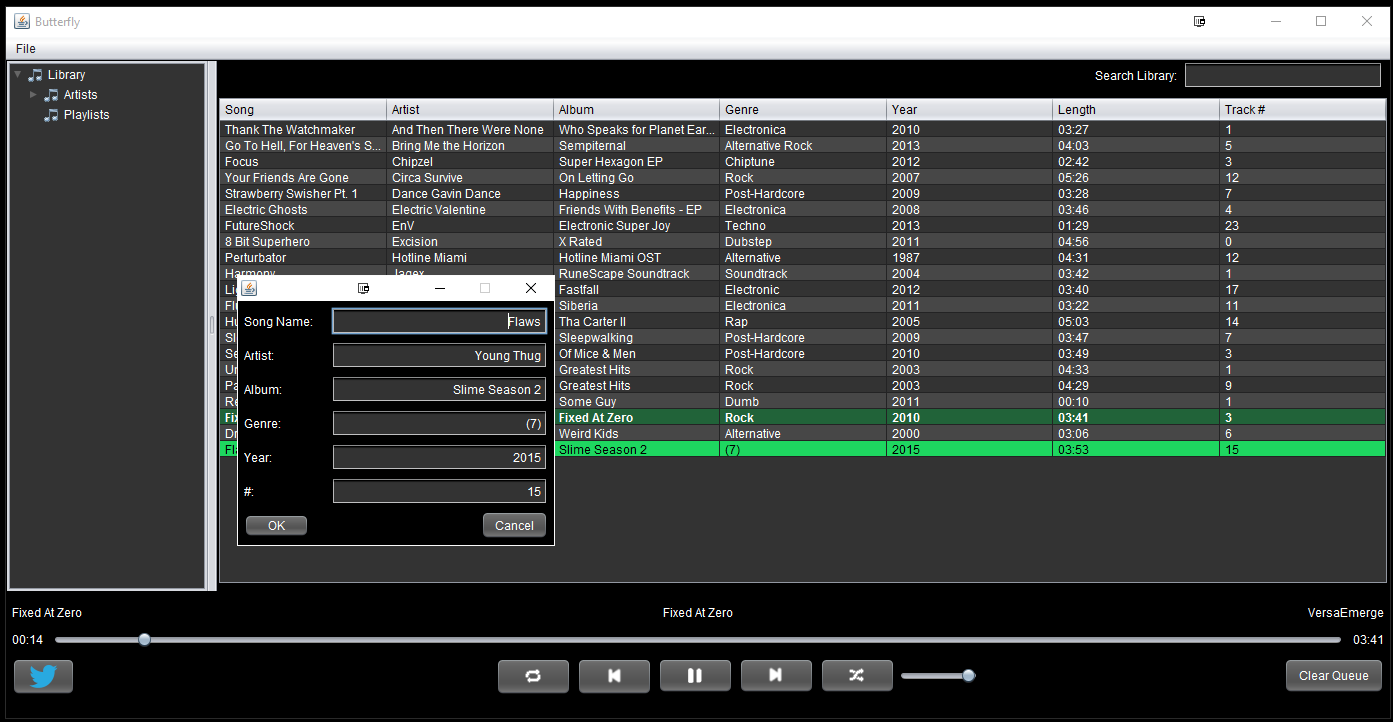
# 14. Class Diagram 3 (Final System Model)

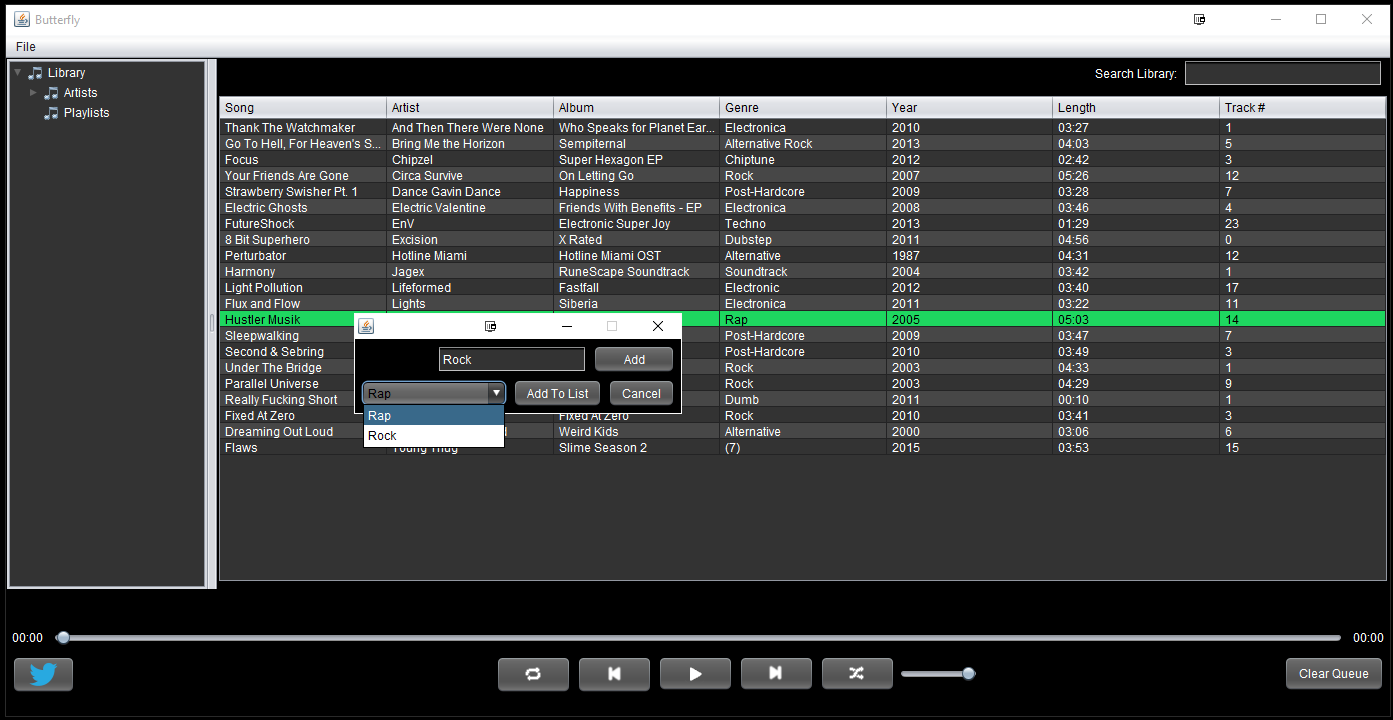
# 15. Project Deployment

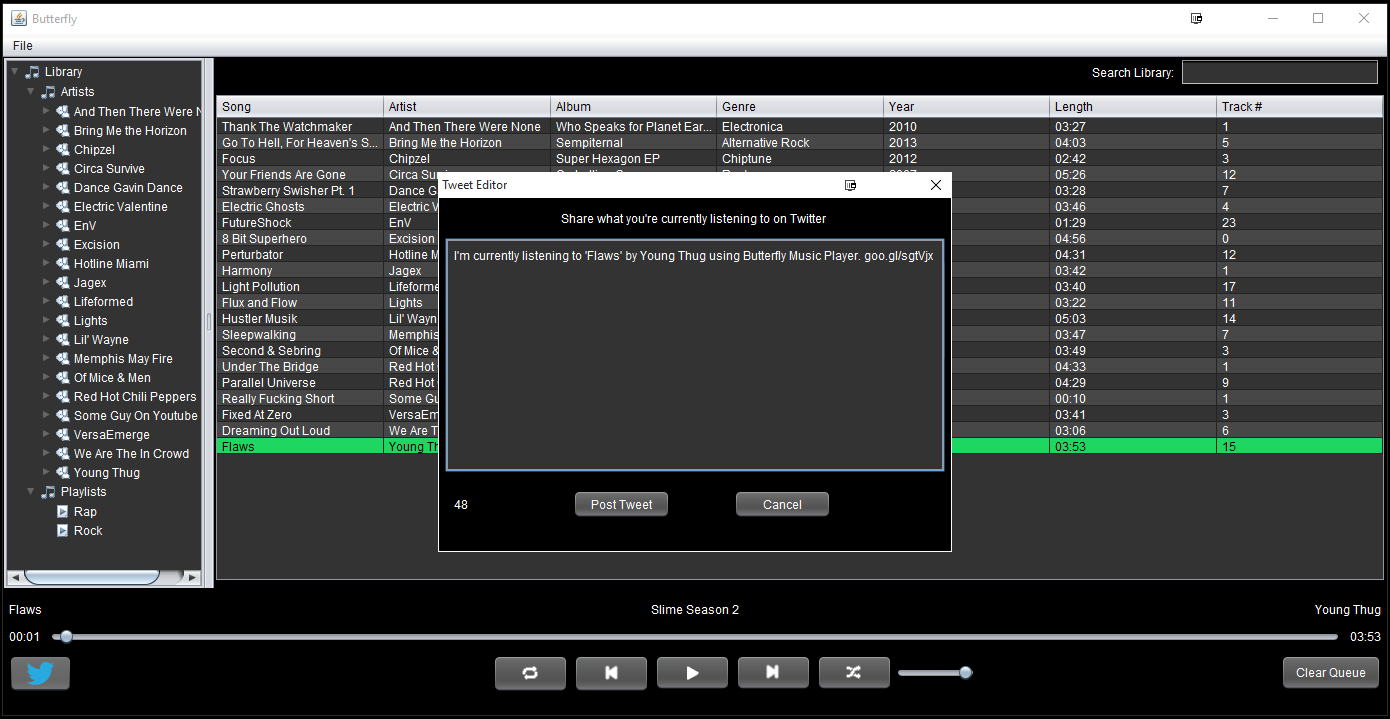
<https://github.com/jakewheeler/SWENG411-Butterfly>

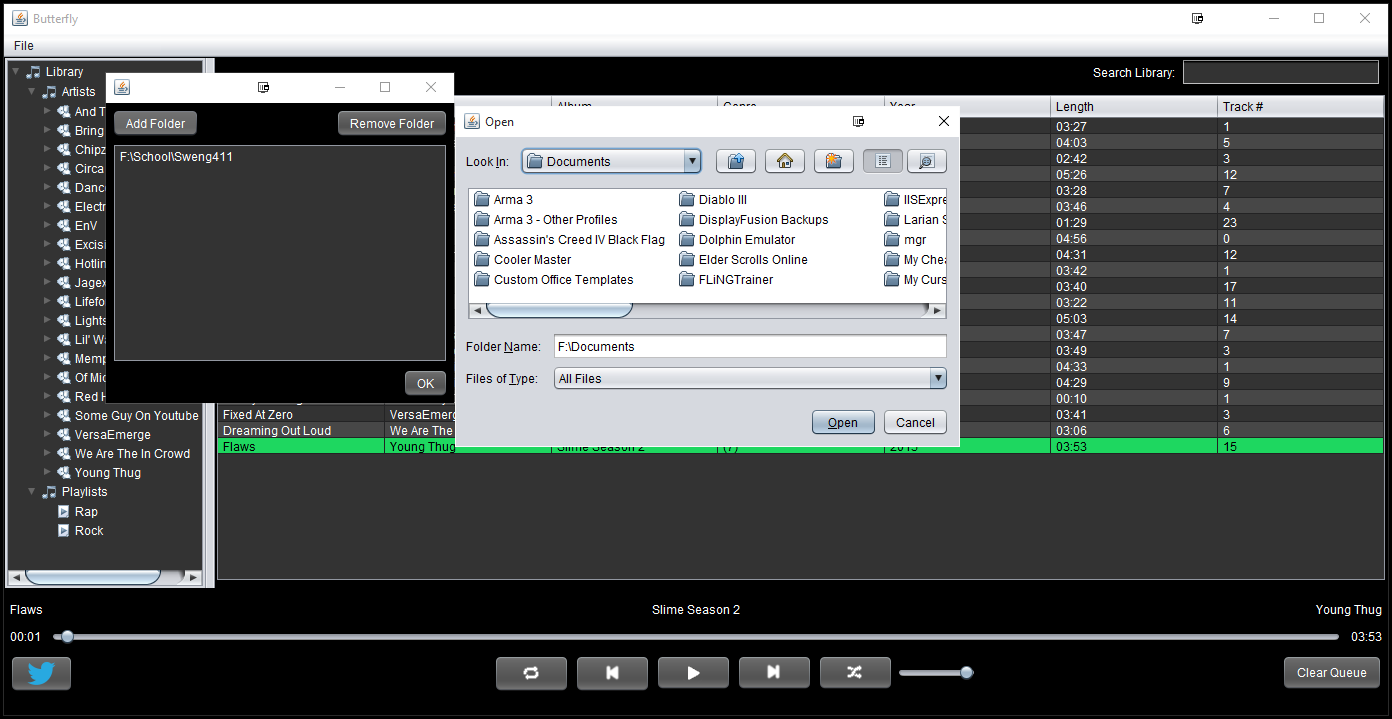
# 16. System Screenshots

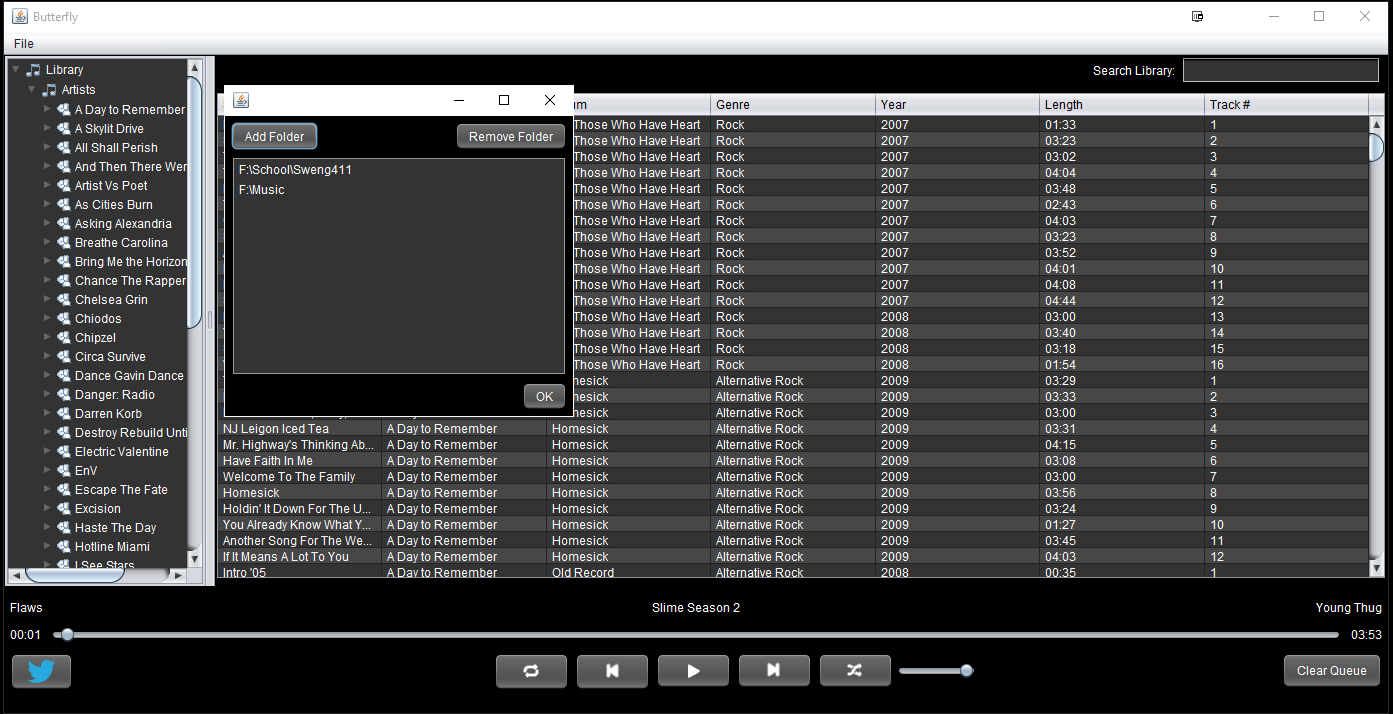












# 17. Project Source Codes

Attached separately

# 18. Project Presentation Slides

Attached separately

# 19. Project Video Clips

Attached seperately

# 20. Open Issues

Given enough time, we would have liked to implement a couple of additional features. We would have liked to give the user the ability to customize the appearance of the application. We also would have liked to save the user’s settings to a file and load them on startup. We would have liked to update the song time while dragging the bar to skip through the song.

# 21. Software Process Model

Our team used the agile method of development while working on Butterfly. We used this methodology so that we can continually increment on our design and add additional features as we develop and test.

# 22. Software Architecture Used

We used the MVC design architecture for Butterfly. Our model is the Song and SongList classes, the view is all of the UI elements, and the controllers are the systems that Butterfly uses.

# 23. Design Patterns

We used general hierarchy for the SongList portion of the project. We used the façade pattern for Butterfly’s systems that interact. We used the singleton pattern for the RightClickMenu.

# 24. Key Design Decisions

We initially were going to have YouTube functionality with our music player, however we discovered that it is against YouTube’s terms of service to pull audio from their videos.

# 25. Team Communications

Our team met once a week after labs on Fridays, and we also occasionally met and discussed the project throughout the week when necessary. In order to keep track of source code, we used Git version control.

# 26. Task Allocation and Responsibilities

We kept a to-do list in a folder on the repository, and communicated the tasks that each of us were working on so that no overlap occurred.

# 27. Lessons learned by Each Team Member

Nate Christiansen –

Jake Wheeler –

Nick Kapty –