|  |  |
| --- | --- |
| INTUNE | Abstract  This application will be a single music player that integrates multiple music streaming services that allows the user to listen to music throughout each service and access playlists with the ability to download certain songs and sync them to a local playlist.  Gregory Montilla  CS 003B |

Table of Contents

[1.0 Problem Statement 2](#_Toc500445507)

[2.0 Bottom up Design 5](#_Toc500445508)

[Nouns: 5](#_Toc500445509)

[Verbs: 1](#_Toc500445510)

[Combined nouns and verbs: 2](#_Toc500445511)

[Classes: 3](#_Toc500445512)

[3.0 Top down design 4](#_Toc500445513)

[CRC Cards: 5](#_Toc500445514)

[Classes based on CRC cards: 8](#_Toc500445515)

[4.0 Merged list 9](#_Toc500445516)

[5.0 Second look at list of classes 10](#_Toc500445517)

[Table of domain and range: 11](#_Toc500445518)

[6.0 UML diagram 13](#_Toc500445519)

[7.0 JavaDoc 13](#_Toc500445520)

[8.0 Additional Topic 14](#_Toc500445521)

[9.0 Test plan 15](#_Toc500445522)

[10.0 Installation Guide 16](#_Toc500445523)

[11.0 User Guide 16](#_Toc500445524)

[12.0 Project Analysis 17](#_Toc500445525)

# 1.0 Problem Statement

The application will be a music player that allows a user to access playlists from music streaming services such as Spotify, Soundcloud and YouTube.

In order for the user to maximize the potential of the application they may connect their account and add their location. The user’s location will be used to find local events. The user may connect to their accounts using their credentials. This application will use OAuth 2.0 to allow the user to connect to the various services. The application will need a “Client ID” and a “Callback URL”. Spotify also needs a “Client Secret”. IF the user decides not to connect their account an “API key” is still needed to obtain data from the music services. In order for Spotify to properly stream their music, a user will need to have a premium account.

If connected to their personal accounts, the user will be able to see playlists that they have saved online in the application. These playlists should be able to be edited by adding a song or removing a song.

The main activity of the application will be a media controller that allows the user to navigate through their media.

A newsfeed will present to the user newly released songs from artists and events within the user’s area. These artists will be artists that are found in the user’s playlists from the music streaming services. If a user clicks a card in the newsfeed, it will redirect the user to either three things: artist, event, or song. By default, it will either redirect the user to the song or playlist, depending on which it is. In order to be redirected to the artist, they will have to specially click the artist’s name. A user directed to a to an event will have their default web application opened to a webpage that allows the user to buy a ticket to the event if they wish. If the user is clicking a song, it will automatically start streaming in the background. If the user chooses the artist, they will be directed to the artist’s page that shows more information about the artist.

The music player will control the playback of the music and control the volume. It will allow the user to skip, go to the previous, play, pause, repeat and shuffle. A user will be able to scrub to a certain position in the song.

When searching for a song, the top result will be the one that most resembles what the user searched for. It will still show results from the other services and will give them an option to stream, add to a local or streaming playlist, and download.

The user will be able to see playlists from local and streaming services. They will be able to edit the playlist by adding or removing a song. They can edit the name and create new playlists. These newly created playlists may either be local or specified to a specific music streaming service.

Songs will show songs that are in the playlists of the user and local songs on the device. Clicking the song will allow the song to start playing, while a long hold on a local song will allow the user to edit the meta date of the song. The meta data includes the title, artist, album and album art.

Artists will be determined by the artists that are found by parsing through local and online playlists. A user may click on that artist and bring up information pertaining to the specific artist.

User will be allowed to enter their zip code in order to see local events from artists that are coming up. The user will also have personal statistics pertaining to their music listening habits. These personal statistics will include most played songs, most played artists and most played albums.

The user will be able to search through their local songs, local playlists, online playlists and multiple streaming services for a song. Songs found from Soundcloud and YouTube will be able to be downloaded. All songs will be able to be added to a playlist that is specific to the music streaming service it was found from.

# 2.0 Bottom up Design

## Nouns:

1. Account
2. Activity
3. Album
4. Album art
5. API key
6. Application
7. Area
8. Artist
9. Artist page
10. Artists
11. Callback URL
12. Card
13. Client id
14. Client secret
15. Credentials
16. Data
17. Event
18. Information
19. Local playlist
20. Local song
21. Location
22. Media controller
23. Meta data
24. Most played albums
25. Most played artists
26. Most played songs
27. Music player
28. Music streaming service
29. Name
30. Newly released songs
31. Newsfeed
32. OAuth
33. Online playlists
34. Personal account
35. Personal statistics
36. Playlist
37. Premium account
38. Result
39. Song
40. Soundcloud
41. Spotify
42. Streaming playlist
43. Ticket
44. Title
45. User
46. Web application
47. Webpage
48. YouTube
49. Zip code

## Verbs:

1. Access
2. Add
3. Allows
4. Buy
5. Click
6. Connect
7. Control
8. Create
9. Download
10. Edit
11. Find
12. Give
13. Maximize
14. Need
15. Parsing
16. Pause
17. Play
18. Present
19. Previous
20. Redirect
21. Removing
22. Repeat
23. Resembles
24. Scrub
25. Searching
26. See
27. Show
28. Shuffle
29. Skip
30. Stream
31. Using

## Combined nouns and verbs:

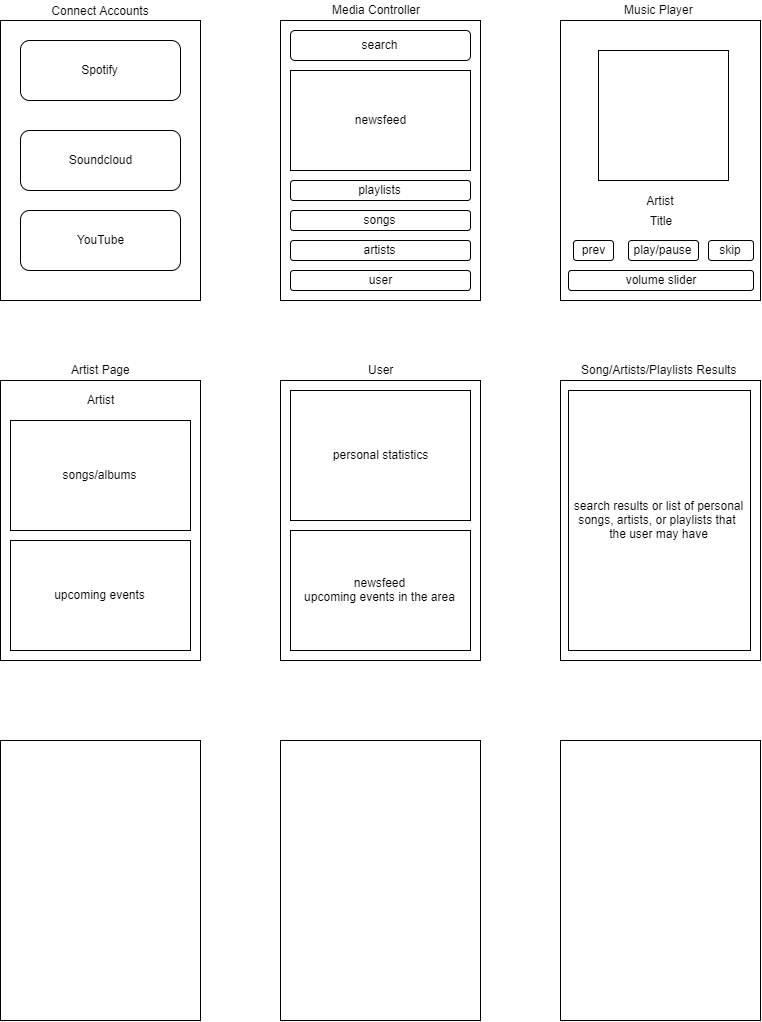
|  |  |
| --- | --- |
| Activity  Application  Media Controller Web application  Web page | Search Find Redirect |
| Album  Album art  Artist  Song  Name  Title | Show |
| Artist  Albums  Information Event  Newly released songs | Show |
| Event  Artist  Area Location  Ticket  URL | Buy |
| Song  Title  Artist  Album  Album art  Meta data | Edit Download Play Pause Show |
| Streaming services  Soundcloud  Spotify  YouTube  Account  API Key  Callback URL  Client ID  Client Secret  Credentials  OAuth  Premium account | Access Connect Parse Search Download |

|  |  |
| --- | --- |
| User  Personal statistics  Area  Newsfeed  Information  Events  Most played songs  Most played artists  Most played albums | Edit |
| Music Player  Song  Playlist  Artist | Play Pause Skip Repeat Shuffle Stream Previous Scrub |
| Playlist  Songs  Title | Create Edit Add Remove |

## Classes:

1. Album
2. Application
3. Artists
4. Event
5. Music Player
6. Newsfeed
7. Playlist
8. Songs
9. Streaming Service
10. User

# 3.0 Top down design



## CRC Cards:

|  |  |
| --- | --- |
| Media Controller | |
| Responsibilities:  Navigate through media  Search | Collaborations:  Spotify  Soundcloud  YouTube  Playlists  Artists  Songs  Newsfeed |

|  |  |
| --- | --- |
| Spotify | |
| Responsibilities:  View songs  View playlists  Lower volume | Collaborations:  Media Controller  Music Player  Songs |

|  |  |
| --- | --- |
| Soundcloud | |
| Responsibilities:  View songs  View playlists  Soundcloud title  Soundcloud URL  Lower volume | Collaborations:  Media Controller  Music Player  Songs |

|  |  |
| --- | --- |
| YouTube | |
| Responsibilities:  View playlists  YouTube title  YouTube URLLower volume | Collaborations:  Media Controller  Music Player  Songs |







|  |  |
| --- | --- |
| Music Player | |
| Responsibilities:  Play  Pause  Repeat  Shuffle  Skip  Previous  Adjust Volume | Collaborations:  Media Controller Songs  Playlists  Artists  Spotify  Soundcloud  YouTube |

|  |  |
| --- | --- |
| Playlists | |
| Responsibilities:  Name  Song  Add song  Remove song  Create playlists  View playlists  Delete playlist | Collaborations:  Media Controller  Spotify  Soundcloud  YouTube Songs |

|  |  |
| --- | --- |
| Artists | |
| Responsibilities:  Event  Song  Album  View events  View songs  View albums | Collaborations:  Media Controller  Newsfeed  Songs  Albums  Event |

|  |  |
| --- | --- |
| Songs | |
| Responsibilities:  Title  Artist  Album  Album art  Edit title  Edit artist  Edit album  Edit album art | Collaborations:  Media Controller  Streaming Services  Playlist |

|  |  |
| --- | --- |
| User | |
| Responsibilities:  View events  View personal statistics | Collaborations:  Spotify Soundcloud  YouTube  Songs  Artists  Newsfeed  Playlist |

|  |  |
| --- | --- |
| Newsfeed | |
| Responsibilities:  Show events  Show new songs | Collaborations:  User  Media Controller Artist |

|  |  |
| --- | --- |
| Events | |
| Responsibilities:  Show artist  Show date  Show time Start  Show time End  Show Venue | Collaborations:  User  Newsfeeds Artist |

## Classes based on CRC cards:

1. Artists
2. Events
3. Media Controller
4. Music Player
5. Newsfeed
6. Playlists
7. Songs
8. Soundcloud
9. Spotify
10. User
11. YouTube

# 4.0 Merged list

1. Album
2. Application/Media Controller
3. Artists
4. Event
5. Music Player
6. Newsfeed
7. Playlist
8. Song
9. Soundcloud
10. Spotify
11. Streaming Service
12. User
13. YouTube































# 5.0 Second look at list of classes

|  |
| --- |
| Artists |
| Events |
| Media Controller |
| Music Player |
| Newsfeed |
| Playlists |
| Songs |
| Streaming Services  Soundcloud  Spotify  YouTube |
| User |

## Table of domain and range:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Data Type | Size | Domain/Range | What if? |
| Keyword | String | 25 | All characters | Will not be able to search if left blank |
| Playlist Title | String | 25 | All characters | Playlist name cannot be left blank, prompt |
| Song album | String | 30 | All characters | Song album cannot be left blank, prompt |
| Song album art | Image | Null | Image | If no album art is specified, song will use default album art |
| Song artist | String | 30 | All characters | Song artist cannot be left blank, prompt |
| Song title | String | 30 | All characters | Song title cannot be left blank, prompt |
| Soundcloud client ID | String | 32 | A-z char and int | Application will not be able to communicate with Soundcloud |
| Soundcloud client secret | String | 32 | A-z char and int | Application will not be able to be communicate with Soundcloud |
| Spotify client ID | String | 32 | A-z char and int | Application will not be able to be communicate with Spotify |
| Spotify client secret | String | 32 | A-z char and int | Application will not be able to be communicate with Spotify |
| YouTube API key | String | 39 | A-z char and int | Application will not be able to communicate with YouTube |
| YouTube client ID | String | 72 | A-z char and int | Application will not be able to communicate with YouTube |
| YouTube client secret | String | 32 | A-z char and int | Application will not be able to communicate with YouTube |
| Event Date | Date | Null | Present – Future dates | Will set date to null |
| Event Time Start | Time | Null | All times | Will use default time 12:00 am |
| Event Time End | Time | null | All times | Will use default time 12:00 am |
| Event Venue | Address | 50 | All addresses | Will set address to null |
| Soundcloud title | String | 50 | All characters |  |
| Soundcloud URL | String | 50 | All characters |  |
| YouTube title | String | 50 | All characters |  |
| YouTube URL | String | 50 | All characters |  |
| User Zip | Int | 5 | Int >0 | Will set to null or re prompt |

# 6.0 UML diagram

Included as separate file.

# 7.0 JavaDoc

Included as separate file.

# 8.0 Additional Topic

My additional topic was implementing my application as an Android application. This would be a more practical implementation of my application. If I were to create a desktop implementation, I would have to write the application using JavaScript. This is because the APIs for Soundcloud, Spotify and YouTube may or may not be officially supported for Java but are officially supported Android SDKs. While looking for an API for Java from Soundcloud and Spotify, I was only able to find third party wrappers.

# 9.0 Test plan

|  |  |  |
| --- | --- | --- |
|  | INPUT | EXPECTED |
| Event Date | 1/1/1970 | Re prompts user |
| Event Date | 12/25/2018 | 12/25/2018 |
| Event Time End | Null | 12:00 am |
| Event Time End | 1:00 pm | 1:00 pm |
| Event Time Start | Null | 12:00 am |
| Event Time Start | 6:00 pm | 6:00 pm |
| Event Venue | Null | Null |
| Event Venue | 123 Test Lane CA 12345 | 123 Test Lane CA 12345 |
| Keyword | “test” | Results including “test” |
| Playlist Title | “test” | “test” |
| Playlist Title | Null | Re prompts user |
| Song album | “test” | “test” |
| Song album | “test-album!” | “test-album!” |
| Song album | Null | Re prompts user |
| Song album art | Null | Default album art |
| Song artist | “test” | “test” |
| Song artist | “test-remix!” | “test-remix!” |
| Song artist | Null | Re prompts user |
| Song title | “test” | “test” |
| Song title | “test-artist!” | “test-artist!” |
| Song title | Null | Re prompts user |
| Soundcloud client ID | SPECIFIC TO APP | SPECIFIC TO APP |
| Soundcloud client secret | SPECIFIC TO APP | SPECIFIC TO APP |
| Spotify client ID | SPECIFIC TO APP | SPECIFIC TO APP |
| Spotify client secret | SPECIFIC TO APP | SPECIFIC TO APP |
| YouTube API key | SPECIFIC TO APP | SPECIFIC TO APP |
| YouTube client ID | SPECIFIC TO APP | SPECIFIC TO APP |
| YouTube client secret | SPECIFIC TO APP | SPECIFIC TO APP |
| Soundcloud API key | SPECIFIC TO APP | SPECIFIC TO APP |
| Spotify API key | SPECIFIC TO APP | SPECIFIC TO APP |
| ZipCode | “” | Null |
| ZipCode | 123456 | Re prompts user |
| ZipCode | 12345 | 12345 |

# 10.0 Installation Guide

This application is intended for devices with a minimum of 15. Any device with an SDK lower than 15 will not be able to install the device.

In order for a user to install the application, the user will have to go to the Google Store and download the application. The application will ask for the user to access their local files in order to read media off the device’s external storage. There is code inside the program that allows the application to use Internet, access network state and get accounts. Once installed the application should be able to work without any problems.

# 11.0 User Guide

When first opening the application, the user will be prompted to connect to their accounts for Soundcloud, Spotify and YouTube. A premium Spotify account will be needed to stream and view their data from Spotify. Connecting their Soundcloud and YouTube accounts will allow the user to see their online playlists. An account will not be needed to stream songs from Soundcloud or YouTube. A user will be able to search throughout their local library or between the three music streaming services.

The newsfeed will contain events or new songs of artists. Clicking an event will show information about the event. From there they will be able to buy the ticket if they want to. Clicking a song will start to stream the song.

Users may edit their location by going to the user page and clicking their location.

The music player allows for the user to pause, play, skip, previous, repeat, shuffle or change the volume. This depends all on the button they are pressing.

# 12.0 Project Analysis

I designed this application originally as a non-mobile application. After trying to implement so Android devices may use it, some of my design was unusable. I had to create some classes that I did not create in my design process. An example of this would be an Adapter and a Card class for the Songs, Artists and Results. I had to learn about intents and extras for intents. Creating the graphic user interface was also a lot different than normal Java implementation. Using an API from YouTube also tinkered with my design process.

I created a Card class for the search results from YouTube that required on OnItemClickListener method that allowed the user to open the YouTube video. I imagine that the process would be similar for Spotify and Soundcloud. I did not think of this during the design process.

The only API that I was able to get working was the YouTube API. I was not able to get the OAuth working for YouTube and Spotify. Soundcloud was not allowing people to register for API usage.

I learned a lot from the design process. I think if from the beginning I began designing this project with Android implementation in mind, I would be able to utilize my design to the full extent.