
Environmental Heights

AlbumTunes Software Requirements Specification

Version 1.0

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Revision History

Date	Version	Description	Author
09/29/2016	1.0	Initial Document	Kyle Neuman

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Table of Contents

1.	Introduction	4
1.1	Purpose	4
1.2	Scope	5
1.3	Definitions, Acronyms and Abbreviations	5
1.4	References	5
1.5	Overview	6
2.	Overall Description	6
2.1	Use-Case Model Survey	6
2.2	Assumptions and Dependencies	7
3.	Specific Requirements	8
3.1	Use-Case Reports	8
3.2	Supplementary Requirements	9

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Software Requirements Specification

1. Introduction

Environmental Heights has seen a growing need for a free lightweight and nimble media player that can manage large collections of songs. This demand is a result of both iTunes and Windows Media Player growing to immense sizes and hogging system resources.

Furthermore, data loss is an increasing problem due to new computers being purchased by users, hardware failure and/or new versions of iTunes and Windows Media Player being downloaded. AlbumTunes attempts to overcome this problem with playlist and song data being stored in a single XML file that can be easily moved to a new computer.

This document is the Software Requirement Specification (SRS) for the media player AlbumTunes. It will outline all of the currently existing features as well as features that need to be developed in the future.

1.1 Purpose

This document serves the purpose of outlining all of the requirements of the finished AlbumTunes product. The requirements will be broken out into four sections as outlined below:

1. Songs management and playback
 - a. Play (button)
 - b. Pause (button)
 - c. Restart (button)
 - d. Dragable trackbar representing duration
 - e. Trackbar back button (replays song)
 - f. Trackbar forward button (Plays next song)
 - g. Create playlist (button)
 - h. Delete playlist (delete button keyboard event)
 - i. Add songs to playlist (button and TableView)
 - j. Shuffle (checkbox)
 - k. Search box (text field)
 - l. Search for MP3s on machine (button)
 - m. Delete song from user defined playlist (delete button keyboard event)
 - n. Delete song from main playlist (delete button keyboard event)
 - o. Play song from playlist (double click mouse event)
 - p. Organize songs based on column headers
 - q. Display song duration each second
 - r. Display song metadata (Label)
 - s. Highlight song currently being played in Metadata Table
 - t. Edit song metadata
 - u. Edit playlist name

2. Preferences

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

- a. Option to close program from File > close
 - b. Option to change preferences from Edit > Preferences
 - c. Save data to XML file upon closing the program
 - d. Backup playlist button (button/text field)
 - e. Allow user to choose a different background image (button/text field)
 - f. Provide option to set all values back to default (button)
3. Supplementary Requirements
 - a. Metadata Table
 - b. Playlist Table

1.2 Definitions, Acronyms and Abbreviations

Term definitions:

Trackbar – slider contained within a bar that can be dragged back and forth while taking note of its precise position

Dragable – The ability of an object to be moved by holding the primary mouse button down, moving the mouse some distance and then letting go of the primary mouse button when the desired position is reached

TableView – A viewable data structure provided by the Java programming language. This structure is comprised of rows and columns used to display data to a user

Playlist – A list of songs that is named and managed by the user

Main Playlist – A type of playlist. The main playlist is a master list of all the songs present in AlbumTunes. This playlist has a default name of “ALL MUSIC,” but can be renamed. All imported music shall be imported to this playlist. Any song deleted from this playlist shall also be deleted in all other playlists. This playlist cannot be deleted. A user can add as many songs as they like

User Defined Playlist – A type of playlist. A user defined playlist is a list of songs created, named and managed by the user. A user can create as many user defined playlists as they choose. A user can delete a user defined playlist. If a user deletes a song from a user defined playlist, the song shall not be removed from any other playlist unless the user explicitly deletes the song from another playlist. A user can add as many songs as they want.

Album – A list of songs set to be played. This list is generated by the system and may be shuffled or played sequentially depending on the settings. The word album does not refer to the name of a retail compilation of songs created by an artist, unless otherwise specified with the: (retail).

Column Header – The name of the column seen on the TableView data structure. The first row of the TableView data structure contains the column headers. Column headers in AlbumTunes are: Song Name, Artist, Album (retail), Duration, Playlists.

Song metadata – The information depicting the song file. This metadata is displayed in the TableView data structure. Song metadata is as follows: name of the song, artist of the song, album (retail) the song is a member of, duration of the song.

XML information file – This file contains a list of all the playlists (main playlist and user defined playlists). Each playlist contains a list of songs contained in the playlist. This file is read and written each time AlbumTunes is launched or closed.

First song – Indicates the song at the top of the metadata table unless specified otherwise

Next song – The next song in the album generated by the system

Previous song – The previous song in the album generated by the system

States of AlbumTunes – AlbumTunes can be in only 1 state at a time. Only specific features can change the state of the player. The states of AlbumTunes are as follows: Pause, Play and Stopped.

Pause – A previously playing song is in a state of suspended playback until the user changes the state to Play

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Play – The state of a song file being read and played actively

Stopped – The state of the player as soon as AlbumTunes is launched. Once the user changes the state to Play, the system cannot return to the Stopped state unless the user closes AlbumTunes and re-launches the application

Music files – A music file ONLY refers to the song file types supported by AlbumTunes. These file types can be found in section 2.2 Assumptions and Dependencies.

1.3 References

1. Oracle. (2008, 2014). Package javafx.scene.media. Oracle. Retrieved from: <https://docs.oracle.com/javafx/2/api/javafx/scene/media/package-summary.html>
2. Oracle. (1993, 2016). Package javax.imageio. Oracle. Retrieved from: <https://docs.oracle.com/javase/7/docs/api/javax/imageio/package-summary.html>

1.4 Overview

AlbumTunes is a media player with many features a user can interface with. In addition to many playback and song organization options, AlbumTunes also provides user with some preferences and customization features. The system is designed to be used by those who both consider themselves computer savvy and those who do not.

2. Overall Description

2.1 Use-Case Model Survey

4. Songs management and playback
 - a. Play (button)
 - b. Pause (button)
 - c. Restart (button)
 - d. Draggable trackbar representing duration
 - e. Trackbar back button (replays song)
 - f. Trackbar forward button (Plays next song)
 - g. Create playlist (button)
 - h. Delete playlist (delete button keyboard event)
 - i. Add songs to playlist (button and TableView)
 - j. Shuffle (checkbox)
 - k. Search box (text field)
 - l. Search for MP3s on machine (button)
 - m. Delete song from user defined playlist (delete button keyboard event)
 - n. Delete song from main playlist (delete button keyboard event)
 - o. Play song from playlist (double click mouse event)
 - p. Organize songs based on column headers
 - q. Display song duration each second
 - r. Display song metadata (Label)
 - s. Highlight song currently being played in Metadata Table
 - t. Edit song metadata
 - u. Edit playlist name

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

5. Preferences

- Option to close program from File > close
- Option to change preferences from Edit > Preferences
- Save data to XML file upon closing the program
- Backup playlist button (button/text field)
- Allow user to choose a different background image (button/text field)
- Provide option to set all values back to default (button)

6. Supplementary Requirements

- Metadata Table
- Playlist Table

2.2 Assumptions and Dependencies

Assumptions:

1. user will have Java 1.8 or later installed on the target machine as a JVM is not bundled with the software.
2. User will be interfacing with AlbumTunes using a mouse/trackpad and keyboard (does not support touch based devices).
3. User has music files on the target machine and the user knows what directory/directories the music files are located in
4. Music files are limited to the capabilities of `□□□□□□ □□□□□□ □□□□□□`^[1]

a. Supported Encoding Types

An encoding type specifies how sampled audio or video data are stored. Usually the encoding type implies a particular compression algorithm. The following table indicates the encoding types supported by Java FX Media.

Encoding	Type	Description
AAC	Audio	Advanced Audio Coding audio compression
MP3	Audio	Raw MPEG-1, 2, and 2.5 audio; layers I, II, and III; all supported combinations of sampling frequencies and bit rates.
PCM	Audio	Uncompressed, raw audio samples
H.264/AVC	Video	H.264/MPEG-4 Part 10 / AVC (Advanced Video Coding) video compression
VP6	Video	On2 VP6 video compression

b. Supported Container Types

A container type specifies the file format used to store the encoded audio, video, and other media data. Each container type is associated with one or more MIME types, file extensions, and file signatures (the initial bytes in the file). The following table indicates the combination of container and encoding types supported by Java FX Media.

Container	Description	Video Encoding	Audio Encoding	MIME Type	File Extension
AIFF	Audio Interchange File Format	N/A	PCM	audio/x-aiff	.aif, .aiff
FXM, FLV	FX Media, Flash Video	VP6	MP3	video/x-javafx, video/x-flv	.fxm, .flv
HLS (*)	MP2T HTTP Live Streaming (audiovisual)	H.264/AVC	AAC	application/vnd.apple.mpegurl, audio/mpegurl	.m3u8

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

HLS (*)	MP3 HTTP Live Streaming (audio-only)	N/A	MP3	application/vnd.apple.mpegurl, audio/mpegurl	.m3u8
MP3	MPEG-1, 2, 2.5 raw audio stream possibly with ID3 metadata v2.3 or v2.4	N/A	MP3	audio/mpeg	.mp3
MP4	MPEG-4 Part 14	H.264/AVC	AAC	video/mp4, audio/x-m4a, video/x-m4v	.mp4, .m4a, .m4v
WAV	Waveform Audio Format	N/A	PCM	audio/x-wav	.wav

5. The supported image types are limited to the capabilities of the ☐☐☐☐ ☐☐☐☐☐☐/ ☐ ☐☐☐^[2]
- a. Supported operations

	Reading	Writing	Notes	Metadata
JPEG	yes	yes	none	JPEG metadata format
PNG	yes	yes	none	PNG metadata format
BMP	yes	yes	none	BMP metadata format
WBMP	yes	yes	none	WBMP metadata format
GIF	yes	yes	GIF plug-in notes	GIF metadata format

6. All windows shall adhere to the window design of the underlying OS. This means the close, restore and minimize commands shall mirror the windows displayed in the underlying OS. The shape, color and design of the windows shall adhere to the underlying OS and the order of the buttons (Cancel, OK, etc) shall adhere to the underlying OS.

3. Specific Requirements

3.1 Use-Case Reports

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

3.1.1 Song Management and Playback

Use Case Name: Play (button)

Summary: One of the following two events shall occur:

1. If the system is in the “stopped” state, the system shall start playing a selected song and queue up the rest of the current playlist for playback. The order of queued songs shall take into consideration whether Shuffle is selected or not. If selected, queued songs shall be shuffled. After the selected song begins to play, the words on the Play/Pause button shall change from “Play” to “Pause.” The song metadata shall be displayed to the user.
2. If the system is not in the “stopped” state, it is in the “paused” state, therefore the system shall Resume a currently paused song

Intent: As a user who intends to play music, this button shall either let me resume playing a paused song or initiate playback of a new song

Triggers: Actor clicks the button that reads “Play”

Precondition 1: No music is playing in the AlbumTunes application

Actor	User has just opened AlbumTunes
System	No song is playing and no song is paused in the AlbumTunes Application
Screen system is Stopped state	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic course of events (scenario 1)

Actor	User clicks the button labeled “Play”
System	Plays currently selected song and displays song metadata. Button label shall change from “Play” to “Pause.” Current song duration shall be displayed above the Trackbar.
Screen showing player after song started playing	<picture>

Precondition 2. A song is currently paused in the AlbumTunes application

Actor	User has pressed the button that reads “pause”
System	Song metadata and current duration is displayed. Button that reads “Play” is present
Screen showing player in the Paused state	<picture>

Basic course of events (scenario 2)

Actor	User presses the button that reads “play”
System	Currently paused song resumes playback. Button that reads “Play” is changed to “Pause”
Screen showing player in the Play state	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Play Song from Metadata Table

Summary: The purpose of this feature is to give the user the capability to play any song from the metadata table by double clicking the song. This button shall change the state of the media player to “play” no matter what state the player was in before triggering this event. The player shall stop any song that is playing or paused and shall commence playing the song selected (song is double clicked) by the user immediately. Upon selection, the media player shall carry out exactly the same chain of events outlined in the Use Case: Play (button)

Intent: As a user who intends to play music, this interactive row shall allow me to play any song by double clicking on the row I would like to play.

Triggers: The user double clicks (with primary mouse button – usually left button) a song in the metadata

Precondition: none

Basic Chain of Events (Scenario)

Screen of nothing playing	<picture>
User	user double clicks (with primary mouse button – usually left button) a song in the metadata
System	The system stops the currently playing or paused song if a currently playing or paused song exists. Then the system shall commence playing the selected song immediately in exactly the same way as outlined in the Use Case: Play (button).
Screen of song playing after triggers event	<picture>

Use Case Name: Pause (button)

Summary: Resume playback of a previously playing song

Intent: As a user who wants to pause music, the pause button shall allow me to pause a currently playing song.

Triggers: User presses the button that reads “Pause”

Preconditions: A song has begun playback. A button that reads “Pause” shall be present

Basic course of events (Scenario)

Actor	User pressed the button that reads “Pause”
System	Currently playing song shall be paused. Button that reads “Pause” shall be changed to “Play”
Screen system in the Paused state	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Restart (button)

Summary: One of the following events shall occur:

1. If the box labeled “Shuffle” is not selected, AlbumTunes shall stop the current song (if a song is playing) and begin playing the first song in the currently selected playlist.
2. If the box labeled “Shuffle” is selected, AlbumTunes shall stop the currently playing song (if a song is playing) and start playing the first song in the currently selected playlist, but shall continue playing songs in a shuffled order thereafter.

Intent: As a user who wants to listen to a playlist from the beginning, this button shall allow me to restart the currently playing playlist from the beginning.

Triggers: Actor clicks the button labeled “Restart”

Precondition 1: Box labeled “Shuffle” is not selected. A song may or may not be playing/paused.

Actor	Actor has selected the box labeled “Shuffle”
System	A song may or may not be playing/paused. If song is playing/paused, subsequent songs shall be playing sequentially in the order seen in the metadata table
Screen of shuffle deselected in the Paused state	<picture>

<picture>

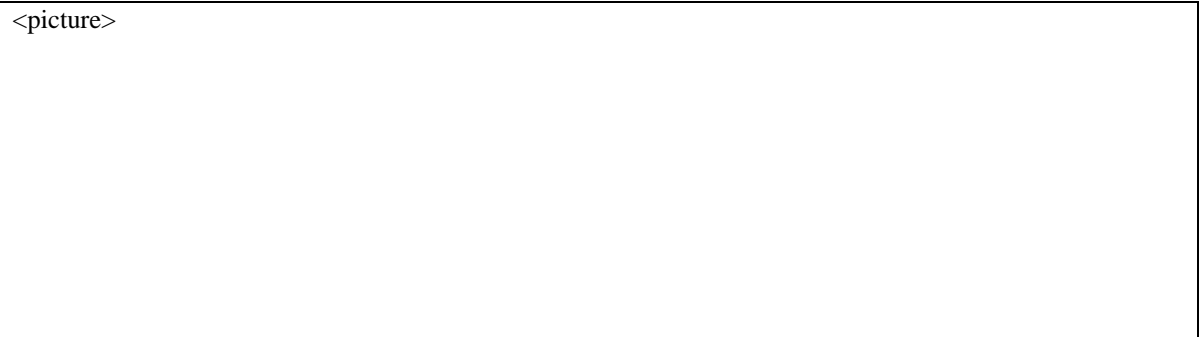
Basic course of events (scenario 1)

Actor	Actor clicks the button labeled “Restart”
System	AlbumTunes shall play the first song in the currently selected playlist and shall play subsequent songs in a sequential order thereafter as seen on the metadata table
Screen of first song playing in the playlist	<picture>

Precondition 2: Box labeled “Shuffle” is selected. A song may be playing or paused

Actor	Actor has selected the box labeled “Shuffle”
System	A song may be playing or paused. Subsequent songs shall be played from the currently selected playlist in a shuffled order.

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Screen showing the shuffle checkbox selected and system in the Play state	<p><picture></p> 
--	---

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic course of events (scenario 2)

Actor	Actor has selected the button labeled “Restart”
System	The first song in the currently selected playlist shall be played. Subsequent songs in the currently selected playlist shall be played in a shuffled order thereafter
Screen of first song in playlist playing	<picture>

<picture>

Use Case Name: Draggable Trackbar Representing Duration

Summary: The purpose of the draggable trackbar is multifaceted. The trackbar shall be used to display the current duration of the song each second as a string above the trackbar. The trackbar shall also update its position relative to the portion of the song completed (i.e. if half the song has been played back, the position of the trackbar should be half the distance from its start and end points). The trackbar shall also be interactive by allowing the user to click and drag the trackbar to a different position. AlbumTunes shall seek to the duration chosen by the user and commence playing through the song from that point. The draggable trackbar shall not change the current state of the player (i.e. if music is not playing, currently playing, or paused, the draggable trackbar shall not change that state).

Intent: As a user who wants to skip to a certain part of a song, this trackbar shall allow me to navigate to a certain part of a song by clicking and dragging the trackbar to a new position.

Triggers: User shall click and drag the trackbar to a new position

Precondition: A song is currently paused or playing

Actor	Actor began playing a song. Actor can then choose to pause the current song or let it continue to play
System	System is either currently playing a song or a song is currently paused
Screen of system in the Play state	<picture>

Basic course of events (Scenerio)

Actor	Actor has clicked and dragged the trackbar to a new position
System	AlbumTunes shall seek to the duration chosen by the user. If the current state of the player is “playing,” AlbumTunes shall resume playback at the duration chosen by the user. If the current state of the player is “paused” AlbumTunes shall seek to the duration chosen by the user, but shall not resume playback of the song from the duration specified by the user until the user clicks the button labeled “Play.” In either of these two cases, the current duration of the song shall be changed to reflect the new duration chosen by the user. If AlbumTunes is not paused or not currently playing a song AlbumTunes shall do nothing.

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Screen of same song playing at a longer duration	<picture>
---	-----------

<picture>



AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Previous Song Button

Summary: One of the following shall occur:

1. If a song has played past a duration of 1 second and the user presses the button labeled "<<", the song shall restart beginning at the duration 0.00. If the button labeled "<<" is pressed while AlbumTunes is paused, the duration shall be set to 0.00, but the song shall not commence playback until the user presses the button labeled "Play."
2. If a song has not played past a duration of 0.01 seconds and the user presses the button labeled "<<", the previously played song shall start from the beginning. If AlbumTunes is paused, the previous song shall be queued, but shall not commence playback until the user selects the button labeled "Play."

Intent: As a user who wants to replay a previous song or restart the current song, clicking this button shall allow me to replay a previous song or restart the currently playing song.

Trigger: User clicks the button labeled "<<"

Precondition 1: The duration of the current song is > 1 second.

Actor	User has pressed the button labeled "Play" and let the song play for at least 1 second. User may or may not have pressed the pause button thereafter
System	The song metadata is displayed and the duration above the trackbar reads a value greater than or equal to "0.01"
Screen of song past duration of 0.01	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic chain of events (Scenerio 1):

Actor	User clicks the button labeled "<<"
System	The system shall seek to the duration 0.00 seconds. If the system is paused, playback shall not commence until the user has pressed the button labeled "Play." If the system is playing, the next song shall seek to 0 seconds and commence playback immediately thereafter.
Screen of same song reset back to duration of 0.00	<picture>

Precondition 2: The duration of the current song is < 1 second.

Actor	Player has played through or skipped at least 1 song. The song currently playing or paused has a current duration that is less than 1 second.
System	The song metadata is displayed and the duration above the trackbar reads a value less than or equal to "0.01"
Screen of song with current duration that is less than 0.01	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic chain of events (Scenario 2):

Actor	User clicks the button labeled "<<"
System	The system shall queue the previously played song. If there is no previous song to queue for any reason, the system shall carry out the events outlined in scenario 1. After the song has been queued, the system shall commence playback of the queued song immediately. If the system is paused, the system shall not commence playback after the song has been queued until the user presses the button labeled "Play"
Screen of previous song queued and set to a duration of 0.00	<picture>

<picture>

Use Case Name: Next Song Button

Summary: This button shall be labeled ">>." This button shall skip to the next song in the playlist. If the box labeled shuffle is not checked, the next song in the playlist shall be song displayed below the currently playing song in the metadata table. If the shuffle box is checked, the next song could be any song in the playlist that has not already been played. If there is not a "next song" because the currently playing song is the last song in the playlist, AlbumTunes shall end the current song and the song metadata display should read "Album Finished." The Next Song Button shall not change the current state of the player (i.e. if music is not playing, currently playing, or paused, the Next Song Button shall not change that state).

Intent: As a user who wants to skip the current song, this button shall allow me to begin playing the next song in the sequence by clicking the button labeled ">>".

Trigger: User clicks the button labeled ">>"

Precondition: A song has started playing

User	The user began playing a song. The user may or may not have paused the song thereafter
System	The current song metadata shall be displayed and the current duration of the song shall be displayed above the trackbar
Screen of system after song has started	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic Chain of Events(Scenerio):

User	User clicks the button labeled ">>"
System	If the box labeled "Shuffle" is not selected, the system shall play the song below the currently playing song in the metadata table. If the box labeled "Shuffle" is selected, any song in the current playlist that has not been played yet shall be played. If a song is currently playing, the system shall stop the current song and immediately and commence playing the next song. If a song is currently paused, the next song shall be queued, but shall not commence playback until the user presses the button labeled "Play."
Screen of next song playling	<picture>

<picture>

Use Case Name: Create Playlist (button)

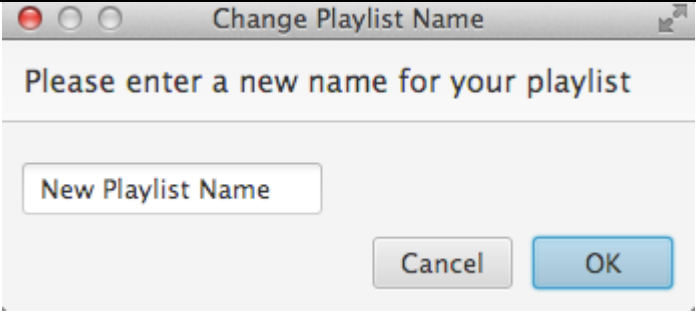
Summary: The purpose of this button is to give the user the ability to add a new user defined playlist. Upon pressing the button labeled "+Add Playlist" The user shall be prompted to enter a name for the new playlist. The user shall enter the name and then press enter. This shall add a new playlist to the playlist table that has the name defined by the user.

Intent: As a user who wants to organize my music, this button shall allow me to create playlists with custom names which shall hold songs of my choosing by clicking the "+Add Playlist" button.

Triggers: User clicks the button labeled "+Add Playlist." User enters the name of the new playlist and presses the "Enter" button on the users keyboard

Preconditions: None

Basic Chain of Events (Scenerio):

Screen before adding playlist	<picture>
User	User clicks the button labeled "+Add Playlist"
System	The system prompts the user for a playlist name
Screen prompting user for playlist name	
User	User presses the "Enter" key on their keyboard after they have finished typing their desired playlist name or clicks the OK button.

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

System	The system creates a new playlist that has the name defined by the user and displays the new playlist in the playlist table
Screen after adding playlist	<picture>

Use Case Name: Delete Playlist (Delete button keyboard event)

Summary: The purpose of this feature is to give the user the option to delete a playlist that they have created. The user must select the playlist they would like to delete in the playlist table and then press the “delete” or “backspace” key on their keyboard. They shall be prompted with a confirmation window to confirm their choice to delete the playlist. The user shall not have the authority to delete the main playlist. The user shall be prompted with an error message indicating that the user cannot delete the main playlist

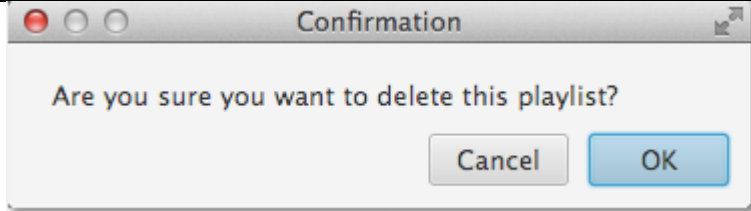
Intent: As a user who wants to organize my music, this feature shall allow me to delete an entire playlist of music that I no longer want by selecting the playlist and pressing the delete or backspace button.

Triggers: The user selects a playlist and then presses the “delete” or “backspace” key on their keyboard while the playlist is still selected. The user clicks the “yes” or the “no” button on the window confirming the user’s decision to delete the playlist selected playlist.

Precondition: The user must have created at least 1 user defined playlist

User	User created a user defined playlist following the step outlined in the Create Playlist (button) use case
System	System displays the user defined playlist in the playlist table
Screen showing user defined playlist	<picture>

Basic Chain of Events (Scenerio):

User	User selects a playlist that the user wants to delete. Then user presses the “delete” or “backspace” key on the user’s keyboard
System	The system prompts the user with “yes” and “no” buttons that ask the user to confirm the user’s choice to delete the selected playlist
Screen prompting user to confirm the user’s choice to delete the selected playlist	
User	User presses the “yes” or “no” button

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

System	If the user selects the “yes” button, the system shall delete the selected user defined playlist and remove the playlist from the playlist table. If the user selects the “no” button, the system shall do nothing, no playlist shall be deleted
Screen showing playlist removed	<picture>



AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Add Songs to Playlist (button and TableView)

Summary: The purpose of this feature is to give the user the capability of adding songs of the user's choosing to a playlist of the user's choosing. The user shall select the song or songs that the user intends to add to the playlist. The user shall then select the playlist that the user intends to add the songs too. Lastly, the user shall press the button labeled "Add songs to playlist."

Intent: As a user who wants to organize my music, this feature shall allow me to add specific songs to a playlist of my choosing by highlighting the playlist and songs, then pressing the "Add Songs to Playlist" button.

Trigger: The user selects a playlist from the playlist table, and the user selects one or more songs from the metadata table, and the user clicks the button labeled "Add Songs to Playlist"

Precondition: The user must have 1 or more songs in the metadata table

Basic chain of events (Scenerio)

User	The user selects a playlist from the playlist table, and the user selects one or more songs from the metadata table, and the user clicks the button labeled "Add Songs to Playlist"
System	The system copies the selected songs from the metadata table and pastes the songs from the metadata table to the selected playlist
Screen of selected playlist, selected songs and "Add Songs to Playlist" button	<picture>
Screen of songs added to selected playlist	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Deleting Song From User Defined Playlist

Summary: The purpose of this feature is to give the user the capability to delete a song from a user defined playlist. The user shall select one or more songs in the playlist they wish to delete. After selecting one or more songs, the user shall press the “delete” or “backspace” key on the user’s keyboard. The system shall remove the selected songs from the playlist.

Intent: As a user who wants to organize my music, this feature shall allow me to delete songs from a playlist that I made, but no others by selecting the song/s and pressing the delete or backspace key.

Triggers: The user selects one or more songs from the user defined playlist. Then the user presses the “delete” or “backspace” key on the user’s keyboard.

Preconditions: The user has created a user defined playlist and added one or more songs to the user defined playlist

Basic Chain of Events (Scenario)

User	The user selects one or more songs to be deleted from the user defined playlist
System	The system deletes the songs from the playlist
Screen of songs selected for deletion	<picture>
User	The user presses the “delete” or “backspace” key to delete the songs from the user defined playlist
System	The system removes the song or songs from the user defined playlist
Screen of songs removed from playlist	<picture>

Use Case Name: Delete Song from Main Playlist

Summary: The purpose of this feature is to avoid orphaned songs in the system. If the user deletes a song from a user defined playlist, the song is only deleted from that playlist and no other as outlined above. On the other hand, if the user deletes a song from the main playlist (this playlist shall have the default name “All Music”), then the deleted song or songs shall be removed from the main playlist and all other user defined playlists. This means the main playlist shall contain any song found in a user defined playlist.

Intent: As a user who wants to remove a song from AlbumTunes, this feature shall allow me to remove a song from the main playlist and all other playlists in AlbumTunes by selecting the song/s and pressing the delete or backspace key.

Triggers: The user selects one or more songs from the main playlist and then presses the “delete” or “backspace” keys

Preconditions. The user must have imported one or more songs into AlbumTunes

Basic Chain of Events (Scenario)

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Screen of songs in user defined playlist	<picture>
User	User selected one or more songs in the main playlist
System	
Screen of selected songs in main playlist	<picture>
User	User presses the “delete” or “backspace” key on the user’s keyboard
System	The system shall delete the selected song or songs from the main playlist and all other user defined playlists
Screen of songs deleted from main playlist	<picture>
Screen of songs deleted from user defined playlist	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Shuffle (checkbox)

Summary: The purpose of this feature is to give the user the capability of shuffling the sequence of songs being played back in a playlist. If the shuffle checkbox becomes selected, it shall trigger the system to stop the current song and queue a shuffled sequence of all the songs in the current playlist. If the shuffle checkbox becomes deselected, the system shall continue playing the current song. A sequence of songs shall be queued starting from the position of the currently selected song to the end of the playlist in the order seen on the metadata table. The shuffle checkbox shall not change the state of the system. If the system is in the stopped state, and the shuffle checkbox becomes selected by the user, the system shall queue a sequence of songs, but shall not commence playback until the user presses the button labeled “Play.” If the system is in the paused state, the system shall queue a sequence of songs, but shall not commence playback until the user presses the button labeled “Play.” If the system is in the play state, and the shuffle checkbox becomes selected by the user, the system shall queue a sequence of songs, and shall commence playback immediately.

Intent: As a user who wants to listen to music in a shuffled order, this checkbox will allow me to listen to each song in the metadata table, but in a shuffled order by clicking the checkbox.

Triggers: The user either selects or deselects the checkbox labeled “shuffle.”

Precondition: None

Basic Chain of Events (Scenerio)

User	User select the checkbox labeled “shuffle”
System	The system queues a shuffled sequence of all the songs in the current playlist, but shall not change the state of the current player.
Screen	<picture>
User	The user deselects the checkbox labeled “shuffle.”
System	The system queues a sequence of songs starting from the position of the currently selected song to the end of the playlist in the order seen on the metadata table.
Screen	<picture>

Use Case Name: Search Box (text field)

Summary: The purpose of the search box text field is to give the user the capability to narrow the contents displayed in the metadata table based on a phrase typed in to the search box. The user shall enter a phrase in the search box and press the “enter” button on the user’s keyboard. This shall trigger the system to search the contents of the metadata table (contents of a particular playlist). The search shall compare the phrase against every song name, artist name, and album (retail) name being displayed in the metadata table. If the phrase matches any of these fields, the song related to the field shall be added to a sublist. Once the system has finished searching the contents of the metadata table for matches, the sublist shall be displayed in the metadata table. In order to return to a full length playlist, the user shall delete all contents in the search box text field and press the “enter” key on the user’s keyboard. This shall populate the metadata table with the contents of the currently selected playlist in the playlist table.

Intent: As a user who only wants to play a specific song or album, this search box shall allow me to type a phrase into the box and narrow the results found in the metadata table.

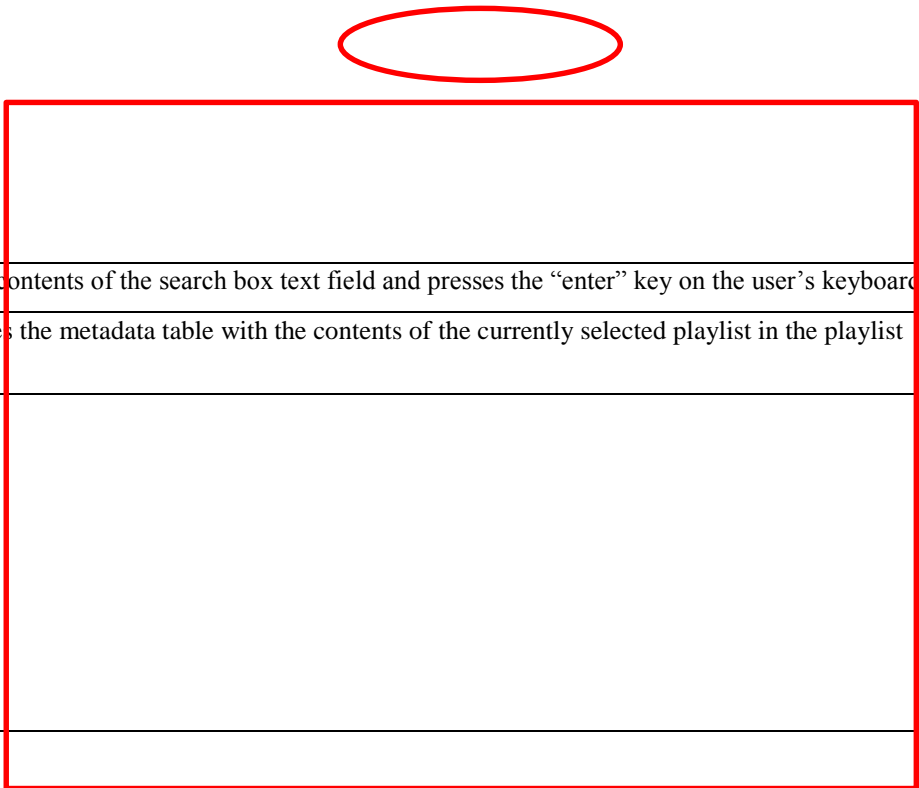
Triggers: User types a phrase into the search box text field and presses the “enter” button on the user’s keyboard. The user deletes the contents in the search box text field and presses the “enter” key on the user’s keyboard.

Preconditions: None

Basic Chain of Events (Scenerio)

User	User types a phrase into the search box text field and presses the “enter” key on the user’s keyboard
-------------	---

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

System	The system shall search the contents of the metadata table and return a sublist of songs matching the phrase entered by the user
Screen showing metadata before table results have been narrowed	<picture>
Screen showing sublist of songs matching user's search phrase	<div> <div><picture></div> <div>  </div> </div>
User	User deleted all the contents of the search box text field and presses the "enter" key on the user's keyboard
System	The system populates the metadata table with the contents of the currently selected playlist in the playlist table
Screen showing user deleting all contents of the search box text area	<picture>
Screen showing metadata table populated with contents of currently selected playlist	<picture>

<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Search for MP3s on Machine (button)

Summary: The purpose of this feature is to allow the user the capability to import songs into AlbumTunes. Upon clicking the button “Mine for mp3s,” The user shall receive a series of prompts. The first prompt shall provide three buttons and an explanation for each. The three buttons shall be labeled “Single mp3,” “Folder of mp3s,” and “Cancel” respectively. Selection of each button is outlined below:

1. Single mp3 – Selection of this button shall prompt the user with a file selection window where the user shall select a single music file and click the button labeled “Open” to import the file into AlbumTunes
2. Folder of mp3s – Selection of this button shall prompt the user with a directory selection window where the user shall select a directory containing music files and then click the button labeled “Open.” The directory the user chooses may have music files contained deep within the directory structure. The System shall search every file in every directory contained in the directory chosen by the user. This means the user could choose the root drive and the system shall find all music files contained on that drive. If the user selects this option, the songs shall be loaded asynchronously in a new thread to avoid freezing the system This operation could take several minutes for directories containing large (>3000) volumes of music files. The metadata table shall be updated for each song so the user can see how far along the system is at parsing the chosen directory.
3. Cancel – This button shall close the current prompt

All files imported from either the Single mp3 button or Folder of mp3s button shall be added to the main playlist in AlbumTunes.

Intent: As a user who wishes to play music in AlbumTunes, this button shall allow me to import music into AlbumTunes by selecting a music file or folder that contains music.

Triggers: User clicks the button labeled “Mine for mp3s.” Then the user clicks one of the three buttons, “Single mp3s,” “Folder of mp3s,” “Cancel.” The user then selects a file or directory respectively and clicks the button labeled “Open.”

Preconditions: None

Basic Chain of Events (Scenario): Button labeled “Mine for mp3s”

User	User clicks the button labeled “Mine for mp3s”
System	System prompts the user to make a selection between the buttons “Single mp3,” “Folder of mp3s,” and “Cancel”
Screen showing “Mine for mp3s” button	<picture>
Screen showing prompt for user to choose “Single mp3,” “Folder of mp3s,” “Cancel”	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Basic Chain of Events (Scenario): User selects “Single mp3”

User	User clicks the button “Single mp3”
System	Prompts the user with a file choosing window
Screen showing file selection window	<picture>
User	User navigates to the mp3 file they want to import. User clicks the button “Open”
System	The system shall import the file into the main playlist in AlbumTunes
Screen showing imported song	<picture>

Basic Chain of Events (Scenario): User clicks the button “Folder of mp3s”

User	The user clicks the button labeled “Folder of mp3s”
System	The system prompts the user with a directory selection window
Screen showing directory selection window	<picture>
User	The user selects a directory and clicks the button labeled “Select Folder”
System	The system shall import the all the music files contained within the directory. The music files may be nested deep within the directory structure of the chosen directory. The system shall import every music file contained in every subfolder of the chosen directory.
Screen showing imported files	<picture>

Basic Chain of Events (Scenario): User clicks the button “Cancel”

User	User clicks the button labeled “Cancel”
System	The system shall close the current prompt and return the user to AlbumTunes
Screen showing return to AlbumTunes	<picture>

<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Organize Songs Based on Column Headers in Metadata Table

Summary: The purpose of this feature is to give the user the capability of organizing their songs based on the information displayed in the metadata table. The user shall click on the column header name. Clicking the column header shall cause an up or a down arrow to appear indicating ascending order (up) or descending order (down). This shall re-arrange the order of all the songs (rows) in the metadata table. If the checkbox labeled shuffle is not selected, the system shall queue a new list of sequential songs starting with the selected song and ending with the last song in the metadata table to reflect the new order of sequential playback.

Intent: As a user who wishes to organize my music alphabetically, this feature shall allow me the freedom to organize my music alphabetically by song name, album name (retail), or artist name by clicking on column headers.

Triggers: User clicks the column header

Precondition: None: ascending and descending arrows shall still appear regardless of whether or not songs exist in the metadata table

Chain of Events (Scenario)

User	User clicks a column header in the metadata table
System	The system shall sort the songs (rows) in ascending order
Screen showing songs displayed in ascending order	<picture>
User	User clicks the same column header in the metadata table again
System	The system shall sort the songs (rows) in descending order
Screen showing songs displayed in descending order	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Display Song Duration Every Second

Summary: The purpose of this feature is to present the user with the current duration of the song being played or paused. This duration shall be presented as a string above the dragable trackbar. The string shall present the user with an updated duration every second and display the duration in the same way the user reads a standard digital clock (The duration shall be displayed *minutes.seconds* 2.45, rather than the decimal form 2.75 for example)

Intent: As a user who wishes to know how far along a song it, this feature will display the current duration to the nearest second and move the trackbar so I can get a feel for how much of the song is left.

Triggers: None

Preconditions: The user has begun playing a song. The user may or may not have stopped the song at any point thereafter

Basic Chain of Events (Scenario)

User	User begins playing a song
System	The system calculates the current duration and displays the duration in the form <i>minutes.seconds</i>
Screen of current duration	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Display Song Metadata (Label)

Summary: The purpose of this feature is to provide the user with easy and constant awareness of the current song being played or paused. As soon as the user begins playing a song, the song metadata shall be displayed to the user above the metadata table and all of the features except those that exist in the menu bar. This is done to increase visibility. The metadata label displayed is all the information contained in one row of the metadata table.

Intent: As a user who wants to know what song is playing at any time, this feature will display the song metadata at the top of AlbumTunes in a place I can't miss it.

Triggers: None

Preconditions: A song has begun playing. The user may or may not have stopped the song at any point thereafter

Basic Chain of Events (Scenario)

Screen before a song has been started	<picture>
User	User begins playing a song
System	The system displayed the song metadata in a label above all the other features except those the features contained in the menu bar
Screen of song metadata displayed in label	<picture>

Use Case Name: Highlight Song Currently Being Played in Metadata Table

Summary: The purpose of this feature is to show the user where the current song is being displayed in the metadata table. The song (row in metadata table) shall be highlighted when each song is played regardless of whether the user explicitly triggered the song to be played or if the song being played is the next song in the playback sequence. This feature shall deselect all currently selected songs in the metadata table upon a new song being played to avoid unexpected results from the user when transferring songs between playlists. This song shall become deselected if a user selects another song unless the user chooses to select multiple songs using hotkeys specific to the user's operating system.

Intent: As a user who wants to know where the current song is playing in the metadata table, this feature will highlight the currently playing song in the playlist until I select a different song.

Triggers: The user begins playing a song

Preconditions: None

Basic Chain of Events (Scenario)

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Screen showing a selected song that is being played	<picture>
User	User plays a different song
System	The system begins playing a different song and highlights the row corresponding to the currently playing song
Screen showing different selected song that is being played	<picture>

Use Case Name: Edit Song Metadata

Summary: The purpose of this feature is to provide the user with the capability to change the metadata relating to a song. This shall not change the playback of the song or the file itself. Changing the metadata shall persist from one use of AlbumTunes to another, but shall remain superficial as this information does not augment anything other than the information presented to the user. The metadata augmented shall be the Song Name, Artist and Album (retail). The song duration shall not be augmentable. The user shall right click on the song (row in metadata table) they would like to change the metadata for. They shall then be prompted in another window with the current metadata presented in editable text fields. The user shall have the choice to change the metadata and then either click the button labeled “OK” or “Cancel.” Selecting the button labeled “OK” shall replace the metadata in the row with the metadata chosen by the user. If the user selected the button labeled “Cancel” the system shall do nothing and close the window.

Intent: As a user who wants to customize the song data displayed in the metadata table, this feature shall allow me to change the metadata displayed to me in the metadata table by right clicking on a row.

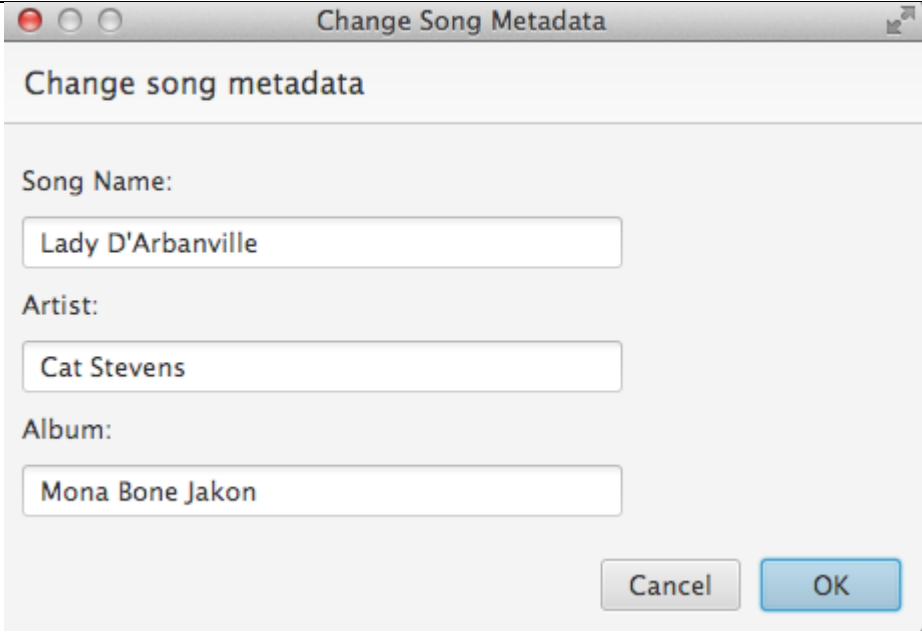
Triggers: The user shall secondary click (usually the right mouse button) a row in the metadata table. The user may or may not edit the metadata contained in the row in a new window. The user then may either click the button labeled “OK” or “Cancel”

Preconditions: The user must have imported one or more songs into AlbumTunes

Basic Chain of Events (Scenario)

Screen showing metadata before editing	<picture>
User	User right clicks on a song they would like to change the metadata for
System	The system loads the metadata contained in the selected row and populates the data in 3 editable text boxes that the user can chose to edit

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Screen showing new window with editable metadata	
User	User edits the information provided in the text box
System	The system saves the changes made to the selected row in the underlying data structure
Screen showing edited metadata in new window	<picture>
Screen showing changed metadata in metadata table	<picture>
User	The user clicks the button labeled cancel
System	The system makes no changes to the underlying data structure
Screen show the same metadata before the user right clicked the row	<picture>

Use Case Name: Edit Playlist Name

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	


Summary: The purpose of this feature is to give the user the capability to change the name of any playlist including the main playlist. The user shall right click on a playlist (row in playlist table). The user shall then be prompted to change the name of the playlist in an editable text field and then click the button labeled “OK” or “cancel.” If the user clicks the button labeled “OK” the system shall change the name of the selected playlist. If the user clicks the button labeled “cancel” the system shall close the window and do nothing.

Intent: As a user who wants to rename a playlist, this feature allows me to edit the name of any playlist by right clicking on the playlist.

Triggers: The user clicks a playlist the secondary mouse button (usually the right mouse button). The user then clicks the button labeled “OK” or “cancel” in a new window.

Precondition: None

Basic Chain of Events (Scenario)

Screen of playlist before right click	<picture>
User	User right clicks on a playlist
System	The system shall prompt the user with a new window to change the name of the playlist and two buttons, one labeled “OK” and the other labeled “cancel”
Screen of new window to edit playlist name	<picture> 
Screen of playlist name changed (OK button)	<picture>
Screen of playlist name not changed (Cancel button)	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

3.1.2 Preferences

Use Case Name: Option to Close Program with File > Close

Summary: The purpose of this feature is to give the user the option to close the program in some other way than closing the window using the means provided by the underlying OS. The user shall click the file dropdown menu and then they shall click on the option labeled “Close.” This shall terminate the program and shall trigger an event to save information to the hard drive. The saving of information shall be documented and covered in the use cases named: Save Playlists to XML Upon Closing, and Save Preferences to XML Upon Closing.

Intent: As a user who wants to close the program, this feature allows me the ability to close the program from the file menu.

Triggers: The user shall navigate to the File dropdown menu and click the option labeled “Close.”

Precondition: None

Basic Chain of Events (Scenario)

User	The user navigates to the file menu and clicks the menu option labeled “Close”
System	The system shall close the GUI and save information to the hard drive
Screen of close menu option	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Change Preferences from Edit > Preferences

Summary: The purpose of this feature is to give the user special options to interface with AlbumTunes. The specific features that shall be in the preferences menu shall be discussed in later use cases. The scope of this use case deals only in the accessing and presenting of the preferences menu. The preferences menu shall be launched in a new window and shall be accessed by navigating to the Edit menu located at the top left corner of the application. The user shall then click on the menu option labeled “Preferences”

Intent: As a user who wants to change the preferences of AlbumTunes, this feature shall allow me to access the preferences menu from Edit > Preferences.

Triggers: The user navigates to the Edit menu and then clicking on the option labeled “Preferences”

Basic Chain of Events (Scenario)

Screen of Edit > Preferences	<picture>
User	The user navigates to the Edit menu and then clicks on the option labeled “Preferences”
System	The system shall launch a new window containing all of the preferences available for the user to interface with
Screen of preferences options menu	<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Save Data to XML Upon Closing

Summary: The purpose of this feature is to save user and application data to the hard drive of the target machine. The data saved can be categorized into three groups:

1. Preferences data - location of the playlists XML file, the value of the checkbox labeled “Shuffle” and the location of the current background image.
2. song data – song name, artist, album (retail), duration and location on the host machine
3. Playlist data - contents of the playlists that contain songs, song order within the playlist and playlist names.

The user does not knowingly save the data to the disk, instead, upon closing the application, the user shall also be triggering the event to save the data. The data shall be saved if the user closes the application using File > Close or if the user closes the AlbumTunes window using the window interface options provided by the underlying OS (clicking on the red “X” at the top right corner of the window on windows operating systems). The data shall be saved to the hard drive in XML format version 1.0 using an encoding of UTF-8. The saving of playlist data shall be carried out in a new thread. The graphical user interface (GUI) shall be closed immediately, however the saving process shall continue until it is finished.

Intent: As a user who wants AlbumTunes to appear just as I left it last time I used AlbumTunes, this feature will save everything that I interact with in AlbumTunes.

Triggers: The user shall trigger the closing of the program either by navigating to the menu option File > Close or by closing the application through the window options provided by the underlying OS (red “X” for windows OS).

Precondition: None

Basic Chain of Events (Scenario)

Screen of system before closing	<picture>
User	User either navigates navigating to the menu option File > Close or by closing the application through the window options provided by the underlying OS (red “X” for windows OS).
System	The system shall close the AlbumTunes window but keep the main thread active. In a new thread, the system shall save the save the data to XML files and then exit the main thread
Screen of “Close” menu options	<picture>

<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Backup Playlist (Button)

Summary: The purpose of this feature is to give the user the ability to backup the playlist data that the user created in AlbumTunes. This file shall contain the song data and playlist data that was outlined in the use case named Save Data to XML Upon Closing. The user shall navigate to the preferences menu (Edit > Preferences) and then click the button labeled "Backup Playlist." The user shall then be prompted with a directory choosing window where the user shall navigate to the directory that they would like to save the backup playlist XML file to. The user shall then click the button labeled "OK" or "Cancel." The OK button shall save the XML data file to the chosen directory in a new thread so that the system does not freeze. The Cancel button shall close the directory choosing window and shall do nothing.

Intent: As a user who wants to save all my playlist data to a new location for use on a different machine, this feature shall allow me to save all my playlist data anywhere I want.

Trigger: The user navigates to the preferences menu (Edit > Preferences). In the directory choosing window, the user navigates to the directory that the user wants to save the backup XML data file to. The user then clicks either the button labeled, "OK" or "Cancel."

Preconditions: None

Basic Chain of Events (Scenario)

User	The user navigates to the preferences menu using the Edit menu bar and then clicking the preferences option
System	The system launches the preferences options in a new window
Screen of preferences window	<picture>
User	The user clicks the button labeled "Backup Playlists"
System	The system launches the directory chooser
Screen of directory choosing window	<picture>
User	The user navigates to the directory that the user wants to save the XML data file to
System	The system writes and saves the XML to the chosen directory in a new thread
Screen of file saved in chosen location	<picture>
User	The user clicks the button labeled "Cancel"
System	The system closes the file choosing window and does nothing

<picture>

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

Use Case Name: Allow User to Change Background Image (button/textfield)

Summary: The purpose of this feature is to allow the user the ability to customize the background image of AlbumTunes. The user shall navigate to the preferences menu (Edit > preferences) and then click on the tab labeled “Background Image Location.” From here, the user shall either paste an absolute path to the image they would like to use as their background and then press enter or click the button labeled “Change.” Clicking the change button shall prompt the user with a file choosing window where the user shall navigate to the file that the user would like to use as the background. The user shall then click the button labeled “OK” to change the background or “Cancel” to escape from the window.

Intent: As a user who wants to customize my experience with AlbumTunes, this feature allows me to change the background image of AlbumTunes to a picture of my choosing on my machine.

Triggers: User navigates to the preferences menu, then clicks on the tab labeled “Background Image Location.” From the “Background Image Location” tab, the user either clicks the button labeled “Change,” or pastes an absolute path to an image on their machine and presses the enter key on their keyboard.

Precondition: None

Basic Chain of Events (Scenario)

User	User clicks on the tab labeled “Background Image Location”
System	System changes to the “Background Image Location” tab
Screen of “Background Image Location” tab	<picture>
User	User clicks the button labeled “Change”
System	The system loads the image and displays the image as the background
Screen of file choosing window	<picture>
User	User navigates to the image file that they would like to use as their background and clicks the button labeled “Open”
System	The system loads the image and sets the image as the new background
Screen of new background	<picture>
User	User copies and pastes the absolute path of an image on their machine and presses the enter key on the user’s keyboard
System	The system loads the image and displays the image as the background
User	User clicks the button labeled “Cancel”
System	The system closes the window and does nothing

Use Case Name: Provide Option to Set All Preferences Back to Default (button)

Summary: The purpose of this feature is to allow the user the ability to change all the preferences back to default. The user shall navigate to the preferences menu and click the button labeled “Reset Preferences to Default.” Clicking the button labeled “Reset Preferences to Default” shall trigger the system to read in a default preferences XML file stored in the system resources. The default preferences file shall contain the following read-only information:

1. Shuffle checkback value – False. This shall deselect the shuffle checkbox if it is selected

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

2. Default image location – This shall contain a relative path to the default background image in the systems resources. This image shall ship with AlbumTunes
3. The default calculated parent info directory for AlbumTunes. If the default parent info directory doesn't exit, it shall be created.

Clicking the button labeled “Reset Preferences to Default” shall also calculate the root information directory of AlbumTunes and/or create the default information parent directory of the playlist and song data XML file and user defined preferences XML file. The root information directory shall be two parent directories removed from the AlbumTunes JAR file. The directory structure of the AlbumTunes information parent directory shall be as follows:
 MEDIA_PLAYER_5000/infoDirectory/. The directory named “infoDirectory” shall contain the playlist and song data XML file as well as the user defined preferences XML file. Diagram shown below.

Intent: As a user who wants to return to all the default settings that AlbumTunes ships with, this button shall reset all my preferences and background image.

Triggers: The user navigates to the preferences menu (Edit > preferences) and clicks the button labeled “Reset Preferences to Default.”

Preconditions: None

Basic Chain of Events (Scenario)

User	User navigates to the preferences menu (Edit > preferences) and clicks the button labeled “Reset Preferences to Default.”
System	The system reads the default preferences XML file, resets the default background image, resets the shuffle box to unselected, and calculates/creates the default parent directory for the playlist and song data XML file.
Screen of preferences menu	<picture>

<picture>

3.2 Supplementary Requirements

Supplementary Requirement Name: Metadata Table

Summary: The primary purpose of this feature is to provide the user with an intuitive means of interacting with AlbumTunes. The metadata table contains many use cases:

1. Play A Song From Metadata Table
2. Delete Song From user Defined Playlist
3. Delete Song From Main Playlist
4. Add Songs to Playlist
5. Sort Song Metadata
6. Organize Songs Based on Column Headers
7. Edit Song Metadata
8. Highlight song currently being played in Metadata Table

The secondary purpose of this feature is to display song metadata. This metadata shall include four columns displayed in the following order from left to right in the metadata table:

AlbumTunes<Project Name>	Version: 1.0
Software Requirements Specification	Date: 11/2/2016
EH-AT-001	

1. Song name – This column shall display the name of the song
2. Artist – This column shall display the artist of the song
3. Album – This column shall display the album (retail) the song belongs to
4. Duration – This column shall display the maximum duration of the song

The metadata table shall have the capability to sort and edit songs. The metadata table shall also consume the majority of the screen real-estate in AlbumTunes.

Intent: As a user who wants to an easy interaction with my music, this feature allows me to play/change/display songs while using album tunes.

Supplementary Requirement Name: Playlist Table

Summary: The purpose of this feature is to provide the user an interactive list of playlists the user has created. This table has only one column but contains many use cases:

1. Delete Song From user Defined Playlist
2. Delete Song From Main Playlist
3. Add Songs to Playlist
4. Add User Defined Playlist
5. Delete Playlist
6. Play A Song From Metadata Table
7. Edit Playlist Name
8. Main Playlist

The playlist table shall always have at least 1 playlist: the main playlist. This playlist's name can be changed but it cannot be deleted.

Intent: As a user who wishes to interact and modify the way I organize my music, this feature allows me display, change and organize my music into playlists.