

PROJECT FRONT SHEET

Class	Object Oriented Programming		
Submission date	22/04/2025		
Student Name	Nguyen Manh Hung	Student ID	001419592
Class	COMP-1752	Assessor name	Nguyen The Nghia
<input type="checkbox"/> Summative Feedback: Feedback:			
Grade:	Assessor Signature:	Date:	

Table of Contents

1. [Introduction](#)
2. [Design and Development](#)
 - a. [Overview](#)
 - b. [Project Structure](#)
 - c. [UI Design](#)
 - d. [Data Management](#)
3. [Development Stages](#)
 - a. [Stage 1: Basic Understanding](#)
 - b. [Stage 2: Outline Implementation](#)
 - c. [Stage 3: Basic Working Version](#)
 - d. [Stage 4: Testing and Validation](#)
 - e. [Stage 5: Innovations](#)
4. [Testing and Validation](#)
5. [Conclusions, Further Development and Reflection](#)
6. [Appendix](#)
 - A. [Commented Code for Stage 1](#)
 - B. [Test Table and Results](#)
7. <https://github.com/Hungloi56565/COMP1752-COURSEWORK-Nguyen-Manh-Hung>

1. Introduction

This report presents the development of a Jukebox simulation application designed to manage and organize a music library. The project was developed using Python, with the Tkinter library utilized to build a user-friendly graphical interface. The application enables users to browse music tracks, create and manage playlists, and update track metadata efficiently.

Track data is stored and managed using CSV files:

- **music.csv**: Main track database containing track name, artist, and rating
- **user.csv**: User login information
- **list1.csv, list2.csv**: Saved playlists with selected track IDs and status

The development process followed the five key stages outlined in the coursework, integrating various features such as a unified GUI design, dynamic playlist control, track search capabilities, and data validation. The application meets all fundamental requirements and incorporates enhancements aimed at improving usability and functionality.

2. Design and Development

a. Overview

The Jukebox application was designed as a single-window interface utilizing multiple tabs to streamline navigation and functionality. This integrated approach improves usability and provides a consistent user experience, compared to earlier designs that relied on multiple separate windows. The application's core components include:

- **View Tracks**: Displays all available tracks with integrated search and filtering options
- **Playlists**: Supports the creation, editing, and saving of user-defined playlists
- **Update Library**: Offers tools to add new tracks and update existing track information, including validation

This modular design promotes maintainability while ensuring all essential features are easily accessible.

b. Project Structure

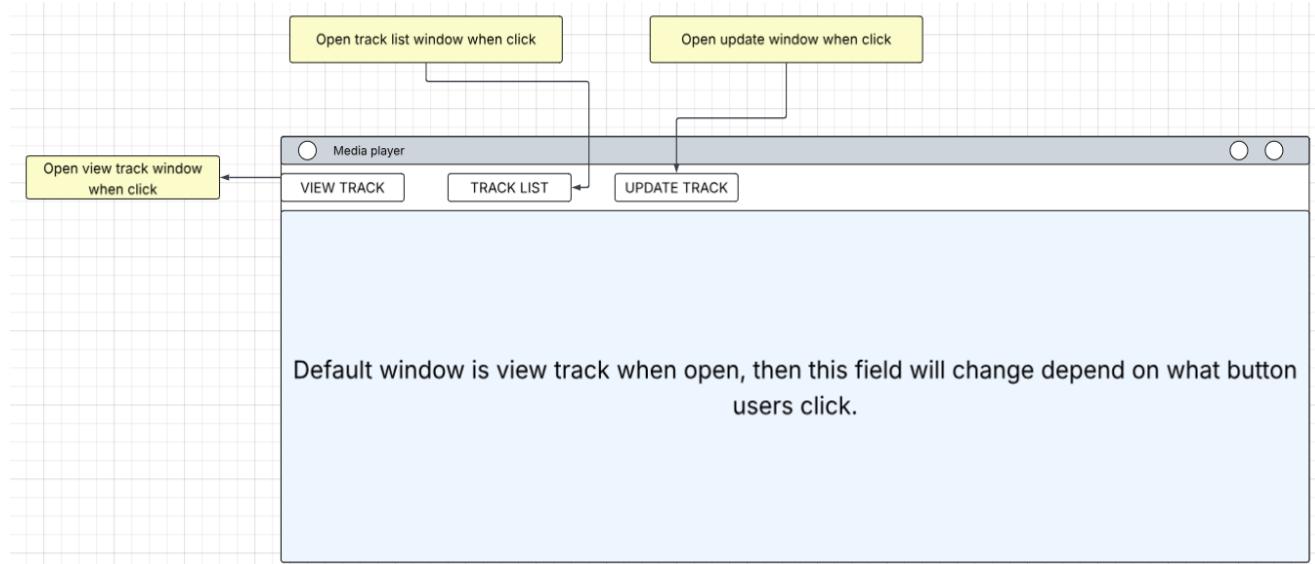
The application is organized into several Python files, each fulfilling a distinct role in the system:

- **Login.py:** Handles user authentication and registration (additional feature)
- **jukebox.py:** Main entry point that initializes the graphical user interface and integrates core components
- **library_item.py:** Defines the LibraryItem class representing individual music tracks
- **track_library.py:** Manages the music database and provides functions to retrieve and update track data
- **viewtrack.py:** Enables users to view, search, and display track details
- **tracklist.py:** Manages playlist creation, track selection, and playlist persistence
- **update.py:** Allows users to add, edit, and remove tracks from the music library
- **font_manager.py:** Applies custom font settings across the application interface

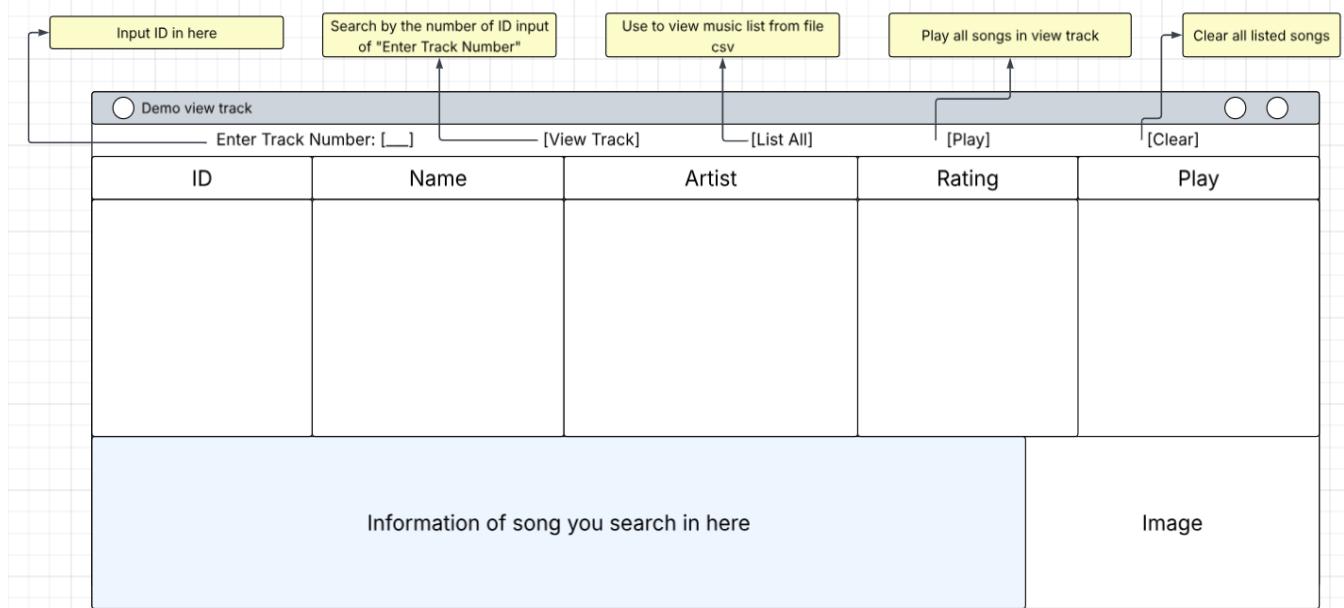
c. UI Design

The interface was designed using Tkinter with a modern, user-friendly layout. Key design elements include:

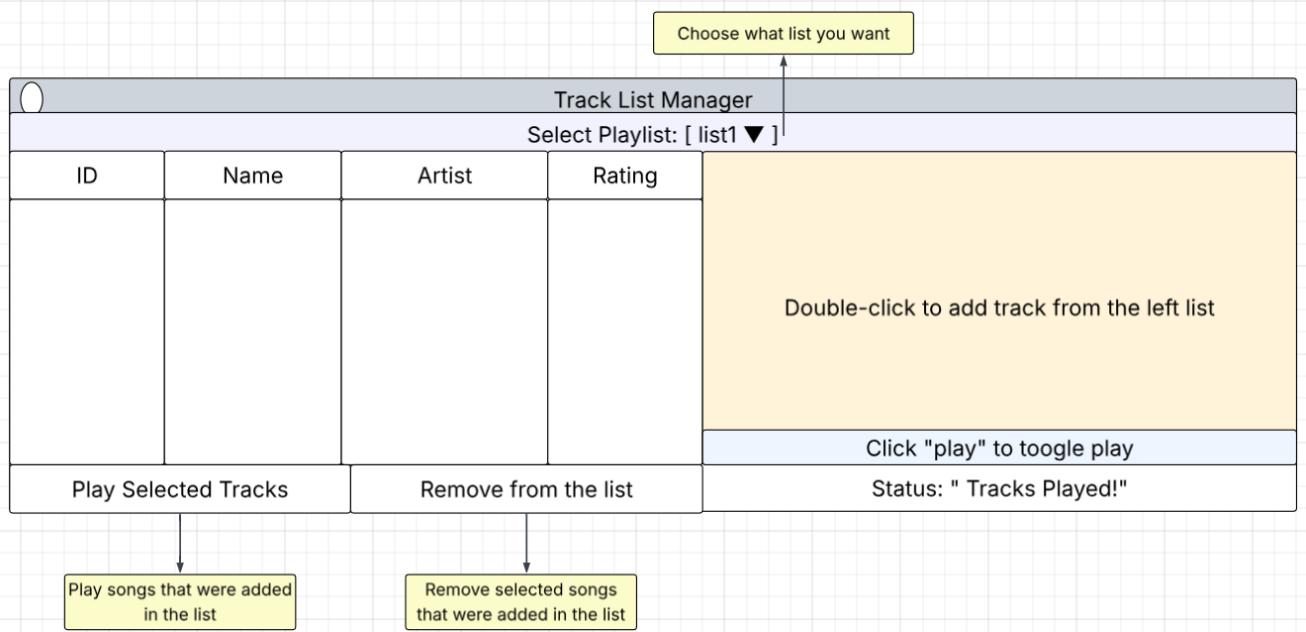
1. Main Interface



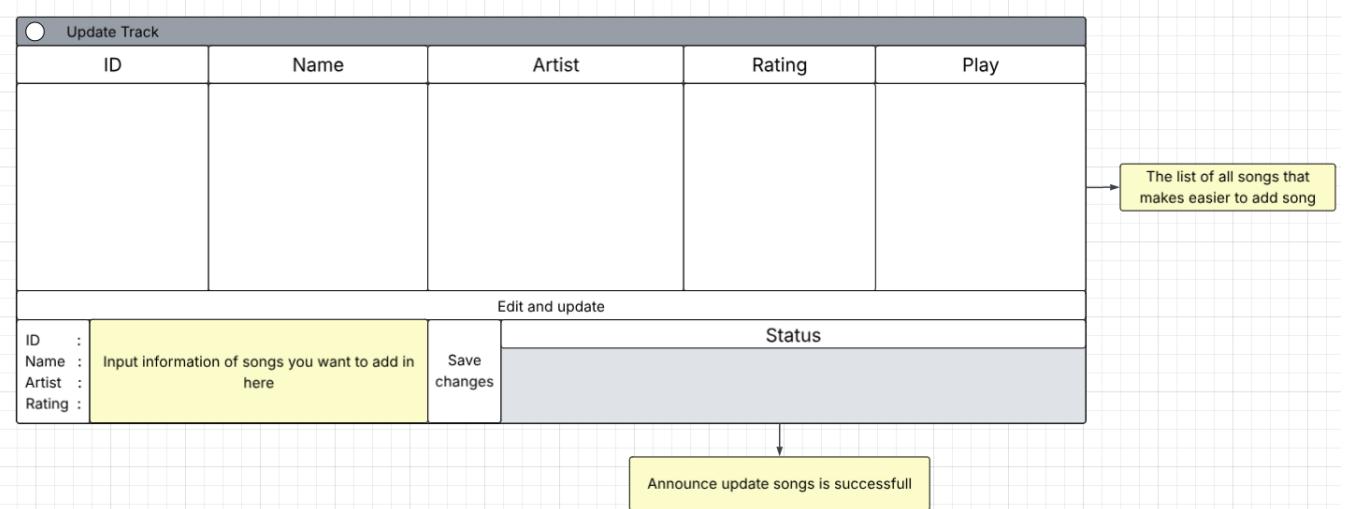
2. View Track



3. Track List



4. Update Track



d. Data Management

File Structures:

1. music.csv:

```
1. name,artist,rating
2. "Another Brick in the Wall","Pink Floyd",4
3. "Stayin' Alive","Bee Gees",5
4.
```

2. Playlist files (list1.csv):

```
1. 01,1
2. 03,0
```

Implementation Details:

Track Library: track_library.py loads/saves from music.csv

```
1. def load_library(filename="music.csv"):
2.     with open(filename, "r") as file:
3.         reader = csv.reader(file)
4.         next(reader) # Skip header
5.         for index, row in enumerate(reader, start=1):
6.             library[f"{index:02d}"] = LibraryItem(row[0], row[1], int(row[2]))
7.
```

3. Development Stages

a. Stage 1: Basic Understanding

The Jukebox application is a Python program with a GUI built using tkinter. It allows users to view local tracks, manage playlists, and update song metadata. All track data is stored in a CSV file.

The app is divided into three main features:

- **View Tracks:** Displays all songs in a table format.
- **Playlists:** Users can add or remove tracks to create custom playlists.
- **Update Library:** Users can edit song title, artist, and number of plays.

Each feature is implemented in a separate Python file for modularity. The GUI includes a top navigation bar to switch between tabs. The system does not include online search or music playback, focusing only on local data management.

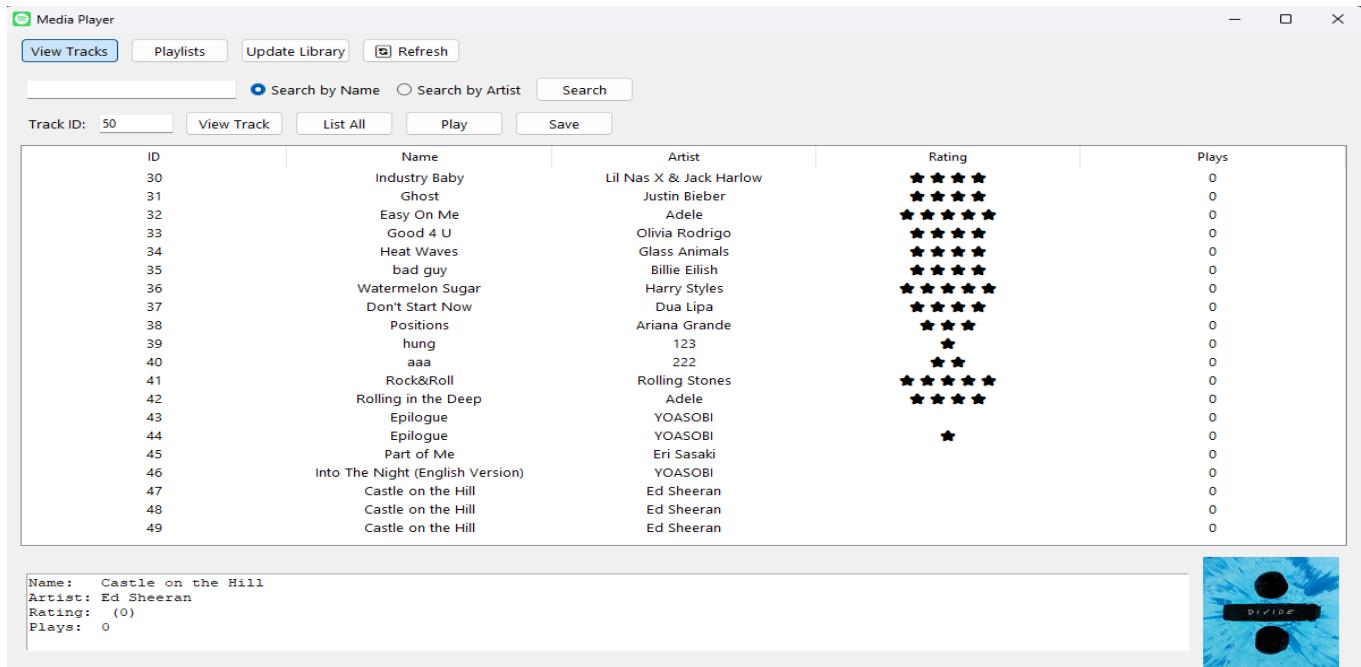
b. Stage 2: Outline Implementation

1. Overview

The Jukebox application was developed using Python and the tkinter library for the graphical user interface. The application is focused on managing track data and playlists. The two core functionalities remaining are:

- **View Tracks:** Allows users to browse all available songs stored locally.

Figure 1: The "View Tracks" tab showing a table of all available songs stored locally.



- **Playlists:** Users can build and save custom playlists from the available tracks.

Figure 2: The "Playlists" tab where users can add or remove tracks in playlist.

All Tracks					Selected Tracks					
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play
01	Smells Like Teen	Nirvana	★★★★★	0	01	Smells Like Teen	Nirvana	★★★★★	0	0
02	Uptown Funk	Bruno Mars	★★★★★	0	04	Viva La Vida	Coldplay	★★★★★	0	0
03	Shake It Off	Taylor Swift	★★★★★	0	03	Shake It Off	Taylor Swift	★★★★★	0	0
04	Viva La Vida	Coldplay	★★★★★	0	02	Uptown Funk	Bruno Mars	★★★★★	0	0
05	Counting Stars	OneRepublic	★★★★★	0	05	Counting Stars	OneRepublic	★★★★★	0	0
06	Radioactive	Imagine Dragon:	★★★★★	0	25	Peaches	Justin Bieber	★★★★★	0	0
07	Chandelier	Sia	★★★★★	0	27	An	Taylor Swift	★★★★★	0	0
08	Take On Me	a-ha	★★★★★	0	06	Radioactive	Imagine Dragon:	★★★★★	0	0
09	Bad Guy	Billie Eilish	★★★★★	0	11	Wonderwall	Oasis	★★★★★	0	0
10	Can't Stop	Red Hot Chili Pej	★★★★★	0	13	Feel It Still	Portugal. The Ma	★★★★★	0	0
11	Wonderwall	Oasis	★★★★★	0	24	As It Was	Harry Styles	★★★★★	0	0
12	Royals	Lorde	★★★★★	0	47	Castle on the Hil	Ed Sheeran	★★★★★	0	0
13	Feel It Still	Portugal. The Mi	★★★★★	0	14	Havana	Camila Cabello	★★★★★	0	0
14	Havana	Camila Cabello	★★★★★	0	50	Castle on the Hil	Ed Sheeran	★★★★★	0	0
15	Demons	Imagine Dragon:	★★★★★	0	36	Watermelon Sug	Harry Styles	★★★★★	0	0
16	Don't Stop Believ	Journey	★★★★★	0	29	Unholy	Sam Smith & Kin	★★★★★	0	0
17	Pompeii	Bastille	★★★★★	0	34	Heat Waves	Glass Animals	★★★★★	0	0
18	7 Years	Lukas Graham	★★★★★	0	17	Pompeii	Bastille	★★★★★	0	0
19	Có em chór	Min, ERIK	★★★★★	0	16	Don't Stop Believ	Journey	★★★★★	0	0
20	Có em chór	Min, 999999999	★★★★★	0	21	aaa	1	★★★★★	0	0
21	aaa	1	★★★★★	0	18	7 Years	Lukas Graham	★★★★★	0	0
22	Blinding Lights	The Weeknd	★★★★★	0	33	Good 4 U	Olivia Rodrigo	★★★★★	0	0
23	Levitating	Dua Lipa	★★★★★	0	39	hung	123	★	0	0
24	As It Was	Harry Styles	★★★★★	0						
25	Peaches	Justin Bieber	★★★★★	0						
26	Shivers	Ed Sheeran	★★★★★	0						

Buttons at the bottom: Play Selected, Remove Selected, Save Playlist.

- **Update Library:** Enables editing of track metadata such as title, artist, and number of plays.

Figure 3: The "Update Library" tab allowing users to edit song metadata such as title, artist, and number of plays.

Track Name:	ID	Name	Artist	Rating
	01	Smells Like Teen Spirit	Nirvana	6
	02	Uptown Funk	Bruno Mars	4
	03	Shake It Off	Taylor Swift	3
	04	Viva La Vida	Coldplay	4
	05	Counting Stars	OneRepublic	4
	06	Radioactive	Imagine Dragons	5
	07	Chandelier	Sia	4
	08	Take On Me	a-ha	5
	09	Bad Guy	Billie Eilish	3
	10	Can't Stop	Red Hot Chili Peppers	5
	11	Wonderwall	Oasis	5
	12	Royals	Lorde	4
	13	Feel It Still	Portugal. The Man	3
	14	Havana	Camila Cabello	4
	15	Demons	Imagine Dragons	4
	16	Don't Stop Believin'	Journey	5
	17	Pompeii	Bastille	3
	18	7 Years	Lukas Graham	4
	19	Có em chór	Min, ERIK	5
	20	Có em chór	Min, 999999999	2
	21	aaa	1	2
	22	Blinding Lights	The Weeknd	5
	23	Levitating	Dua Lipa	4
	24	As It Was	Harry Styles	4
	25	Peaches	Justin Bieber	3
	26	Shivers	Ed Sheeran	4
	27	An	Taylor Swift	5
	28	STAY	The Kid LAROI & Justin Bieber	4
	29	Unholy	Sam Smith & Kim Petras	3

Buttons at the top: Add Track, Save Changes, Delete Selected.

Track data is stored and managed using CSV files, ensuring persistence between sessions.

c. Stage 3: Basic Working Version

- Each feature is implemented in a dedicated Python module:
 - `view_tracks.py`: Displays all track records in a table format.
 - `create_track_list.py`: Allows adding/removing tracks from playlists.
 - `update_tracks.py`: Handles editing track metadata and updating the CSV file.
 - `jukebox.py`: The main GUI file that connects all modules and controls navigation.
- The GUI is organized into frames, with navigation handled by custom-styled buttons at the top of the window.

d. Stage 4: Testing and Validation

Extensive testing was conducted to ensure proper functionality:

- Input validation for track IDs, ratings, etc.
- Unit tests for core functionality
- Integration testing of all components

e. Stage 5: Innovations

Several innovative features were added:

1. **Enhanced Playlist System**: Save/load playlists with track selection
2. **Click to add tracks to list**: Add tracks more quickly
3. **Search Functionality**: Search tracks by name or artist

4. **Track Images:** Display album art for tracks
5. **Single GUI:** Improve ease of use and testing
6. **Allow editing by clicking in update:** Edit information of track more quickly

4. Testing and Validation

A comprehensive testing strategy was implemented:

A. Unit Testing

PyTest was used to test core functionality of the LibraryItem class:

1. Test loading data from CSV file

```

1. def test_load_library():
2.     """Test loading library from CSV"""
3.     lib.load_library("music.csv")
4.     assert len(lib.library) > 0 # Check if library is loaded
5.     assert lib.get_name("01") == "Smells Like Teen Spirit"
6.     assert lib.get_artist("01") == "Nirvana"
7.     assert lib.get_rating("01") == 5 # Rating 6 should be converted to 5
8.

```

3. Test displaying the list of tracks

```

1. def test_list_all():
2.     """Test list_all() output format"""
3.     lib.load_library("music.csv")
4.     output = lib.list_all()
5.     assert "01\tSmells Like Teen Spirit\tNirvana\t★★★★★\t0" in output
6.     assert "02\tUptown Funk\tBruno Mars\t★★★★★\t0" in output

```

4. Test information retrieval functions

```

1. def test_get_methods():
2.     """Test all getter methods"""
3.     lib.load_library("music.csv")
4.     assert lib.get_name("01") == "Smells Like Teen Spirit"
5.     assert lib.get_artist("01") == "Nirvana"
6.     assert lib.get_rating("01") == 5
7.     assert lib.get_play_count("01") == 0
8.     assert lib.get_name("99") is None # Non-existent ID
9.

```

5. Test changing track ratings

```
1. def test_set_rating():
2.     """Test updating track rating"""
3.     lib.load_library("music.csv")
4.     lib.set_rating("01", 2)
5.     assert lib.get_rating("01") == 2
6.     # Verify unchanged play count
7.     assert lib.get_play_count("01") == 0
8.
```

6. Test incrementing play count

```
1. def test_increment_play_count():
2.     """Test play count increment"""
3.     lib.load_library("music.csv")
4.     lib.increment_play_count("01")
5.     assert lib.get_play_count("01") == 1
6.     lib.increment_play_count("01")
7.     assert lib.get_play_count("01") == 2
```

7. Test handling of Non-existent track in data

```
1. def test_nonexistent_track_operations():
2.     """Test operations on non-existent tracks"""
3.     lib.load_library("music.csv")
4.     # Should not raise errors but return default values
5.     assert lib.get_name("99") is None
6.     assert lib.get_play_count("99") == -1
7.     # These should silently fail (no exception)
8.     lib.set_rating("99", 3)
9.     lib.increment_play_count("99")
10.
```

8. Test handling of missing files

```
1. def test_file_not_found():
2.     """Test handling of missing CSV file"""
3.     with pytest.raises(NotFoundError):
4.         lib.load_library("nonexistent.csv")
5.
```

9. Tests detailed track information

```
1. def test_track_information():
2.     """Test getting information of specific tracks"""
3.     lib.load_library("music.csv")
4.
5.     assert lib.get_name("01") == "Smells Like Teen Spirit"
6.     assert lib.get_artist("01") == "Nirvana"
7.     assert lib.get_rating("01") == 5
8.     assert lib.get_play_count("01") == 0
9.
```

10. Tests rating conversion

```
1. def test_track_rating_conversion():
2.     """Test rating conversion for different values"""
3.     lib.load_library("music.csv")
4.
5.     # Test rating above 5 (should be converted to 5)
6.     assert lib.get_rating("01") == 5 # Original rating is 6
7.
8.     # Test rating below 0 (should be converted to 0)
9.     assert lib.get_rating("43") == 0 # Original rating is -1
```

11. Tests handling of duplicate tracks

```
1. def test_track_duplicates():
2.     """Test handling of duplicate tracks"""
3.     lib.load_library("music.csv")
4.
5.     # Test tracks with same name but different artists
6.     assert lib.get_name("09") == "Bad Guy"
7.     assert lib.get_artist("09") == "Billie Eilish"
8.     assert lib.get_rating("09") == 3
9.
10.    assert lib.get_name("35") == "bad guy"
11.    assert lib.get_artist("35") == "Billie Eilish"
12.    assert lib.get_rating("35") == 4
13.
14.    # Test multiple tracks with same name and artist
15.    assert lib.get_name("43") == "Epilogue"
16.    assert lib.get_artist("43") == "YOASOBI"
17.    assert lib.get_rating("43") == 0
18.
19.    assert lib.get_name("44") == "Epilogue"
20.    assert lib.get_artist("44") == "YOASOBI"
21.    assert lib.get_rating("44") == 1
22.
```

12. Test the creation of a Library_Item object

```
1. def test_library_item_creation(sample_library_item):
2.     """Test LibraryItem initialization with real data"""
3.     assert sample_library_item.name == "Smells Like Teen Spirit"
4.     assert sample_library_item.artist == "Nirvana"
5.     assert sample_library_item.rating == 5 # Rating 6 should be converted to 5
6.     assert sample_library_item.play_count == 0
```

13. Test changing the rating

```
1. def test_library_item_rating_setter(sample_library_item):
2.     """Test rating setter with valid values"""
3.     sample_library_item.rating = 5
4.     assert sample_library_item.rating == 5
5.     sample_library_item.rating = 0
6.     assert sample_library_item.rating == 0
7.
```

14. Test handling of invalid ratings

```
1. def test_library_item_invalid_rating(sample_library_item):
2.     """Test rating validation"""
3.     # Test with non-integer value
4.     sample_library_item.rating = "invalid" # Should be converted to 0
5.     assert sample_library_item.rating == 0
6.
7.     # Test with out of range values
8.     sample_library_item.rating = -1 # Should be converted to 0
9.     assert sample_library_item.rating == 0
10.
11.    sample_library_item.rating = 6 # Should be converted to 5
12.    assert sample_library_item.rating == 5
13.
```

15. Test incrementing play count

```
1. def test_library_item_play_count(sample_library_item):
2.     """Test play count increment"""
3.     sample_library_item.play_count += 1
4.     assert sample_library_item.play_count == 1
```

```

5.     sample_library_item.play_count += 5
6.     assert sample_library_item.play_count == 6
7.

```

Test Result

```

collected 14 items

test_library_item.py::test_library_item_creation PASSED
test_library_item.py::test_library_item_rating_setter PASSED
test_library_item.py::test_library_item_invalid_rating PASSED
test_library_item.py::test_library_item_play_count PASSED
test_track_library.py::test_load_library PASSED
test_track_library.py::test_list_all PASSED
test_track_library.py::test_get_methods PASSED
test_track_library.py::test_set_rating PASSED
test_track_library.py::test_increment_play_count PASSED
test_track_library.py::test_nonexistent_track_operations PASSED
test_track_library.py::test_file_not_found PASSED
test_track_library.py::test_track_information PASSED
test_track_library.py::test_track_rating_conversion PASSED
test_track_library.py::test_track_duplicates PASSED

===== 14 passed in 0.06s =====

```

Test Case 1: Search by ID

ID	Description	Input	Expected Output	Actual Output	Evidence	Result
1.1	Single-digit ID	ID = "1"	Track 01 found	Not found	Evidence 1.1	FAILED
1.2	Two-digit ID	ID = "01"	Track 01 found	Track found	Evidence 1.2	PASSED
1.3	Negative ID	ID = "-01"	Error message	No message	Evidence 1.3	FAILED
1.4	ID = 0	ID = "00"	Error message	No message	Evidence 1.4	FAILED
1.5	Non-existent ID	ID = "99"	Track not found	Track not found	Evidence 1.5	PASSED

Test Case 2: Search by Name/Artist

ID	Description	Input	Expected Output	Actual Output	Evidence	Result
2.1	Uppercase name	"ROLLING"	Find "Rolling"	Found	Evidence 2.1	PASSED
2.2	Lowercase name	"rolling"	Find "Rolling"	Found	Evidence 2.2	PASSED
2.3	Name with spaces	" rolling"	Find "Rolling"	Not found	Evidence 2.3	FAILED
2.4	Name with special characters	"Rock&Roll"	Song found	Found	Evidence 2.4	PASSED
2.5	Name with accents	"Có em chò"	Song found	Found	Evidence 2.5	PASSED

Test Case 3: Playlist Management

ID	Description	Input	Expected Output	Actual Output	Evidence	Result
3.1	Duplicate playlist	Name = "list1"	Ask to overwrite	Still direct to old "list1"		FAILED
3.2	Name with space	"my list"	Valid file created	File created	Evidence 3.2	PASSED
3.3	Name with special characters	"my@list"	Valid file created	File created	Evidence 3.3	PASSED
3.4	Overly long name	"a"*100	Error message	No limit	Evidence 3.4	FAILED
3.5	Play Selected songs	Click songs to appear “ ” icon and play	Play selected	Play selected	Evidence 3.5	PASSED

			songs and count	songs and count		
3.6	Move song	Drag and drop	Position changed	Not supported		FAILED
3.7	Add the same songs to the list	Choose the song	Error message	Error message	Evidence 3.7	PASSED
3.8	Delete multiple songs in the list	Ctrl + click to choose multiple songs	Delete successfully	Delete successfully	Evidence 3.8	

Test Case 4: Update track validation

ID	Description	Input	Expected Output	Actual Output	Evidence	Result
4.1	Add duplicate song	Song already in list	Ask user, add based on choice	Added successfully	Evidence 4.1	PASSED
4.2	Remove song	Choose and Click Remove	Ask user and remove from list	Removed successfully	Evidence 4.2	PASSED
4.3	Empty name		Error message	Error message	Evidence 4.3	PASSED
4.4	Delete multiple songs	Select and delete multiple	All deleted	Deleted successfully	Evidence 4.4	PASSED
4.5	Save added-songs to csv file	Click Save changes	Save songs to csv file	Work correctly	Evidence 4.5	PASSED
4.6	Refresh all views	Click “Refresh” button	Data in all tabs updates correctly	Work correctly		PASSED

Test Case 5: Validate Rating

ID	Description	Input	Expected Output	Actual Output	Evidence	Result
5.1	Alphabetic rating	Rating = "abc"	Error message	Error message	Evidence 5.1	PASSED
5.2	Decimal rating	Rating = "3.5"	Error message	Error message	Evidence 5.2	PASSED
5.3	Negative rating	Rating = "-1"	Error message	Error message	Evidence 5.3	PASSED
5.4	Rating > 5	Rating = "6"	Error message	Error message	Evidence 5.4	PASSED
5.5	Empty rating	Rating = ""	Set default = 0	Set rating = 0	Evidence 5.5	PASSED

5. Conclusions, Further Development and Reflection

The Jukebox application successfully implements all required functionality while incorporating several innovative features. The single-window design provides a more modern user experience compared to the original multi-window approach.

Further Development

With additional time, several improvements could be made:

1. **Cloud Integration:** Sync playlists across devices
2. **Advanced Playback Features:** Equalizer, crossfade
3. **Metadata Editing:** Edit ID3 tags of audio files
4. **Social Features:** Share playlists with other users

Reflection

This project provided valuable experience in GUI development and object-oriented design in Python. Key challenges included:

- Managing the state of the music player and keeping UI elements synchronized
- Designing an intuitive interface that accommodates all functionality
- Implementing smooth audio playback with seek functionality

The most rewarding aspect was seeing the music player functionality come together, transforming the simulation into an actual working application. This project has significantly improved my understanding of event-driven programming and GUI design principles.

6. Appendix

B. Test Table and Results

Evidence 1.1: Single-digit ID

Track ID:	1	View Track	List All	Play	Clear	
ID		Name		Artist	Rating	Plays
01		Rolling in the Deep		Adele	★★★★★	0
02		Smells Like Teen Spirit		Nirvana	★★★★★	0
03		Uptown Funk		Bruno Mars	★★★★★	0
04		Perfect		jls	★★★★★	0
05		Shake It Off		Taylor Swift	★★★★	0
06		Viva La Vida		Coldplay	★★★★★	0
07		Counting Stars		OneRepublic	★★★★★	0
08		Radioactive		Imagine Dragons	★★★★★	0
09		Chandelier		Sia	★★★★★	0
10		Take On Me		a-ha	★★★★★	0

Track 1 not found.

No image

Evidence 1.2: Two-digits ID

Track ID: 01 View Track List All Play Clear

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0
02	Smells Like Teen Spirit	Nirvana	★★★★★	0
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jls	★★★★★	0
05	Shake It Off	Taylor Swift	★★★☆☆	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragons	★★★★★	0
09	Chandelier	Sia	★★★★★	0
10	Take On Me	a-ha	★★★★★	0

Name: Rolling in the Deep
 Artist: Adele
 Rating: ★★★★☆ (4)
 Plays: 0



Evidence 1.3: Negative ID

Track ID: -01 View Track List All Play Clear

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0
02	Smells Like Teen Spirit	Nirvana	★★★★★	0
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jls	★★★★★	0
05	Shake It Off	Taylor Swift	★★★☆☆	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragons	★★★★★	0
09	Chandelier	Sia	★★★★★	0
10	Take On Me	a-ha	★★★★★	0

Track -01 not found.

No image

Evidence 1.4: ID = 0

Track ID: 00 View Track List All Play Clear

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0
02	Smells Like Teen Spirit	Nirvana	★★★★★	0
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jls	★★★★★	0
05	Shake It Off	Taylor Swift	★★★☆☆	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragons	★★★★★	0
09	Chandelier	Sia	★★★★★	0
10	Take On Me	a-ha	★★★★★	0

Track 00 not found.

No image

Evidence 1.5: Non-existent ID

Track ID: 99

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0
02	Smells Like Teen Spirit	Nirvana	★★★★★	0
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jls	★★★★★	0
05	Shake It Off	Taylor Swift	★★★★	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragons	★★★★★	0
09	Chandelier	Sia	★★★★★	0
10	Take On Me	a-ha	★★★★★	0

Track 99 not found.

Evidence 2.1: Uppercase name

ROLLING Search by Name Search by Artist

Track ID:

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0

Evidence 2.2: Lowercase name

rolling Search by Name Search by Artist

Track ID:

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0

Evidence 2.3: Name with spaces

The screenshot shows a search interface for a music player. The search bar contains the text "rolling". Below the search bar are buttons for "Search by Name" (selected) and "Search by Artist". There are also buttons for "View Track", "List All", "Play", and "Clear". A "Search" button is located at the top right of the search bar area. Below these buttons is a table header with columns: ID, Name, Artist, Rating, and Plays. The main content area displays a message: "Track 99 not found." To the right of this message is a placeholder text "No image".

Evidence 2.4: Name with special characters

The screenshot shows a search interface for a music player. The search bar contains the text "Rock&Roll". Below the search bar are buttons for "Search by Name" (selected) and "Search by Artist". There are also buttons for "View Track", "List All", "Play", and "Clear". A "Search" button is located at the top right of the search bar area. Below these buttons is a table header with columns: ID, Name, Artist, Rating, and Plays. The main content area displays one track entry: ID 45, Name Rock&Roll, Artist Rolling Stones, Rating ★★★★☆ (4 stars), and Plays 0.

Evidence 2.5: Name with accents

The screenshot shows a search interface for a music player. The search bar contains the text "Có em chờ". Below the search bar are buttons for "Search by Name" (selected) and "Search by Artist". There are also buttons for "View Track", "List All", "Play", and "Clear". A "Search" button is located at the top right of the search bar area. Below these buttons is a table header with columns: ID, Name, Artist, Rating, and Plays. The main content area displays two track entries: ID 21, Name Có em chờ, Artist Min, ERIK, Rating ★★★★☆ (4 stars), and Plays 0; and ID 22, Name Có em chờ, Artist Min, 999999999, Rating ★★☆ (2 stars), and Plays 0.

Evidence 3.2: Name with space

All Tracks							Selected Tracks				
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play	
01	Rolling in the Deep	Adele	★★★★★	0	05	Shake It Off	Taylor Swift	★★★★	0		
02	Smells Like Teen	Nirvana	★★★★★	0	10	Take On Me	a-ha	★★★★★	0		
03	Uptown Funk	Bruno Mars	★★★★★	0							
04	Perfect	jls	★★★★★	0							
05	Shake It Off	Taylor Swift	★★★★	0							
06	Viva La Vida	Coldplay	★★★★	0							
07	Don't Stop Believin'	Foreigner	★★★★	0							

Evidence 3.3: Name with special characters

All Tracks							Selected Tracks						
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play			
01	Rolling in the De	Adele	★★★★★	0	05	Shake It Off	Taylor Swift	★★★★	0				
02	Smells Like Teen	Nirvana	★★★★★	0	10	Take On Me	a-ha	★★★★★	0				
03	Uptown Funk	Bruno Mars	★★★★★	0									
04	Perfect	jls	★★★★★	0									
05	Shake It Off	Taylor Swift	★★★★	0									
06	Viva La Vida	Coldplay	★★★★★	0									
07	Counting Stars	OneRepublic	★★★★★	0									
08	Radioactive	Imagine Dragon:	★★★★★	0									

Evidence 3.4: Overly long name

All Tracks					Selected Tracks					
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play
01	Rolling in the Deep	Adele	★★★★★	0						
02	Smells Like Teen	Nirvana	★★★★★	0						
03	Uptown Funk	Bruno Mars	★★★★★	0						
04	Perfect	J. S. Bach	★★★★★	0						
05	Shake It Off	Taylor Swift	★★★★★	0						
06	Viva La Vida	Coldplay	★★★★★	0						
07	Counting Stars	OneRepublic	★★★★★	0						
08	Radioactive	Imagine Dragons	★★★★★	0						
09	Chandelier	Sia	★★★★★	0						
10	Taylor Gang	Drake	★★★★★	0						

Evidence 3.5: Play selected songs

Select Playlist: list1

All Tracks				
ID	Name	Artist	Rating	Plays
01	Rolling in the De	Adele	★★★★★	21
02	Smells Like Teen	Nirvana	★★★★★	21
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jis	★★★★★	0
05	Shake It Off	Taylor Swift	★★★★	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragon:	★★★★★	0
09	Chandelier	Sia	★★★★★	21
10	Take On Me	a-ha	★★★★★	0
11	Bad Guy	Billie Eilish	★★★★	21
12	Can't Stop	Red Hot Chili Pej	★★★★★	21
13	Wonderwall	Oasis	★★★★★	21

Selected Tracks					
ID	Name	Artist	Rating	Plays	Play
01	Rolling in the De	Adele	★★★★★	21	∅
04	Perfect	jis	★★★★★	0	∅
03	Uptown Funk	Bruno Mars	★★★★★	0	∅
02	Smells Like Teen	Nirvana	★★★★★	21	∅
05	Shake It Off	Taylor Swift	★★★★	0	∅
25	Blinding Lights	The Weeknd	★★★★★	0	∅
27	As It Was	Harry Styles	★★★★★	21	∅
06	Viva La Vida	Coldplay	★★★★★	0	∅
11	Bad Guy	Billie Eilish	★★★★	21	∅
13	Wonderwall	Oasis	★★★★★	21	∅
15	Feel It Still	Portugal. The Ma	★★★★	0	∅
19	Pompeii	Bastille	★★★★	21	∅
12	Can't Stop	Red Hot Chili Pej	★★★★★	21	∅

Evidence 3.7: Add the same song to the list

Select Playlist: list1

All Tracks				
ID	Name	Artist	Rating	Plays
01	Rolling in the De	Adele	★★★★★	0
02	Smells Like Teen	Nirvana	★★★★★	0
03	Uptown Funk	Bruno Mars	★★★★★	0
04	Perfect	jis	★★★★★	0
05	Shake It Off	Taylor Swift	★★★★	0
06	Viva La Vida	Coldplay	★★★★★	0
07	Counting Stars	OneRepublic	★★★★★	0
08	Radioactive	Imagine Dragon:	★★★★★	0
09	Chandelier	Sia	★★★★★	0
10	Take On Me	a-ha	★★★★★	0
11	Bad Guy	Billie Eilish	★★★★★	0

Duplicate Track

Track 'Rolling in the Deep' by 'Adele' is already in the playlist.

OK

Selected Tracks					
ID	Name	Artist	Rating	Plays	Play
01	Rolling in the De	Adele	★★★★★	0	∅
02	Smells Like Teen	Nirvana	★★★★★	0	∅
03	Uptown Funk	Bruno Mars	★★★★★	0	∅
04	Perfect	jis	★★★★★	0	∅
05	Shake It Off	Taylor Swift	★★★★	0	∅
06	Viva La Vida	Coldplay	★★★★★	0	∅
07	Counting Stars	OneRepublic	★★★★★	0	∅
08	Radioactive	Imagine Dragon:	★★★★★	0	∅
09	Chandelier	Sia	★★★★★	0	∅
10	Take On Me	a-ha	★★★★★	0	∅
11	Bad Guy	Billie Eilish	★★★★★	0	∅
13	Wonderwall	Oasis	★★★★★	0	∅
15	Feel It Still	Portugal. The Ma	★★★★	0	∅

Evidence 3.8: Delete multiple songs in the list

Select Playlist: list1

All Tracks					Selected Tracks					
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play
01	Rolling in the De	Adele	★★★★★	0	01	Rolling in the De	Adele	★★★★★	0	0
02	Smells Like Teen	Nirvana	★★★★★	0	04	Perfect	jis	★★★★★	0	0
03	Uptown Funk	Bruno Mars	★★★★★	0	03	Uptown Funk	Bruno Mars	★★★★★	0	0
04	Perfect	jis	★★★★★	0	02	Smells Like Teen	Nirvana	★★★★★	0	0
05	Shake It Off	Taylor Swift	★★★★	0	05	Shake It Off	Taylor Swift	★★★★	0	0
06	Viva La Vida	Coldplay	★★★★★	0	25	Blinding Lights	The Weeknd	★★★★★	0	0
07	Counting Stars	OneRepublic	★★★★★	0	27	As It Was	Harry Styles	★★★★★	0	0
08	Radioactive	Imagine Dragon:	★★★★★	0	06	Viva La Vida	Coldplay	★★★★★	0	0
09	Chandelier	Sia	★★★★★	0	11	Bad Guy	Billie Eilish	★★★★	0	0
10	Take On Me	a-ha	★★★★★	0	13	Wonderwall	Oasis	★★★★★	0	0
11	Bad Guy	Billie Eilish	★★★★	0	15	Feel It Still	Portugal. The Ma	★★★★	0	0
12	Can't Stop	Red Hot Chili Pe	★★★★★	0	24	Hahaha	hihi	★★★	0	0
13	Wonderwall	Oasis	★★★★★	0	09	Chandelier	Sia	★★★★★	0	0
14	Royals	Lorde	★★★★★	0	17	Demons	Imagine Dragon:	★★★★★	0	0
15	Feel It Still	Portugal. The Ma	★★★★	0	35	Ghost	Justin Bieber	★★★★★	0	0
16	Havana	Camila Cabello	★★★★★	0						
17	Demons	Imagine Dragon:	★★★★★	0						
18	Don't Stop Believ	Journey	★★★★★	0						
19	Pompeii	Bastille	★★★★	0						
20	7 Years	Lukas Graham	★★★★★	0						
21	Có em chór	Min, ERIK	★★★★★	0						
22	Có em chór	Min, 999999999	★★★	0						
23	aaa	1	★★★	0						
24	Hahaha	hihi	★★★	0						
25	Blinding Lights	The Weeknd	★★★★★	0						
26	Levitating	Dua Lipa	★★★★★	0						

Buttons: ▶ Play Selected | ━ Remove Selected

Select Playlist: list1

All Tracks					Selected Tracks					
ID	Name	Artist	Rating	Plays	ID	Name	Artist	Rating	Plays	Play
01	Rolling in the De	Adele	★★★★★	0	01	Rolling in the De	Adele	★★★★★	0	0
02	Smells Like Teen	Nirvana	★★★★★	0	04	Perfect	jis	★★★★★	0	0
03	Uptown Funk	Bruno Mars	★★★★★	0	03	Uptown Funk	Bruno Mars	★★★★★	0	0
04	Perfect	jis	★★★★★	0	02	Smells Like Teen	Nirvana	★★★★★	0	0
05	Shake It Off	Taylor Swift	★★★★	0	05	Shake It Off	Taylor Swift	★★★★	0	0
06	Viva La Vida	Coldplay	★★★★★	0	25	Blinding Lights	The Weeknd	★★★★★	0	0
07	Counting Stars	OneRepublic	★★★★★	0	27	As It Was	Harry Styles	★★★★★	0	0
08	Radioactive	Imagine Dragon:	★★★★★	0	06	Viva La Vida	Coldplay	★★★★★	0	0
09	Chandelier	Sia	★★★★★	0	11	Bad Guy	Billie Eilish	★★★★	0	0
10	Take On Me	a-ha	★★★★★	0	13	Wonderwall	Oasis	★★★★★	0	0
11	Bad Guy	Billie Eilish	★★★★	0						
12	Can't Stop	Red Hot Chili Pe	★★★★★	0						
13	Wonderwall	Oasis	★★★★★	0						
14	Rovals	Lorde	★★★★★	0						

Evidence 4.1: Add duplicate song

Track Name: Rolling in the Deep Artist: Adele Rating: 2 + Add Track Save Changes Delete Selected

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02		Nirvana	5
03		Bruno Mars	4
04		jis	5
05		Taylor Swift	3
06		Coldplay	4
07		OneRepublic	4
08		Imagine Dragons	5
09		Sia	4

Duplicate

The track 'Rolling in the Deep' by 'Adele' already exists. Do you still want to add it?

Yes No

Rolling in the Deep Search by Name Search by Artist

Track ID:

ID	Name	Artist	Rating	Plays
01	Rolling in the Deep	Adele	★★★★★	0
46	Rolling in the Deep	Adele	★★★★★	0

Evidence 4.2: Remove song

Track Name: Artist: Rating:

ID	Name	Artist	Rating
02	Smells Like Teen Spirit	Nirvana	5
03	Uptown Funk	Bruno Mars	4
04		jls	5
05		Taylor Swift	3
06		Coldplay	4
07		OneRepublic	4
08		Imagine Dragons	5
09		Sia	4
10		a-ha	5
11		Billie Eilish	3
12	Can't Stop	Red Hot Chili Peppers	5

Confirm Delete

Are you sure you want to delete 1 track(s)?

Yes No

Track Name: Artist: Rating:

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02	Smells Like Teen Spirit	Nirvana	5
03	Perfect	jls	5
04	Shake It Off	Taylor Swift	3

Evidence 4.3: Empty name

Track Name: Artist: Rating:

ID	Name	Artist	Rating
01		Adele	4
02		Nirvana	5
03		Bruno Mars	4
04		jls	5
05		Taylor Swift	3
06		Coldplay	4
07		OneRepublic	4

Input Error

Track name and artist cannot be empty.

OK

Evidence 4.4: Delete multiple songs

The screenshot shows a music library interface with a table of tracks. A selection of five tracks (IDs 01, 02, 03, 04, 05) is highlighted in blue. A confirmation dialog box titled "Confirm Delete" is displayed in the center, asking "Are you sure you want to delete 4 track(s)?".

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02	Smells Like Teen Spirit	Nirvana	5
03	Uptown Funk	Bruno Mars	4
04	Perfect	jis	5
05	Shake It Off	Taylor Swift	3
06		Coldplay	4
07		OneRepublic	4
08		Imagine Dragons	5
09		Sia	4
10		a-ha	5
11		Billie Eilish	3
12		Red Hot Chili Peppers	5
13		Oasis	5

The screenshot shows the same music library interface after the deletion. A success dialog box is displayed, stating "Successfully deleted 4 track(s)".

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02	Viva La Vida	Coldplay	4
03	Counting Stars	OneRepublic	4
04	Radioactive	Imagine Dragons	5
05	Chandelier	Sia	4
06		a-ha	5
07		Billie Eilish	3
08		Red Hot Chili Peppers	5
09		Oasis	5
10		Lorde	4
11		Portugal. The Man	3
12		Camila Cabello	4
13		Imagine Dragons	4

Evidence 5.1: Alphabetic rating

The screenshot shows a music library interface with a table of tracks. In the "Rating" column for track ID 01, the value "abc" is entered. An input error dialog box is displayed, stating "Rating must be a number between 0 and 5.".

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02	Smells Like Teen Spirit	Nirvana	5
03	Uptown Funk	Bruno Mars	4
04	Perfect	jis	5
05	Shake It Off	Taylor Swift	3
06	Viva La Vida	Coldplay	4
07	Counting Stars	OneRepublic	4

Evidence 5.2: Decimal rating

The screenshot shows a music library interface with a table of tracks. In the "Rating" column for track ID 01, the value "3.5" is entered. An input error dialog box is displayed, stating "Rating must be a number between 0 and 5.".

ID	Name	Artist	Rating
01	Rolling in the Deep	Adele	4
02	Smells Like Teen Spirit	Nirvana	5
03	Uptown Funk	Bruno Mars	4
04	Perfect	jis	5
05	Shake It Off	Taylor Swift	3
06	Viva La Vida	Coldplay	4
07	Counting Stars	OneRepublic	4
no	Radioactive	Imagine Dragons	4

Evidence 5.3: Negative Rating

Track Name: 123		Artist: 123	Rating: -1	+ Add Track	Save Changes	Delete Selected
ID	Name					Rating
01	Rolling in the Deep					4
02	Smells Like Teen Spirit					5
03	Uptown Funk					4
04	Perfect					5
05	Shake It Off					3
06	Viva La Vida					4
07	Counting Stars					4

Evidence 5.4: Rating > 5

Track Name: 123		Artist: 123	Rating: -1	+ Add Track	Save Changes	Delete Selected
ID	Name					Rating
01	Rolling in the Deep					4
02	Smells Like Teen Spirit					5
03	Uptown Funk					4
04	Perfect					5
05	Shake It Off					3
06	Viva La Vida					4
07	Counting Stars					4

Evidence 5.5: Empty rating

Track Name: 123		Artist: 123	Rating:	+ Add Track	Save Changes	Delete Selected
ID	Name	Artist	Rating			
49	Part of Me	Eri Sasaki	0			
50	Into The Night (English Version)	YOASOBI	0			
51	123	123	1			
52	123	123	0			