# Design Overview for <<HD GUI MUSIC PLAYER>>

Name: Michael Haryanto

Student ID: 103841613

# Summary of Program

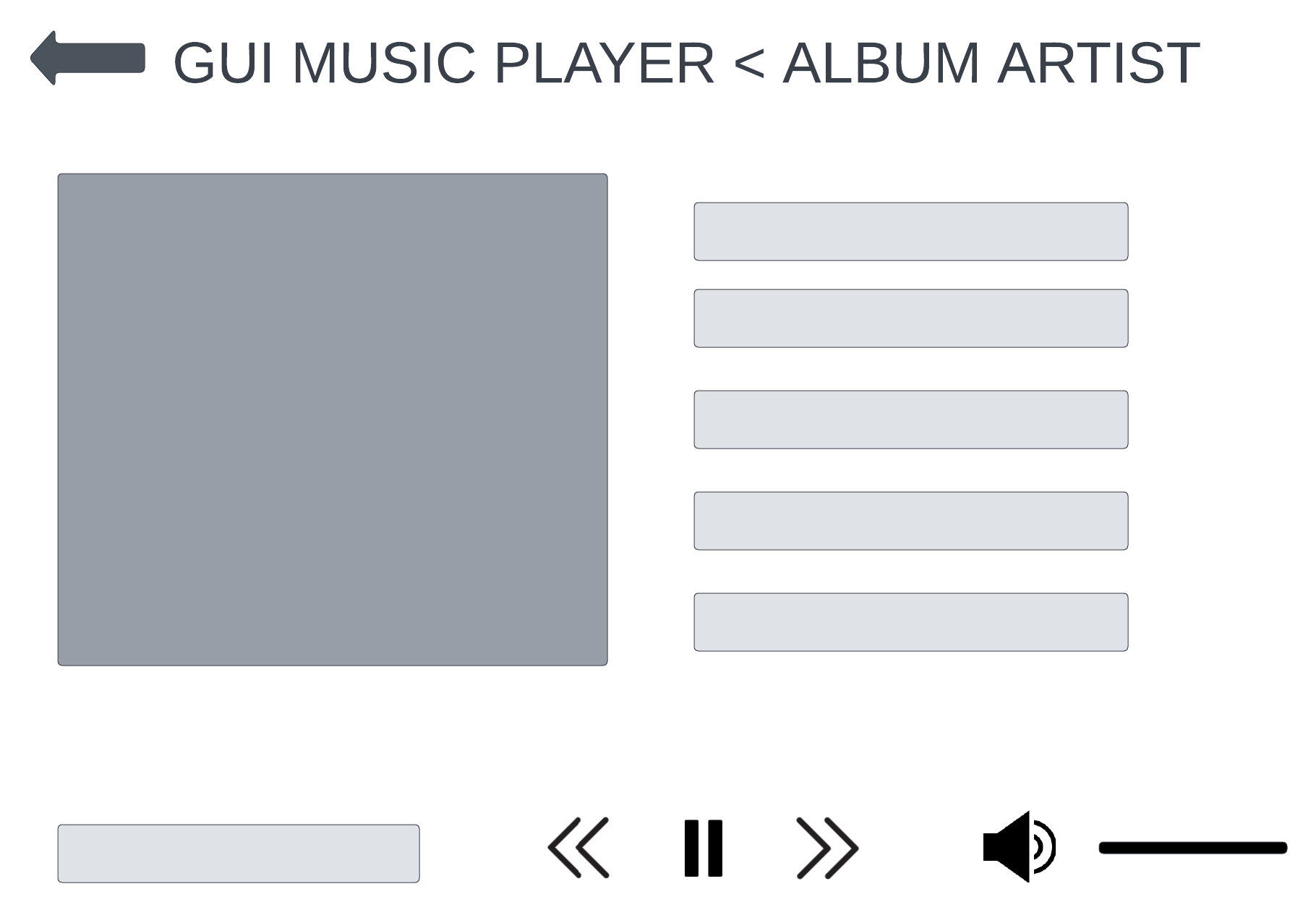
The desired “HD GUI MUSIC PLAYER” is meant to be an extended version of 7.3D task, which will fulfil the criteria of Task 9.2 and 9.3. Whereas the 7.3D task can perform a simple multiple album music player, the extended version will provide several additional features that similarly resembles several well-known music players.

The program is ought to be able to pause and play the playing song, skips a song and moves back a song. Not only that, the program should be able to set the song volume, on the scale of 0.0 to 1.0. Furthermore, the program is desired to be able to be explored in different pages, with an additional feature to add or delete several playlists made by the user.

Idea Sketches:



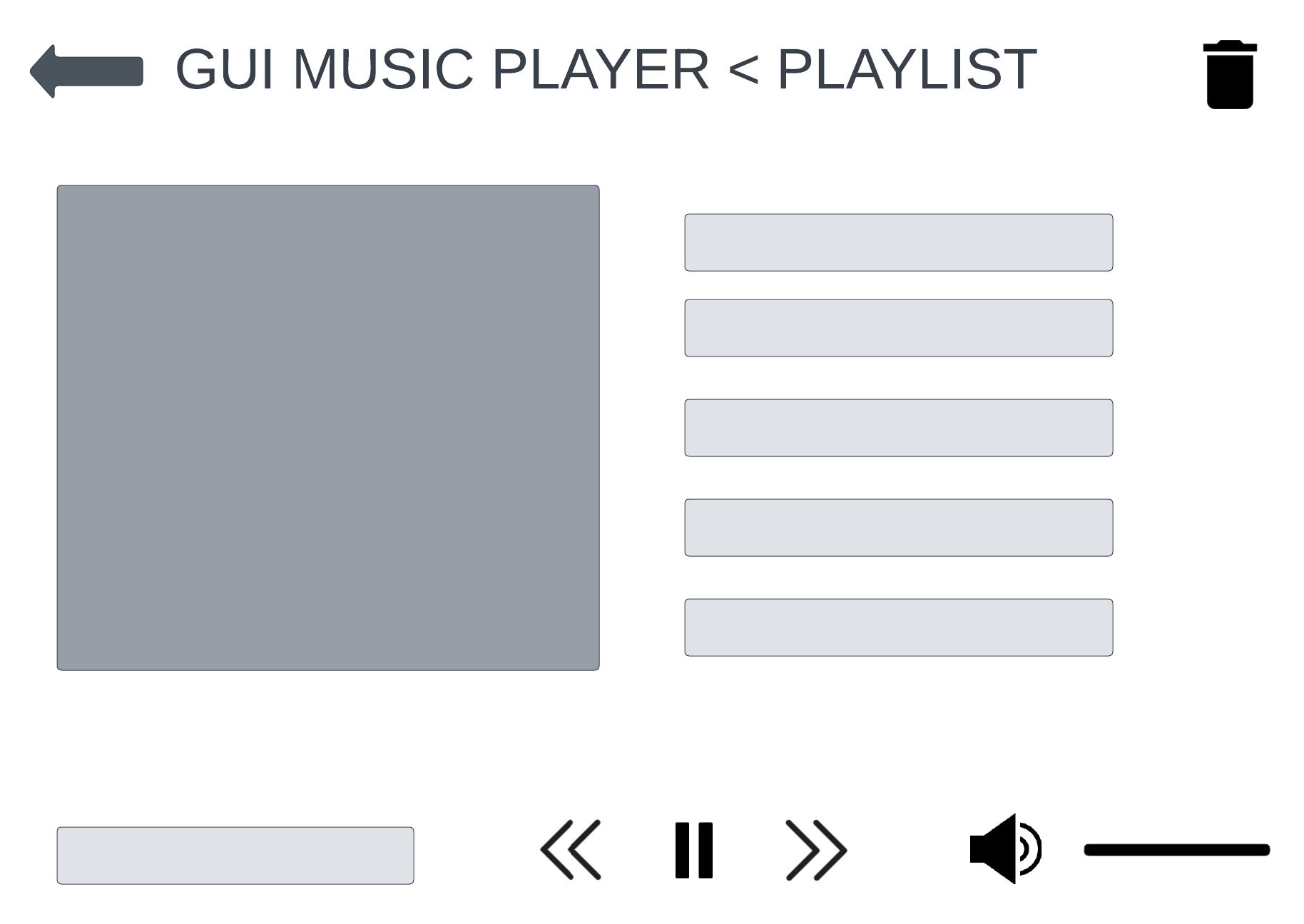
*Image 1 : Idea Sketch 1*



*Image 2 : Idea Sketch 2*



*Image 3 : Idea Sketch 3*



*Image 4 : Idea Sketch 4*

# Required Data Types

Describe each of the records and enumerations you will create using the following table (one per record).

Table : <<Album>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| title | String | The name of album’s title |
| artwork | String | The file location of the album’s artwork |
| artist | String | The name of album’s artist |
| tracks | Array | Contains an array of records of the tracks details ( name, location, position ) of an album |
| pos | Record | Contains a record of the album artwork’s position |

Table 2: <<Track>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| name | String | The name of the track’s name |
| location | String | The location of the track’s mp3 |
| box | Record | Contains a record of each track’s box position |

Table 3: <<Box>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| leftX | Integer | The coordinate of track’s left X position |
| topY | Integer | The coordinate of track’s top Y position |
| rightX | Integer | The coordinate of track’s right X position |
| bottomY | Integer | The coordinate of track’s bottom Y position |

Table 4: <<Artworkpos>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| leftX | Integer | The coordinate of album’s artwork left X position |
| topY | Integer | The coordinate of album’s artwork top Y position |
| width | Integer | The width of album’s artwork |

Table 5: <<Playlist>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| title | String | The name of playlist’s title |
| artwork | String | The location of the playlist’s artwork |
| tracks | Array | Contains an array of records of the tracks details ( name, location, position ) of a playlist |
| pos | Record | Contains a record of the playlist artwork’s position |

Table 6: <<Plisttrack>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| name | String | The name of the playlist track’s name |
| location | String | The location of the playlist track’s mp3 |
| box | Record | Contains a record of each track’s box position |

Table 7: <<Addplaylist>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| title | String | The name of the track’s title that is meant to be added to a playlist |
| location | String | The string of the track’s location that is meant to be added to a playlist |

Table 7: <<Artwork>> details

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Notes |
| bmp | String | The string of inputted artwork’s file location |

Table : << ZOrder>> details

|  |  |
| --- | --- |
| Value | Notes |
| 0 | ZOrder::BACKGROUND represents value of 0 in zpos as the layer position of the components in the program |
| 1 | ZOrder::PLAYER represents value of 1 in zpos as the layer position of the components in the program |
| 2 | ZOrder::UI represents value of 2 in zpos as the layer position of the components in the program |

# Overview of Program Structure

List of functions/procedures:

1. Load\_albums

The load\_albums function contains several functions to read the “albums.txt” file that contains all of the albums’ details. If the function were to be called, it will return an array containing all of the albums’ details.

1. Load\_playlists

Similar to load\_albums, this function is used to read the “playlists.txt” file and returns all of the playlists’ details if called.

1. Rewrite\_playlist

As the user will be able to add playlists and discard them, the function of rewriting to the “playlists.txt” file is needed. After the user has done some changes towards the playlists, it will rewrite the playlists file in order to save the changes made.

1. Initialize

The initialize procedure contains the initialize commands to run when the program first runs, such as calling load\_albums and load\_playlists, with several more basic commands such as “*super* WIDTH, HEIGHT” to initialize the window size.

1. Draw

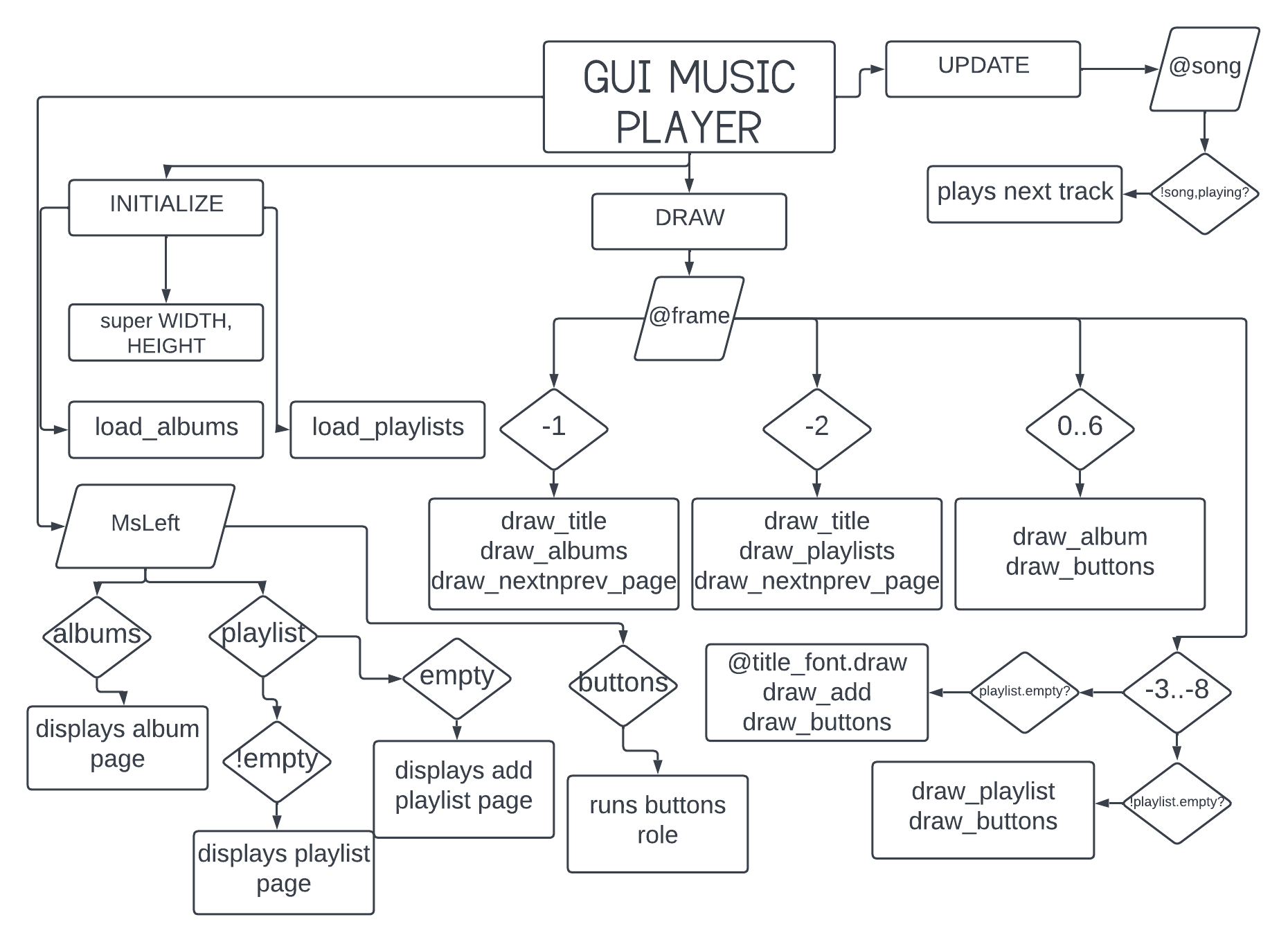
The draw procedure contains all the functions to fill the program with the components displays, such as the albums with its’ artworks, titles of each tracks in an album, playlists, etc.

1. Update

The update procedure contains several functions to keep the program running with several updates on user’s interface. As an example, the track will keep on going if it’s done playing a track until the end of album/playlist. Furthermore, the volume of the song is scaled in the update procedure

1. Button\_down

Button\_down contains all the functions to make the program interactive, which is by mouse clicks done by the user. Main example that’s inside the button\_down function is the clicking of an album in the main page will change the page into a single album’s playing track page, next page of the main page that leads to playlist page, the left click on tracks that allows user to play a specific track, button’s clicks to initialize some button’s functions, etc.



*Image 5 : Program Flowchart*