

Table of Contents

[**Introduction 1**](#_gjdgxs)

[**Description Model 1**](#_30j0zll)

[**Class Diagram 1**](#_1fob9te)

[**Use Case Diagram 1**](#_3znysh7)

[**Use Case Scenarios 1**](#_2et92p0)

[**System Sequence Charts 1**](#_tyjcwt)

# **Introduction**

*Describe the purpose of this requirements document and outline what it contains.*

# **Description Model**

*Using text, describe the requirements for your system. Expand on the function section from your project plan. Include requirements for the following categories: Output, Input, Processes, Performance and Security.*

* **Music Streaming**
  + **Output**: music audio
  + **Input**: User selects the song/album/artist/playlist they want to listen to. User can play, pause, fast forward, or rewind songs
  + **Processes**:
    - User plays music
      * User clicks play on music
      * System gets music selection from database
      * System plays music
    - User pauses music
      * User pauses music
      * System saves timestamp of song
      * System stops playing music
    - User rewinds music with button
      * User holds down rewind button
      * System rewinds the song at a 1-sec rate continuously until the User lets go of the rewind button
      * System continues playing song at new timestamp
    - User fast forwards music with button
      * User holds the fast forward button
      * System progresses song twice as fast
      * User releases fast forward button
      * System plays song at normal rate
    - User plays previous song
      * User hits previous song button
      * System plays previously played song
    - User skips a song
      * User hits the skip song button
      * System stops playing the song (If it already is) and moves on to the next song
    - User slides song duration bar
      * User slides song duration bar
      * System stops playing song
      * User releases song duration bar
      * System gets new timestamp
      * System plays song from new timestamp
  + **Performance**: N/A
  + **Security**: Users are unable to directly access the music database
* **Information on Artists/Bands**
  + **Output**: Display information on currently playing music including
    - genre
    - description of music
    - where the artist/band originated from
    - story of the band/artist
    - number of listeners and followers
    - reviews from other users
  + **Input**: Lyrical Loom will know what music is playing already; user will interact with the song to access its information
  + **Processes**:
    - User opens music information
      * User scrolls down to music information section
      * System gets music information from database
      * System displays music information
    - User closes music information
      * User scrolls up out of the music information section
      * System stops displaying information
  + **Performance**: N/A
  + **Security**: All the information will be stored on our server so there is no data to be concerned about.
* **Personal Profiles**
  + **Output**: Display the user’s profile including their
    - name, picture, and bio
    - favorite songs, artists, and albums they liked/favorited
    - posted reviews
  + **Input**: The user will choose a username, can upload a profile picture, and enter in their biography when creating their profile. Favorites can be added/removed at any time, and reviews can be posted, edited, and deleted at any time.
  + **Processes**:
    - Adding information
      * User selects information to add
      * User inputs information
      * System saves information
      * System displays information
    - Setting profile as private or public
      * User selects private or public
      * System saves choice
      * System displays information if public, and hides information if private
    - Adding reviews
      * User selects add review
      * User inputs review
      * User selects done
      * System saves review
      * System posts review
    - Editing reviews
      * User selects review to edit
      * User edits review
      * System saves edits
      * System updates review
    - Deleting reviews
      * User selects review to delete
      * System confirms deletion
      * System removes review
    - Adding favorites
      * User clicks favorite on a song, album, artist, or playlist
      * System saves selection to the favorites section
    - Removing favorites
      * User clicks unfavorite on a song, album, artist, or playlist
      * System removes selection from the favorites section
  + **Performance**: N/A
  + **Security**: The user can decide if they want their profile to be private or public. Private profiles won’t show any information about the user, public profiles will allow the user to choose what to display. Any contact, billing, or personal information will not be shown to any other users.
* **Social Activities**
  + **Output**: Display a friends list, blocked user list, and other user profiles. Also plays music if the user is in a listening party.
  + **Input:** The user can add/remove friends and search for other users. The user can also create or join a listening party.
  + **Processes**:
    - User searches for other user
      * User types in a username of another user
      * System will display search results of other users correlated with that username
      * User will select the user of choice
      * System will display their personal profile information
    - User sends friend request
      * User sends friend request
      * System notifies other user of friend request
    - User accepts friend request
      * User accepts friend request
      * System adds users to each other’s friends list
      * System notifies users of new friend
    - User declines friend request
      * System notifies other user of the denied request
      * User has the option to add the user again
    - User opens friends list
      * User opens their friend’s list to view all their added friends
      * User has the option to remove any friends from the list
      * User has the option to star friends and label them as their favorites
    - User blocks another user
      * User blocks another user
      * System adds user to blocked user list
      * System prevents blocked user from contacting or adding other user
  + **Performance:** N/A
  + **Security**: Users can only see private profiles if they are friends and friend requests need to be accepted from both sides. Anyone can see a public profile, but the user chooses what is shown (explained above).
* **Custom Playlists**
  + **Output**: Display a collection of playlists based on the user’s mood input or songs the user recently listened to. When the user opens a playlist; display the songs within it.
  + **Input**: The user can enter the mood of the playlist they want and Lyrical Loom will create a playlist based on that mood. The user then selects a playlist out of the options.
  + **Processes**:
    - Create playlist based on mood
      * User selects a mood
      * System gathers songs related to that mood
      * System creates multiple playlists from gathered songs
    - Create custom playlist
      * User selects create playlist
      * System asks user for playlist name, and offers a default name
      * User enters playlist name
      * System creates an empty playlist
      * System saves empty playlist to library
    - Choose pre-made playlist
      * User clicks on pre-made playlist
      * User clicks save playlist
      * System saves playlist to library
    - Add songs to playlist
      * User clicks add to playlist
      * User selects playlist to add song/album to
      * System updates playlist
    - Remove songs from playlist
      * User selects remove song from playlist
      * System confirms removal
      * System removes updates playlist
  + **Performance**: N/A
  + **Security**: Users have the option to make their playlists public or private. If it is private, their playlists are kept in their profile exclusive to the User. If it is public, the playlist will be shared to all of their added friends and will pose as a recommended playlist for their friends to listen to

# **Class Diagram - T.S.**

<https://github.com/aryansingh-ccm/BinaryBandits/blob/5793bfe77cb80fac221386885a9a2eb1ebc2c4fd/SystemRequirements/LyricalLoomClassDiagram.vsdx>

# **Use Case Diagram**

<https://github.com/aryansingh-ccm/BinaryBandits/blob/5793bfe77cb80fac221386885a9a2eb1ebc2c4fd/SystemRequirements/LyricalLoomUseCaseDiagram.vsdx>

# **Use Case Scenarios**

<https://github.com/aryansingh-ccm/BinaryBandits/blob/5793bfe77cb80fac221386885a9a2eb1ebc2c4fd/SystemRequirements/LyricalLoomUseCaseDescriptions.xlsx>

# **System Sequence Charts - H.M.**

[*https://github.com/aryansingh-ccm/BinaryBandits/blob/5793bfe77cb80fac221386885a9a2eb1ebc2c4fd/SystemRequirements/LyricalLoomSSD.vsdx*](https://github.com/aryansingh-ccm/BinaryBandits/blob/5793bfe77cb80fac221386885a9a2eb1ebc2c4fd/SystemRequirements/LyricalLoomSSD.vsdx)