# User

**getUserName(): String**

Return userName

**getReviews(): Review[]**

Return list of reviews

**getFriendsList(): User[]**

Return list of users

**getPrivacy(): boolean**

Return private boolean

**setPrivacy(boolean privacy): void**

Set private boolean to privacy parameter

**sendFriendRequest(User user): void**

Search for user

If found, send user a friend request

Else, notify sender user was not found

**acceptFriendRequest(): void**

Add user to friends list

Add user that accepted to senders’ friends list

**getBlockedList(): User[]**

Return list of blocked users

**unblockUser(User user): void**

Remove user from blocked list

**blockUser(User user): void**

Return add user to blocked list

**searchUser(String userName): User**

Search database for user with userName

If found, return the user

Else, return null

**createListeningParty(): void**

Set private boolean in MusicStream to false

**joinListeningParty(User user): void**

Check if user’s MusicStream is private

If not private, join user’s MusicStream

Else, notify requesting user that user’s MusicStream is private

# CustomPlaylist

**getNumSongs(): int**

Return number of songs in CustomPlaylist

**getSongs(): Song[]**

Return list of songs in CustomPlaylist

**getPlaylistName(): String**

Return playlistName

**createDefaultPlaylists(): void**

Create playlists based off what user listens to

**createPlaylist(String name): void**

Create playlist using with name as playlistName

**removeSong(Song song): void**

Check if song is in playlist

If song found, remove song from CustomPlaylist

Else, notify user song is not in playlist

**addSong(Song song): void**

Add song to CustomPlaylist

**deletePlaylist(CustomPlaylist playlist): void**

Confirm that user wants to delete playlist

If user confirms, delete playlist

Else, do nothing

**MoodPlaylist**

**createMoodPlaylist(String mood): void**

Creates playlist of songs that match the mood user entered

**getMood(): String**

Return mood of playlist

**setMood(String mood): void**

Sets mood of playlist to String mood

# Song

**getMusic(): Music**

Return music from database

**getDuration(): int**

Return duration (in seconds) of song

**getArtist(): Artist**

Return Artist of song

**getSongName(): String**

Return songName

# Artist

**getArtistName(): String**

Return artistName

**getGenre(): String**

Return artistGenre

**getMusicDescription(): String**

Return musicDescription

**getArtistOrigin(): String**

Return artistOrigin

**getArtistStory(): String**

Return artistStory

**getListeners(): int**

Return number of listeneres

**getFollowers(): int**

Return number of followers

**getReviews(): Review[]**

Return list of reviews on artist

**createArtist(): void**

Set artist name

Set artist genre

Set musicDescription

Set artistOrigin

Set artistStory

**deleteArtist(Artist artist): void**

Confirm deletion of artist

If confirmed, delete artist

Else, do nothing

**setArtistName(String name): void**

Set artistName to name

**setGenre(String genre): void**

Set artistGenre to genre

**setMusicDescription(String description): void**

Set musicDescription to description

**setArtistDescription(String description): void**

Set artistDescription to description

**setArtistOrigin(String origin): void**

Set artistOrigin to origin

**setArtistStory(String story): void**

Set artistStory to story

**setListeners(int listeners): void**

Set artist listeners to listeners

**setFollowers(int followers): void**

Set artist followers to followers

**addReview(Review review): void**

Add review to artist reviews

**removeReview(Review review): void**

Remove review to artist review

**deleteArtist(Artist artist): void**

Confirm that user wants to delete account

If confirmed, user is deleted

Else do nothing

# MusicStream

**play(Song chosenSong): void**

If chosenSong is null, set chosenSong to currentSong

Get song audio from music database

Play chosenSong

**pause(): void**

Save timeStamp

Pause song

**rewind(): void**

Play reverse 2X speed

**fastForward(): void**

Play song 2X speed

**previousSong(): void**

Play previous song

**skipSong(): void**

Play next song

**slide(): void**

Set timeStamp to location of slider

Play song after slider is released

**getCurrentSong(): Song**

Return currentSong

**getTimeStamp(): int**

Return timestamp

**getSongHistory(): Song[]**

Return songHistory

**chooseRandomSong(): Song**

Start rand song

**startStream(): void**

Call play(chooseRandomSong)

**stopStream(): void**

Call pause()

Delete MusicStream

**getPrivacy(): boolean**

Return private boolean

**setPrivacy(boolean privacy): void**

Set private boolean