Playlist Maker SRS

# Introduction

## Purpose

This document is intended to describe the user requirements for a playlist management abstract system and will outline the users desired functionality. It will try not go into the implementation details e.g. the delegation of system functionality between subsystems such as the server and desktop app.

## Scope

This is a personal hobby project designed to provide experience working with multiple integrated systems. The goal of this project is to deliver a software solution that meets all of the user requirements product using an effective, scalable and collaborative development workflow. The software developed, and technologies/workflows used are planned to be adapted for use in later projects.

This product aims to be released to the Android store for commercial profit via in-app advertisements and donations.

## Pitch

Having to find all your music files to make playlists can be a pain. With ‘Playlist Maker’, you can access your music from anywhere and make playlists where you like, when you like. ‘Playlist Maker’ finds music on all your devices and allows you to make and arrange playlists from anywhere via the cloud.

# Competitive Analysis

## Competition

Comparison of tasks performed by this product versus the competitors.

Areas to research:

* Online presence
* Multi device awareness
* Music management
* Playlist management
* Music discovery
* Music syncing to devices
* Payment method

### ITunes

* Home network music sharing (computers need to be on, streams music)
* Syncs music with devices
* Playlists management, music discovery? (search in itunes store)
* In app purchases
* Doesn’t work with android, but other apps act as an adapter

### Apple music

* Flat monthly fee
* Online streaming
* All devices
* Can upload music to cloud to listen on other devices
* Matches amazon purchases to itunes streamable songs

### ReckordBox

* Key and additional music analysis
* Can export to usb for DJ-ing with metadata associated with tracks
* Tag tracks and view tracks by tags
* Syncs playlists (versioning) from multiple devices / softwares

### WinAmp

* No website

### MusicBee

* No website

### Spotify

* Online streaming
* Music discovery
* Playlists
* All devices

(Find popular mobile variants)

## Recommendations

* Don’t focus on local desktop app, as there are multiple music managers for that.
* Don’t focus on streaming as Apple Music and Spotify fill that niche.
* Assume users will export music into a DJ tool if they DJ. Therefor this product shouldn’t have any copied DJ functionality. This product should be a lightweight first pass.

# Description

## Perspective

Users will access a website that will allow curation of playlists by showing currently available music on each associated device. Apps will be created to allow inspection of devices and syncing on playlists/music. A server will host the website and provide a backend to manage user’s metadata (playlists / available songs).

## Unique Minimal Viable Product Functions

Functions that are required for a viable product that solves the user’s issues whilst being unique in comparison to competitors. Release 1.0

* View music and playlists from any device
  + Create an account
  + Log in account
  + Associate device with account
  + Scan for music
  + View music by artist, album, title, playlist, tag
* Allow construction and modification of playlists
  + Create/Delete playlist
  + Add/Remove song from playlist
  + Add/Remove tag for track
  + Add/Remove playlist from playlist
* Allow youtube streaming playback of music
* Allow syncing of playlists for devices
  + Load playlist to device
  + Update playlist for device

## Secondary Release Functions

Functions expected to be in the second release 2.0

* Delete music from device
* Public playlists
* List of songs ‘yet to be downloaded’
* Website link to activate PC app
* Notification for devices when playlist is updated and sync in required
* Export files for playlist for/to device

## Operating Environment

* Users will be able to use this software from a website GUI. It should work identically on Internet Explorer, Firefox and Chrome on PC's and mobiles.
* Multiple users can use the software at any time.
* There will be an app for music inspection for both PC’s and Androids

## Domain Entities

* ~~User~~
* ~~MP3 metadata~~
* Account
* Track
* Tag / MP3 metadata
* Playlist
* Device

## Actors

* DJ
  + Will want to see what music they own when out and about.
  + Quickly check playlists (scan devices for DJ tool playlists)
  + Share and collaborate with other DJs to make playlists
  + Want to know if song is worth uploading into rekordbox or serato
* Casual User
  + Wants to know what music is on android phone.
  + Wants an easy way to remove music without using file explorer, or connect phone to pc
  + Uses it as a music manager
  + Stages music to be copied over to device
* Curator
  + Wants an easy way to sort music on multiple devices into playlists
  + Wants to be able to find files and upload to device to fill playlist

## List of MVP Tasks

* Sign up
* Associate device
* Un-Associate device
* Login
* Logout
* Scan music
* Set scan folders
* Look through music
* Play music via youtube
* See where track is located
* Sort music into playlist / update playlist
* Sync playlists to device
* Tag tracks

## CRUD check

### Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Account | Track | Tag | Playlist | Device |
| Sign up | CU |  |  |  |  |
| Associate device |  |  |  |  | CR |
| Un-Associate device |  |  |  |  | UD |
| Login | RU |  |  |  |  |
| Logout | R |  |  |  |  |
| Scan music | U | CD |  | C | RU |
| Set scan folders |  |  |  |  | U |
| Look through music | R | R | R | R |  |
| Play music via youtube |  | R |  |  |  |
| See where track is located | R | R |  |  |  |
| Sort music into playlist / update playlist | R | R |  | CRUD |  |
| Sync playlists to device |  |  |  | R | U |
| Tag tracks | R | RU | CRUD |  |  |

### Notes

* Ambiguity between Tag and MP3 Metadata
* User is the same as Account
* Accounts can’t be deleted
* Tracks are synced to scanned devices
* Device seems too abstract
* Might need to add ‘music folder location’

### Recommendations

* Remove user from domain model
* Merge tags and MP3 metadata

# Tasks & Supports

The following are descriptions of the supported user tasks. They don't explicitly outline an intended technical solution but provide a rough suggestion for the reader, as the aim is to outline the problem domain of the system (rather than the solution domain). Inspired from the whitepaper [Task Descriptions as Functional Requirements](http://www.itu.dk/~slauesen/Papers/IEEEtasks.pdf).

## Sign up

* **Purpose:** Register the user with the system
* **Trigger/Precondition:**
* **Frequency + Critical Freq:** Once per user /
* **Subtasks | Example Solution:**
  + Take username
  + Take password
  + Take email
  + Confirm account details
  + Notify user of successful account creation
* **Variants:**
  + Username or email is taken
  + Password or Email format is invalid

## Associate Device

* **Purpose:** Registers a device (PC or Mobile) with a user account
* **Trigger/Precondition:** User has an account
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + Generate account access code
  + Device takes account access code
  + Notify user of successful device-account association

**OR**

* + Login on device (Refer to login task)
  + Notify user of successful device-account association
* **Variants:**
  + Incorrect access code OR login details

## Un-Associate device

* **Purpose:** Remove the device access from the user account
* **Trigger/Precondition:** User has an account, associated device and is logged in on the website.
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User views associated devices
  + User selects the device to edit
  + User then unlinks the device
  + Confirms unlinking device
  + Notifies user the device was unassociated
* **Variants:**

## Login

* **Purpose:** Authenticate the user and provide access to the system
* **Trigger/Precondition:** Userhas an account and isn’t logged in
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + Collect username, password
  + Notify successful login
  + Redirect to landing page
* **Variants:**
  + Incorrect username, password

## Logout

* **Purpose:** Cancel the user’s session with the system
* **Trigger/Precondition:** User is logged in
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User signs out
  + Redirect to login page
* **Variants:**

## Scan music

* **Purpose:** Record tracks stored on user’s devices
* **Trigger/Precondition:** 
  + Current device app is open
  + Device is connected to internet
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User selects ‘scan’
  + Confirmation when scan complete
* **Variants:**
  + First time scanning, ‘Set scan folders’ dialog will popup

## Set scan folders

* **Purpose:** Record root locations for music files for scanning
* **Trigger/Precondition:** 
  + Current device app is open
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User selects ‘music location’
  + User selects ‘add new’, ‘remove’ or ‘edit’
  + Confirmation
* **Variants:**
  + Defaults to scanning all folders for mobile devices.

## Look through music

* **Purpose:** Observe tracks stored on user’s devices
* **Trigger/Precondition:** Logged into website and has a device associated that has been scanned
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User is on the landing page
  + Uses the interface to search and select music to inspect
* **Variants:**

## Play music via youtube

* **Purpose:** Listen to tracks stored on remote devices via youtube
* **Trigger/Precondition:** Logged into website and has a device associated that has been scanned
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + User is on the landing page
  + Uses the interface to search and select music to inspect
  + User selects song
  + User clicks ‘play on youtube’
  + A video embed appears and starts playing the song
* **Variants:**
  + Incorrect song is found, user can click ‘retry’

## See where track is located

* **Purpose:** Observe the stored location of a track
* **Trigger/Precondition:**
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + See ‘Look through music’
  + Location is visible in the track row
* **Variants:**
  + If track cant be found on a device (it has been removed or added as a wishlist) a warning should appear instead

## Sort music into playlist / update playlist

* **Purpose: T**racks can be stored in playlists, and playlists can be nested
* **Trigger/Precondition:**
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + See ‘Look through music’
  + User selects and drags a track onto a playlist
* **Variants:**
  + A selection can be dragged also

## Sync playlists to device

* **Purpose:** Playlists created on the website can be accessed by the device
* **Trigger/Precondition:** The device app is open and associated with an account
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + **User selects sync**
  + **Confirmation of playlists updated, created, or destroyed**
  + **User selects playlist folder**
  + **Can click on playlist to open, or folder to open playlist folder**
* **Variants:**

## Tag tracks

* **Purpose:** Tracks can be tagged, tagged tracks are visible in respective tag playlist
* **Trigger/Precondition:**
* **Frequency + Critical Freq:**
* **Subtasks | Example Solution:**
  + See ‘Look through music’
  + User selects and drags a tag onto a track
  + A tag can also be dragged onto a selection of tracks
* **Variants:**

# Domain Model

# Design & Implementation constraints / Assumptions

# Non-Functional Requirements

## Performance

## Reliability

## Safety

## Usability

## Software Quality