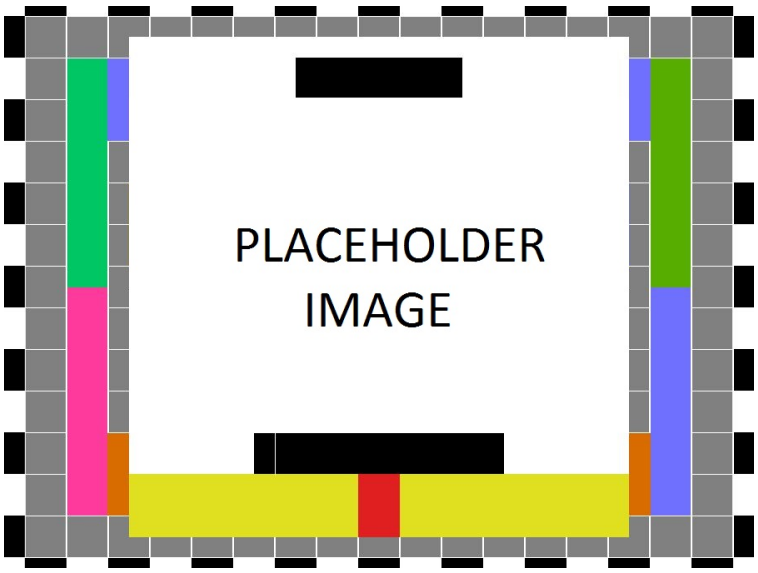
Department of Computer Science & Engineering
The University of Texas at Arlington

Project Charter
CSE 4316: Senior Design I
Fall 2018



The Under Achievers
Synthify

Dominic Young
Kolten Sturgill
Mary Huerta
Quan Nguyen
Mitchel Smith

Endy Pluviose

Revision History

Revision	Date	Author(s)	Description
0.1	10.01.2018	DY	document creation
0.2	10.05.2018	DY, MH, KS	complete draft
0.3	10.12.2018	DY, MH, KS	release candidate 1
1.0	10.23.2018	DY, MH, KS	official release

Contents

1 Vision

Some people use different services for managing and having access to music that isn't available on other platforms. Spotify playlists for one day, YouTube music playlists for another day. Maybe you even have a Soundcloud to listen to the next up & coming artist. It's tedious to manually open each platform to find music not available on the other. Wouldn't it be convenient if there was a way to grab all your playlists and music content and have them all available in one location? An aggregation of playlists, perhaps? Introducing synthify.me, a platform that will allow you to have all your music in one place."

2 Mission

The idea is to grab all the playlists of a user from each platform, and bring them all together and have the playlists unified under one service. The service will auto-update after a certain amount of time to make sure it's consistent with any playlist changes that the user may have done on the respective platform.

3 Success Criteria

Upon completion of the prototype system, we expect the following success indicators to be observed on kiosk stations implementing the new GUI software:

- Successful connection to APIs (Primarily Spotify and YouTube, SoundCloud is a bonus)
- User authentication
- Grabbing music from the authenticated user

Within 3 months after the prototype delivery date, we expect the following success indicators to be observed:

- Successful connection to all APIs
- Displaying playlists in a clear UI that allows all to be seen one in convenient location
- Choosing a song plays the song
- Utilize AJAX to check and get updated playlists after a certain amount of time

Within 6 months after the prototype delivery date, we expect the following success indicators to be observed:

- Each song has options that allow you to view the song depending on which service the original playlist (and song) is from.
- Spotify https://rpl.cat/4CBg2t-gdI6QhixqLtxqOF5aF-loJwZ-RWYtIw5xElU
- YouTube
- Dynamically create new playlists based off of mood, or audio features of a song

4 BACKGROUND

There are multiple types of music platforms available, but some platforms offer more than the other as far as content goes. Some people may be subscribed to more than one platform but desires that all of their content be in one place. Sure you could just have all of those platforms open individually and manually switch to them when you're looking to find content on one that isn't on the other, but it's not the best UX. This project solves this issue and allows you to have all your content in one place.

5 RELATED WORK

Tomahawk Player [**?**]

- Tomahawk is a free multi-source and cross-platform music player. This application can play not only your local files but also stream from services like Spotify, Beats, SoundCloud, Google Music, YouTube, and many others.
- Tomahawk is a player for music metadata. Given the name of a song and artist, Tomahawk will find the right source, for the right user at the right time.
- Tomahawk is primarily a **desktop project**, while Synthify will aim to be a **web application** so that it is available on any device that has internet.
- Tomahawk was an **enthusiast project**, but is essentially an **abandoned** project.

Soundiiz [**?**]

- Soundiiz is a playlist converter/manager for several music streaming sites.Several streaming platforms are available as Deezer, Apple Music, SoundCloud, YouTube, Qobuz, Spotify, Napster.
- Soundiiz seems to only allow you to merge playlists, and not play the content in your playlists like Synthify aims to do.
- Soundiiz is **commercially available**, but offers a free plan.

Amplifind [**?**]

- Amplifind is a music player built by music-lovers for music-lovers. Designed to be simple and fast, Amplifind gets you to the music you want to hear whenever you want to hear it.
- 'Until now, this meant you had to visit each site separately to find and listen to the music you want. Now you can access it all in one place.'
- Amplifind is a mobile application available on the Apple App Store.
- Works with local mp3's, Grooveshark, Spotify, and Soundcloud.
- Also has a 3D music visualizer built in.
- Amplifind was **commercially available**, but is essentially an **abandoned** project with no updates.

6 SYSTEM OVERVIEW

The project will be a web application (session-based), that will have connections to specific API's (Spotify, YouTube, SoundCloud). Our backend server will be used to cache previous searches/download to prevent unnecessary calls to API's.

7 Roles & Responsibilities

The stakeholder of the project will be Christopher Conly. The project does not have a sponsor, but the point of contact between the stakeholder and the team members will be Dominic Young. The team members are:

- Dominic Young
- Kolten Sturgill
- Mary Huerta
- Quan Nguyen
- Mitchel Smith
-

The product owner and scrum master will be alternate between Dominic Young and Kolten Sturgill for the entirety of the project.

8 Cost Proposal

The bulk of our costs will come from the domain name, and running a server for the web application

8.1 Preliminary Budget

Domain name, Servers costs

8.2 Current & Pending Support

CSE Department

9 Facilities & Equipment

We will be utilizing the lab space already available, along with the whiteboards to sketch out our ideas.

We will not need other equipment

10 Assumptions

- We are assuming the user has internet access
- We are assuming the user has a device that can access the product
- We are also assuming the user has accounts with the platforms we are targeting

11 Constraints

- Final prototype demonstration must be completed by May 1st, 2019
- Total development cost must not exceed $800

12 Risks

Risk description	Probability	Loss (days)	Exposure (days)
Not finishing the project	0.20	20	4
Being rate-limited/banned/blacklisted from using external API's	0.20	14	2.8
Breaking changes or depreciation of endpoints from third party API's	0.30	9	2.7

13 Documentation & Reporting

13.1 Major Documentation Deliverables

13.1.1 Project Charter

- Will be maintained under both version control in a git repository and as a word document.
- Will be updated whenever an important aspect (such as implementation, architecture, ideas, etc.) is changed.
- The initial version will be submitted when the first sprint is due (October 23rd).
- The final version will be submitted whenever the final submission of the project is.

13.1.2 System Requirements Specification

- Will be maintained under both version control in a git repository and as a word document.
- Will be updated whenever an important aspect (such as system requirements) is changed.
- The initial version will be submitted whenever it is first due.
- The final version will be submitted whenever the final submission of the project is.

13.1.3 Architectural Design Specification

- Will be maintained under both version control in a git repository and as a word document.
- Will be updated whenever an important aspect (such as the underlying architecture) is changed.
- The initial version will be submitted whenever it is first due.
- The final version will be submitted whenever the final submission of the project is.

13.1.4 Detailed Design Specification

- Will be maintained under both version control in a git repository and as a word document.
- Will be updated whenever an important aspect (such as the design system) is changed.
- The initial version will be submitted whenever it is first due.
- The final version will be submitted whenever the final submission of the project is.

13.2 Recurring Sprint Items

13.2.1 Product Backlog

- Items will be added by order of prioritization. The items will be prioritized by whether or not they are required for the core of the product to work.
- A group vote will be administered to determine how a task is prioritized. Someone in the group is allowed to take care of it sooner than its determined priority if they have the time for it.
- GitHub offers a kanban-style board for projects, so we will be utilize this to keep track of the product backlog.

13.2.2 Sprint Planning

- The team will meet up and discuss the requirements that need to be finished, and the tasks that are on the project board, and align them with the sprint intervals in the Senior Design course.
- There will be 8 sprints total. (4 in SD I, and 4 in SD II).

13.2.3 Sprint Goal

- The group leader and stakeholder will decide the sprint goal once the team has determined what needs to be accomplished in said sprint.

13.2.4 Sprint Backlog

- The team as a whole will decide what items make their way into the backlog.
- GitHub offers a kanban-style board for projects, so we will be utilize this to keep track of the product backlog.

13.2.5 Task Breakdown

- Individual tasks will be assigned based off of skill level, willingness to learn subject material, or if anyone voluntarily claims said task.
- Team members will post updates to assigned tasks using GitHub Issues & Pull Requests.

13.2.6 Sprint Burn Down Charts

- Responsibility of the burndown chart for each sprint will be decided on either a team member taking the initiative, or from whoever has yet to take responsibility for it.

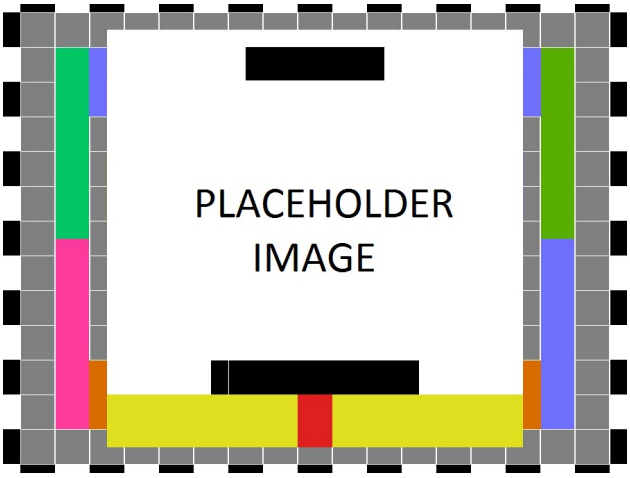


Figure 1: Example sprint burn down chart

13.2.7 Sprint Retrospective

- After the completion of each sprint, the team will meet, discuss, and take notes of the retrospective.
- Topics that will be documented include tasks completed, tasks not completed, risks, implementation issues, personal challenges, etc.
- The retrospective will be submitted shortly after the sprint is completed.

13.2.8 Individual Status Reports

- Statuses that should be reported by each individual team member includes personal issues that give risks to completing the work, challenges faced & lack of understanding in assigned tasks, new implementation ideas, and anything else they see fit to note.

13.2.9 Engineering Notebooks

- Each member will "witness" one other member's notebook during a meeting. A minimum one page will be completed per interval. Each member must be at a meeting to get their ENB witnessed.
- An individual's notebook will need to be updated every two weeks with meeting notes and ideas.

13.3 Closeout Materials

- N/A

13.3.1 System Prototype

- The final system prototype will include the website for aggregating all the music playlists. It will be demonstrated on May 1st, 2019.

13.3.2 Project Poster

- The project poster will include screenshots of the final project being developed, why we created the project and how to use the project.

13.3.3 Web Page

- The project web page will have a link to download the service on it and will explain the service. It will be accessible to the public and will be delivered by May 1st, 2019.

13.3.4 Demo Video

- The demo video will show the user how to sign in to the providers, link their streaming playlists to their current session, and choose music from each playlists from the providers that are aggregated together.

13.3.5 Source Code

- We will maintain our source code by hosting our code base on GitHub. We will adopt Git as our source control. A website will be the source of the product.

13.3.6 Source Code Documentation

- We will use React Docgen for creating documentation (https://github.com/reactjs/react-docgen). Which can be extracted as a json blob.

13.3.7 Hardware Schematics

- N/A (There will be no hardware for this project)

13.3.8 CAD files

- N/A (There will be no hardware in this project. Therefore, there will not be a need for CAD files.)

13.3.9 Installation Scripts

- A project README and installation steps will be provided in the project repo.

13.3.10 User Manual

- A project README and installation steps will be provided in the project repo.