Revision History

Date	Developer(s)	Change
October 6, 2017	Aidan/Sullivan/Karlo	Initial creation of document.
November 28, 2017	Sullivan	Updated document to reflect feedback received
December 5, 2017	Sullivan	Finalized. Fixed bolding of requirements. Didn't make that red because it would look dumb.

SE 3XA3: Software Requirements Specification Shuffle

Team 16, ASK Studios
Aidan Schonewille, schonea
Sullivan Stobo, stobos
Karlo Delalic, delalik

December 6, 2017

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
Oct 4, 2017	1.0	Got a start on filling stuff on inital section.
Date 2	1.1	Finished document.

1 Project Drivers

1.1 The Purpose of the Project

The purpose of our project is take the open source project Shuffle and redevelop it using software engineering principles and design philosophy as well as adding additional features.

1.2 The Stakeholders

1.2.1 The Client

The client is the developer of Shuffle, of which we will be basing our product on.

1.2.2 The Customers

The customer is someone who enjoys watching music videos on their computer. They would most likely be someone who is already familiar with YouTube, but it is possible that they use a different service to view their music videos. YouTube's wide variety of content allows our product to cater to any music listener, regardless of taste.

1.2.3 Other Stakeholders

Developers of YouTube or those who develop using the YouTube API have a stake in this product, as we will be heavily relying on it for the delivery of our service. Music artists and labels hold a stake in that their products will be shown to our customers. The members of ASK Studios are stakeholders as we are the primary developers and the outcome of this product will reflect on us heavily.

1.3 Mandated Constraints

1.3.1 Solution Constraints

The application will be operable on any operating system which supports JavaScript enabled web browsers. This includes Windows, Linux, Mac OSX, Android, and iOS.

Rationale: Clients use each of the above listed operating systems, some even use more than one.

Fit Criterion: The application will be approved by tests on each of the operating systems.

1.3.2 Partner or Collaborative Applications

The application is based on a preexisting YouTube Music Video Shuffler which is written in JavaScript. Therefore the user must be able to support JavaScript on their device in order for the application to be usable.

1.3.3 Off-the-Shelf Software

We will use Bootstrap 4 for easier development of the structure of the website. This will save time and the large Bootstrap community provides as a great support for a bug-free environment.

1.3.4 Budget Constraints

The operating budget is $0 and money is not required to complete any part of this project. Free web hosting and domains are available on the web.

1.3.5 Schedule Constraints

The product must be completed by the week of November 26, 2017.

1.4 Naming Conventions and Terminology

1. **API:** Application Program Interface. An Application Program Interface is a set of routines, protocols, and tools for building software applications. An API specifies how software components should interact. It is used by applications to request services from an operating system, software libraries, or other services provided on a server.

2. **JavaScript:** An object-oriented computer programming language commonly used to create interactive effects within web browsers.

3. **HTML:** HyperText Markup Language. HyperText Markup Language elements are the building blocks of HyperText Markup Language pages. They are represented by tags. Tags label pieces of content such as heading, paragraph, table, and so on. It describes the structure of web pages using markup.

4. **CSS:** Cascading Style Sheets. Cascading Style Sheets describes how HyperText Markup Language elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once.

5. **YouTube:** YouTube is a free video-hosting website that allows members to store and serve video content. YouTube members and website visitors can share YouTube videos on a variety of web platforms by using a link or by embedding HyperText Markup Language code.

6. **Bootstrap:** A free and open-source front-end web framework for designing websites and web applications. It contains HyperText Markup Language and Cascading Style Sheets based design templates for typography, forms, buttons, navigation and other interface components, as well as JavaScript extensions.

7. **UI:** User Interface. The means by which the user and a computer system interact, in particular the use of input devices and software.

8. **ReactJS:** React is a JavaScript library for building user interfaces. React allows developers to create large web-applications that use data and can change over time without reloading the page. It aims to primarily provide speed, simplicity, and scalability.

1.5 Relevant Facts and Assumptions

It is assumed that users are familiar with navigating the internet and using YouTube to view videos. It is also assumed that users will take responsibility for the bandwidth usage that they will incur if such a limit exists in their internet plan. The YouTube API works in such a way that when the program calls the API with a keyword - such as an artist, genre, or song name - a link or links relevant to the keyword are returned and then are served to the user in a viewable way. YouTube does not legally allow for the video and audio of a video to be separated so it is not legally possible for the program to provide a typical music player without displaying the visuals along with the audio.

2 Functional Requirements

Requirement: The product shall allow music videos to be streamed via YouTube.
Rationale: This is the very basic function of our product.

Requirement: When the current music video playing ends, another relevant music video shall begin playing.
Rationale: This is also a very basic function of our product and allow for minimal input use, as the user can simply let videos play.

Requirement: The product shall allow the user to input various artists or genres to provide and begin playing a random music video based on these terms.
Rationale: This allows the user to provide a starting point from which the product will begin. The program can then select music videos that are similar to the initial video selected.

Requirement: The product shall allow the user to skip a video or go back to the previous video played.
Rationale: This allow the user to revisit songs they enjoyed or skip songs that they do not enjoy. This should make our product nicer to use.

Requirement:The product shall provide a viewable playlist to allow the user to see what has previously played an what will be played in the future.
Rationale: Allows the user to view what was previously play to allow them to look up previous songs without having to replay them. It also allows them to look forward to music that will be played or skip ahead to music they would like to hear.

2.1 Scope of Product

We are taking the existing product Shuffle, recreating it and adding additional features that will improve the user's experience.

2.1.1 Context of Work

The product shall run in HTML5 supported browsers and shall require an Internet connection.

2.1.2 Work Partitioning

Karlo shall be the project leader, Sullivan shall be working mostly on front end and Aidan shall be responsible for documentation. These roles are subject to change and we will all be doing work on most aspects of the product.

2.1.3 Use Cases

User provides several artists or genre filters.
User hits a button to apply those filters.
A music video that passes all the filters begins to play.

User presses the skip button.
A new video that passes any filters applied begins to play.

The users presses the previous button.
The video that was playing before the user pressed the skip button begins to play.

3 Non-functional Requirements

3.1 Look and Feel Requirements

Requirement: The product's visual interface should be minimal and not obstruct the video being played.
Rationale: The user's focus should be on the music video displayed, and the visual interface should not distract the user from their video.

3.2 Usability and Humanity Requirements

Requirement: All of the product's features should be accessible from a single page and not interfere with video playback.
Rationale: The main feature of our product is the playback of music videos, and this service should not be interrupted unless absolutely necessary or desired by the user.

Requirement: The product should be easy to understand and operate at a first glance.
Rationale: Our product should appeal to the widest audience by being easy to use and understand.

3.3 Performance Requirements

Requirement: The product should perform similarly to Shuffle and existing YouTube-based web products.
Rationale: To make our product more desirable, it should deliver its services quickly, as to not be outclassed by similar products.

3.4 Operational and Environmental Requirements

Requirement: The product shall run on a desktop HTML5 environment.
Rationale: This is where Shuffle functions, and to fully support its userbase our product shall support it as well.

3.5 Maintainability and Support Requirements

Requirement: The product shall be simple to maintain and support over its lifespan.
Rationale: Should any of the development team or outside developers wish to improve our product, they should not be impeded from doing so. It should be built with this in mind.

3.6 Security Requirements

Requirement: The service shall not leek any of the users personal data in any way.

Rationale: This would obviously turn off customers from using our service. As we don't directly deal with any user data this should not be a large issue.

3.7 Cultural Requirements

Requirement: None of the webpage that we are in direct control over (not the music videos) shall be offensive to any possible user.
Rationale: ASK Studios should avoid offending users at all cost. It's just simply the correct thing to do. We cannot filter potentially offensive music videos without compromising the usability of our service so the best we can do is ensure that everything on our end is non offensive.

3.8 Health and Safety Requirements

Requirement: The user shall be warned that any videos viewed on Shuffle may contain rapidly flashing images that may effect those with epilepsy.
Rationale: ASK Studios does not have control over the content viewed by users, so it is our responsibility to inform the user of the potential risks of using the service.

Requirement: The user shall be warned that extended user of our service may result in eye strain.
Rationale: Extended viewing of any screen can effect the users eyes negatively. ASK Studios should inform the user of any potential risks evolved with using our product.

3.9 Legal Requirements

Requirement: The audio and video from the music video shall not be split.
Rationale: This would break YouTube's terms of service in the use of their API.

4 Project Issues

4.1 Open Issues

Issue: The project is currently missing the search bar. **RESOLVED**
Issue: The project is currently missing the playlist feature. **REMOVED**

4.2 Off-the-Shelf Solutions

YouTube's autoplay function on their site is an existing algorithm that can be used to view related videos. However, the interface is not as simple as Shuffle, and does not restrict by artist or genre.

4.3 New Problems

None.

4.4 Tasks

Tasks are outlined in our development plan.

4.5 Risks

There are a few legal risks associated with the product as we are allow users to access content that we do not own, but anything that is on YouTube should be legal for us to play, so we should not encounter any issues.

4.6 Costs

No known costs are current associated with with our product.

4.7 User Documentation and Training

The product shall provide a simple help page to provide users with a brief set of instructions on how to use the product. Since the product is very simple and should be very straightforward to use, minimal instructions should be required.

4.8 Waiting Room

None, for the moment.

4.9 Ideas for Solutions

None currently.