SE 3XA3: Development Plan
Title of Project

Team 12, DJS
Victor Velenchovsky - velech
Amandeep Panesar - panesas2
Taha Mian - miantm

October 11, 2016

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Wed. Oct. 5	0.1	Basic Outline
Wed. Oct. 5	0.2	Requirements added
Thurs. Oct. 6	0.3	Section 1 added and formatting
Thurs. Oct. 6	0.4	First draft
Thurs. Oct. 6	0.5	Formatting and minor changes
Fri. Oct. 7	0.6	First Revision complete
Sat. Oct. 8	0.7	Section 4 Complete

This document describes the requirements for .... The template for the Software Requirements Specification (SRS) is a subset of the Volere template (Robertson and Robertson, 2012). If you make further modifications to the template, you should explicity state what modifications were made.

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this project is to make it easier for people that attend social gatherings or events to select a songs and form their own playlist according to the mood or perference of the attendees. The current implementation (PlayMyWay) has an unflattering and difficult to use UI, as well as no easy way for the average person to integrate the software into their party. We plan on making a revised version that has an elengant web app interface, and an easy to install server.

Social gatherings are much more enjoyable when most of the attendee's enjoy the music that is being played. This project was inspired by

Social gatherings are much more enjoyable when most of the attendee's enjoy the music that is being played. This project was inspired by

1.2 The Stakeholders

1.2.1 The Client

Event Organizer
The client is the host of the social event or gathering, who is trying to save money by not hiring a DJ and simply relying on this system, which allows the attendees to chose what songs they want to listen to. Having users select music will put less stress on the event organizers, and allow them to focus on other aspects of the event, or let them enjoy themselves.
DJ
The client can also be a DJ, hired by a party planner, who is not willing to put up with people fighting over what song to play next.

1.2.2 The Customers

Event Attendee's
If the system is working properly and attendee's are voting for songs, then the event will be more enjoyable for them.

1.2.3 Other Stakeholders

No other stakeholders have been discovered.

1.3 Mandated Constraints

- The front-end of the product will take the form of a web-app that can run on any javascript-enabled browser. This means the app should be able to accommodate all common Operating Systems (phone and desktop) and all common Javascript-enabled browsers. Internet Explorer may be exempt
- The front-end webapp and server need to both be connected to the same WiFi network.
- The server needs to be stable with very low downtime, in order to prevent scenarios where the music stops playing accidentally.

1.4 Naming Conventions and Terminology

Shorthand	Explanation
JS	Javascript
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
Event	Any event that includes shared music listening. Party, Wedding, etc.
UI	User Interface

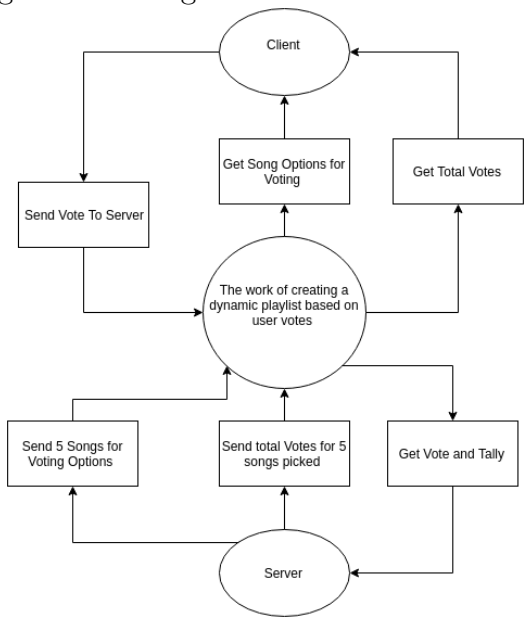
1.5 Relevant Facts and Assumptions

- We assume that users of the app are impatient (they don't want to spend a long time learning the software)

2 Functional Requirements

- The web page created in HTML and Javascript will display graphics and other information to user like title.
- The Javascript will fetch the five options for music playback and will display them on the user's browser.
- The web page should also keep track of the number of votes for any song.
- The html page should also include a graphic where the user can place their vote.
- The html page should also only list valid options for music (should only list songs stored on the server).
- Cookies will also be created to only allow one vote for the user.
- After the song has been selected to play then the html page should update to only show songs that haven't been played and should reset votes.
- The server should pick 5 random unique songs until no unique songs are left and repeat process.
- The server should total votes and share the information with the user through the html page.
- The server should sort the songs after voting and play the most voted song.
- The cookie should also remember what song the user picked and then reset after voting has reset.

Figure 1: A diagram of the context of work



2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

2.1.2 Work Partitioning

2.1.3 Individual Product Use Cases

2.2 Functional Requirements

- The web page created in HTML and Javascript will display graphics and other information to user like title.
- The Javascript will fetch the five options for music playback and will display them on the user's browser.
- The web page should also keep track of the number of votes for any song.
- The html page should also include a graphic where the user can place their vote.

- The html page should also only list valid options for music (should only list songs stored on the server).
- Cookies will also be created to only allow one vote for the user.
- After the song has been selected to play then the html page should update to only show songs that haven't been played and should reset votes.
- The server should pick 5 random unique songs until no unique songs are left and repeat process.
- The server should total votes and share the information with the user through the html page.
- The server should sort the songs after voting and play the most voted song.
- The cookie should also remember what song the user picked and then reset after voting has reset.

3 Non-functional Requirements

3.1 Look and Feel Requirements

We want to make a simple to use UI and one that is visually appealing.

3.2 Usability and Humanity Requirements

- Straight-forward web app (new users should be able to adopt it easily)
- No sign up required
- Automatically connects to server via WiFi with little to no input from user

3.3 Performance Requirements

- Songs should play one after another with small or no delay in between
- Server should have high uptime

3.4 Operational and Environmental Requirements

3.5 Maintainability and Support Requirements

3.6 Security Requirements

- A user should only be allowed to vote once per *'round'* of votes
- Restarting the app/WiFi/phone should not change the fact that a user can only vote once per *'round'*

3.7 Cultural Requirements

There are no cultural requirements for this project.

3.8 Legal Requirements

- Music that is played should have legal rights to be played publically
- The open project we are recreating has an open MIT license that can be found here

3.9 Health and Safety Requirements

This section is not in the original Volere template, but health and safety are issues that should be considered for every engineering project.

4 Project Issues

4.1 Open Issues

The most important issue right now is how exactly do we make sure that a user can only vote once per song, without requiring people to sign up for an account. We also want the user to be able to change their vote multiple times when selecting the next song and it is in queue.

4.2 Off-the-Shelf Solutions

The project that we are modeling (PlayMyWay) already does most of what our project will do.

As for other solutions, there are various libraries that we will use in our development. These libraries will help with various functionality, and include (but are not limited to):

- Express.JS (Server framework for Node.JS)
- Angular.JS (Front-end framework for the webapp)
- nodeunit (Unit testing package)
- Mocha (general testing package)

4.3 New Problems

DJ.Js is based off the open source project called PlayMyWay that can be found here. We are going to recreate the project, in javascript. The original project was written in Jade, which is a Object Oriented programming language based on Java. Jade is kind of outdated and not as universal as javascript, which is the golden standard for web page applications. We don't need any new installations just a device that can access the internet and has a internet browser that runs javascript.

4.4 Tasks

We are given an outline of the things we need for this project which includes a proof of concept, testing plan, a design, and a final presentation. The proof of concept will be early in the project but will allow us to make a very basic outline of the project like be able to get requests from a device to a server vice versa. The test plan will tell us how we are going to test our product so we know whether or not the project has successfully fulfilled our requirements The design of the project will be more specific to coding and will come later but can be broken down in these simple steps:

1. Build a node.js sever using Amandeeps Raspberry Pi for the
2. Build the UI using javascript

3. Testing (unit,general testing)

The Final Presentation will be the last step in the project when it is complete and fully functional.

4.5 Migration to the New Product

There is no transition needed because we are recreating a web app, that already exists.

4.6 Risks

Many risks can occur while trying to implement our product which include:

- The product is a webapp and therefore relies on an internet connection
- Bugs or catastrophic errors in the server could cause the music to start playing sporadically
- Only people with a device that can connect to the internet can use the web app

4.7 Costs

There will be no monetary costs, the music we are using to test our webapp will be unlicensed and free to play, and because the project we are recreating is an open source project which we are free to use in any way. The only other cost is the amount of effort and time we put in the project which should not be more than 100 hours, but could take longer.

4.8 User Documentation and Training

We will integrate a small help button at the end of the webpage that will describe how the webpage works. After this short tutorial the user should know how the website works.

4.9 Waiting Room

Something that is part of our vision is having all genres of music, having a diverse list of music so users can select songs that cater to their tastes. We also want to add "moods" that correspond to certain event you are in, so a wedding would be more of a happy mood, where as a party will have a Festival will have a celebratory mood.

4.10 Ideas for Solutions

References

James Robertson and Suzanne Robertson. *Volere Requirements Specification Template*. Atlantic Systems Guild Limited, 16 edition, 2012.

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.