Requirements Engineering Report

Date: February 17, 2016
Version v1.00

Software Project Name

Prescribe

By: Team StarMony

Project Manager: Jeremy Brown
Quality Assurance: Brandyn Deffinbaugh
Technical Lead: Mitchell Powell

Table of Contents

1. Introduction

   a. Project Objective

      The objective of our project is to provide a web application that can aid users in the discovery of new music based on their musical preferences.

   b. Project Scope

      The user will be able to input an artist or a band into search field which will provide a similar list of artists or bands. If the user creates an account using Facebook or Google+, they will be able to give feedback to train our model by up or down voting the search results. This will help grow the machine to give better feedback to our users. The user can also post their search results to their social media page. Search results will provide the user with a list of similar bands including the bands' information such as their biography and discography. The user also be able to favorite a band for later reference. We also want to include a top artists section that users can view to see which artists have been given the most up votes for the week, month, or year.

Commented [GC1]: Grammar.

   c. Success Criteria

      i. Search Functionality
      ii. Rating system integration
      iii. Smooth Web GUI
      iv. Icon graphics

Commented [GC2]: How is this a success criteria.

Commented [GC3]: How is Icon Graphics a success criteria?

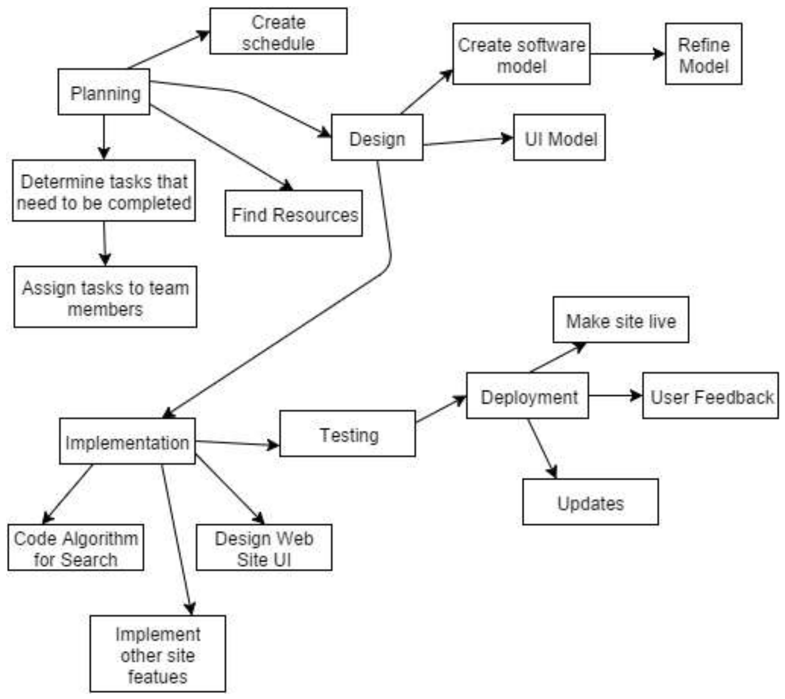
   d. Collaboration with Stakeholders

      - End-Users
      - Facebook
      - Google
      - Spotify
      - Pandora

Commented [GC4]: You are going to talk with Spotify, Pandora, Facebook, Google?

2. Project Plan

a. Work Breakdown Structure



b. Project Resources

Resource Name	Cost	Description	Status
Eclipse IDE	Free	Internal Development Environment for Java programming language	Obtained
Domain Name	$20 / year	The domain name at which the web application will be hosted	In Progress
JavaScript Interpreter/Web Browser	Free	Interpreter for the JavaScript programming language, as the front end will be developed in JavaScript	Obtained
Labor	$9375	1.5 person/months at a rate of $6,250 per person month	In Progress
Web Hosting Server Space	$10 / month	Space on a server to host the data needs of the Prescribe project.	In Progress

Total:

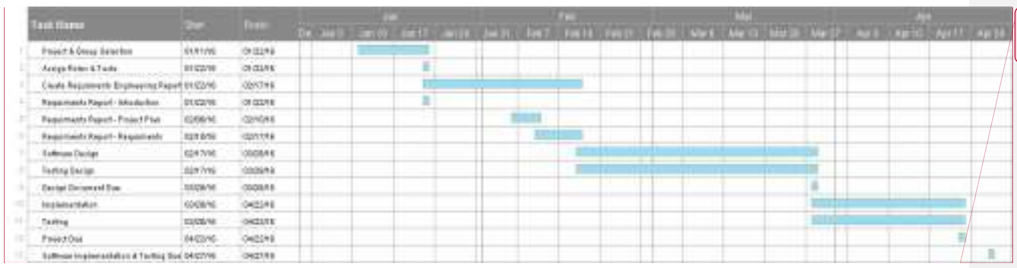
$9375 up front, $140 recurring annually

This will certainly need to be updated after a little more analysis about what will be required for the project.

c. Responsibility Matrix

Task	**Jeremy Brown**	**Brandyn Deffinbaugh**	**Mitchell Powell**
Introduction	▨	▨	▨
Work Breakdown Structure	▨		
Project Resources			▨
Responsibility Matrix	▨		
Gantt Chart	▨		
PERT Chart	▨		
Cost Estimation		▨	
Risk Plan		▨	
Project Monitoring			▨
Control Mechanisms			▨
Major Software Functions	▨	▨	▨
Use Case Diagrams	▨		
Use Case Descriptions	▨		
Sequence Diagrams			▨
Activity Diagrams			▨
Requirements Class Models			▨
Prototype Description	▨	▨	▨
Data Directory		▨	
Limitations & Constraints		▨	
Bibliography	▨	▨	▨
Non-functional Requirements		▨	
Problems Encountered	▨	▨	▨
Product Testing	▨	▨	▨
Quality Assurance		▨	
Project Management	▨		
GUI Design			▨
Implementation	▨	▨	▨
Software Design			▨

d. Gantt Chart



Commented [GC5]: Spelling error: Requirements. Repeated in Pert Chart.

e. Pert Chart

Project Selection				Software Design	Implementation
Start: 1/11/2016	Requirements Report	Requirements Report	Requirements Report	*Start: 2/17/2016*	*Start: 3/28/2016*
End: 1/22/2016	Introduction	Project Plan	Reuirements	*End: 3/28/2016*	*End: 4/22/2016*
	Start: 1/22/2016	*Start: 2/06/2016*	*Start: 2/06/2016*		
Group Seleccion	*End: 1/22/2016*	*End: 2/10/2016*	*End: 2/10/2016*	Testing Design	Testing
Start: 1/11/2016				*Start: 2/17/2016*	*Start: 3/28/2016*
End: 1/22/2016				*End: 3/28/2016*	*End: 4/22/2016*

f. Cost Estimation

i. Function Point Estimation

Information	Simple FP	Average FP	Complex FP	Total FP
User Input	9	4	6	19
User Output	-	10	-	10
User Online Queries	-	8	-	8

Logical Files	7	-	-	7
External Interface	-	21	-	21
Total Function Points	-	-	-	65

Commented [GC6]: 21 external interfaces? That's a lot.

Commented [GC7]: Spend just a few seconds formatting this chart.

Reliable Backup and Recovery	2
Specialized Data Communication	4
Distributed Processing Functions	0
Critical Performance	1
Operation in Heavily Utilized Operational Environment	0
Online Data Entry	3
Online Data Entry via multiple screens	0
ILFs updated online	2
Inputs, Outputs, Files, or Inquiries Complexity	5
Internal Processing Complexity	4
Reusable Code Design	3
Conversion and Installation Inclusion	0
Designed for Multiple Installations in Different Organizations	0
Facilitate Change and Ease of Use	5
Total	29

Total Function Points = FP = 65 X [0.65 + 0.01 X 29] = 61.5

ii. Lines of Code Estimation

Functions	Good LOC Est.	Average LOC Est.	Bad LOC Est.
GUI	250	500	750
Database/API Queries	150	300	450
Backend	500	700	900
Log-in Support	200	400	600
Estimated LOC (*Based on Average Case*)	1900		

With the average productivity rate of 620 LOC/person-month, we can complete 1,860 LOC/month. Assuming $8,000 per person-month of labor the cost using LOC estimation is $24,000 a month. This equates to roughly $12.90 per LOC, with this the total cost of the project will be $24,510.

iii. Cost Estimation

COCOMO II

Objects	Quantity	Weight	Total
Screens	4	4	16
Reports	3	6	18
Components	4	10	40
		Total OP	74

NOP = (OP) X [(100 - %Reusable Code)/100]
NOP = 74 x [(100-10)/100] = 66.6
PROD = NOP/Person-Month
Effort = 66.6/13 = 5.12 Person-Months

Assuming a labor rate of $8,000 per person-month, the COCOMO II model estimates that this project will cost $40,960.

g. Risk Plan

Impact Levels

Commented [GC8]: Access to an appropriate database seems like a risk. Which database will you use? Does it have the information you need?

1-Negligible
2-Moderate
3-Severe
4-Catastrophic

Risk	Probability	Impact	Mitigation
Team member Leaving	15%	4	Redistribute work load. Possible loss of minor features.
Falling behind	25%	2-3	Meet and re-plan work schedule to accommodate work load.
Lack of skill	25%	2-3	Find help, watch videos, ask someone who can help us.

Commented [GC9]: One word.

Commented [GC10]: One word.

h. Project Monitoring & Control Mechanisms

There are several mechanisms that the Starmony group will utilize in order to ensure effective communication and proper maintenance of the code base. As a group, we will meet a minimum of once a week in a formal, face-to-face meeting to discuss the progress of the project, any issues that have arisen and for code review. In addition, the group will maintain regular group-wide electronic communication to ensure that all group members are kept up to date with what progress has been made, short term goals, and current issues. The group will maintain a “TODO” document on the Git repository to ensure that there is effective communication about what current issues are at hand.

It will be the responsibility of all group members to pull any changes from the remote repository whenever they work on their local branches so any integration issues can be caught early and fixed before they become a larger problem. In turn, it will also be the responsibility of each group member to push any completed changes or additions to the master branch of the remote repository as often as possible. Any works in progress should be maintained in a development branch to ensure a compiling codebase on the master branch at all times.

3. Requirements & Analysis Models

   a. Major Software Functions

1. Registered User Functions
    - 1.1 Search
    - 1.2 Login
    - 1.3 Search Result Feedback
        - 1.3.1 Upvote or Downvote
        - 1.3.2 Save Artists
    - 1.4 View Profile Info
    - 1.5 View Artist Info

Commented [GC11]: Level of indentation is wrong.

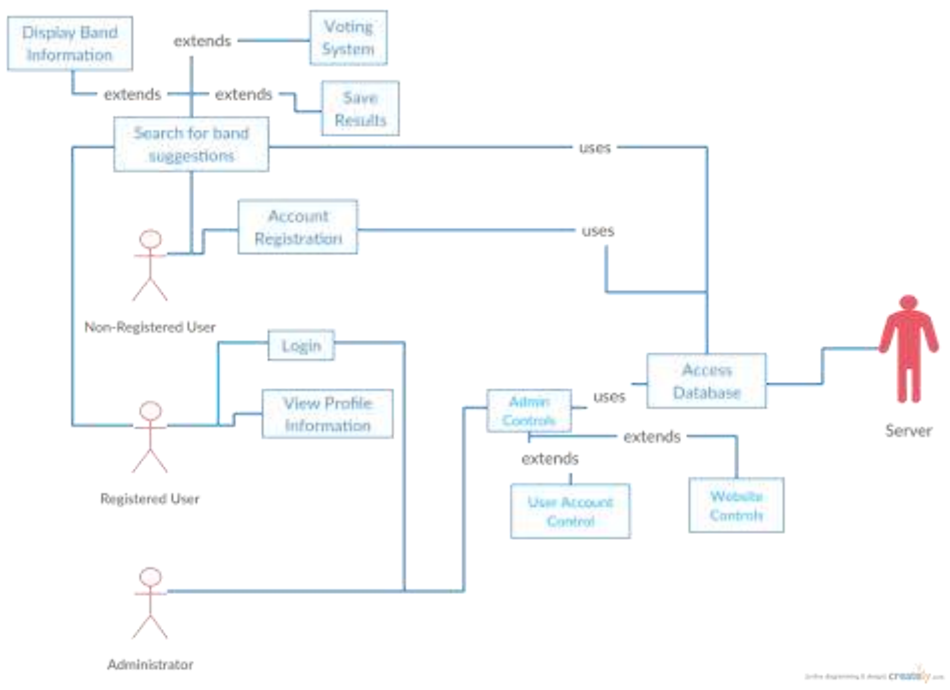
2. Non-Registered User Functions ***
    - 2.1 Search
    - 2.2 Account Registration
    - 2.3 View Artist Information

3. Administrator Functions (Agent)
    - 3.1 Login
    - 3.2 User Account Control
    - 3.3 Website Controls (Manual Override)

4. Server Functions (No Agent)
    - 4.1 Login Authentication
    - 4.2 Search Caching
    - 4.3 Information Retrieval
    - 4.4 Top Artist Info (Most upvotes, searches, etc...)
    - 4.5 Email Confirmation
    - 4.6 Account Removal (Via facebook, Google+

b. Use Case Diagram

Commented [GC12]: Use cases should all be verb phrases. Use ovals for use cases. Databases should be in rectangles. Don't mix objects with use cases.



c. Use Case Descriptions

Use Case Number: 01

Use Case Name: Search for Band Suggestions

Primary Actor: User (both registered and non-registered)

Secondary Actor: Server

Goal: Allow user to input artist and return suggestions

Preconditions: User must be on the website

Trigger: User wants to discover new artists

Scenario:

1. The user accesses the webapp.
2. The user inputs artist they wish for the search results to be based off of.
3. The server takes the input and delivers the search results.

Exception: The artist entered by the user is not a band or is not discoverable.

Priority: Essential, this is the main feature of our software.

Commented [GC13]: How about a particular song rather than the artist?

Commented [GC14]: What do you mean not discoverable? Do you mean the artist isn't in the database?

Use Case Number: 02

Use Case Name: Account Registration

Primary Actor: Non-Registered User

Secondary Actor: Server

Goal: Allow a non-registered user to create an account

Preconditions:

- The system must be set up to allow user to link an account to our webapp
- User must have a Google or Facebook account

Trigger: User wants to create an account.

Scenario:

1. The user accesses the webapp.
2. The user links Google or Facebook account.
3. The server performs the necessary steps to connect account.
4. The server stores this information for future use.

Exceptions: The user does not have a Google are Facebook account

Priority: Moderate. This step will be necessary for us to implement our save and rating features.

Use Case Number: 03

Use Case Name: Login

Primary Actor: Registered User

Secondary Actor: Server

Goal: Allow user to login to their account.

Preconditions: User has already registered.

Trigger: User wants to access their account.

Scenario:

1. User goes to the login page
2. User enters username and password
3. Server confirms and logins in the user.

Exceptions:

- The user does not have an account
- The user enters the wrong information

Priority: Moderate. This step will be necessary for us to implement our save and rating features.

Use Case Number: 04

Use Case Name: Voting System

Primary Actor: Registered User

Secondary Actor: Server

Goal: Allow user to login to upvote or downvote a band in their search results.

Preconditions:

- User is logged in
- User has already searched for results

Trigger: User wants to vote on their results.

Scenario:

1. User searched for suggestions
2. Server has returned results
3. User selects whether they like a specific band (upvote) or if they do not (downvote)

Exceptions:

- The user does not provide a rating

Priority: Low. This step will help us implement our top artists of the week, month, and year page.

Use Case Number: 05

Use Case Name: Save Results

Primary Actor: Registered User

Secondary Actor: Server

Goal: Allow user to save a suggested band for future use

Preconditions:

- User is logged in
- User has already searched for results

Trigger: User wants to vote save their results.

Commented [GC15]: Delete vote.

Scenario:

1. The user has searched for suggestions
2. The server has provided results.
3. The user favorites an artist

Exceptions:

- The user does not save band

Priority: Moderate.

Use Case Number: 06

Use Case Name: Display Band Information

Primary Actor: User (both registered and non-registered)

Secondary Actor: Server

Goal: Allow user to view information on a suggested band

Preconditions:

- User is logged in
- User has already searched for results

Trigger: User wants to learn more about an artist

Scenario:

1. The user has searched for suggestions
2. The server has returned suggestions

3. The user clicks the display information button under an artist’s name
4. The webapp displays the artist’s information.

Exceptions:

- The user does not click the display information button

Priority: Low.

Use Case Number: 07

Use Case Name: View Profile Information

Primary Actor: Registered Information

Secondary Actor: Server

Goal: Allow user to view their personal information

Preconditions:

- User is logged in

Trigger: User wants to view their profile

Scenario:

1. The user clicks a profile button that will take them to their account page
2. Server brings user to that page and displays the user’s information

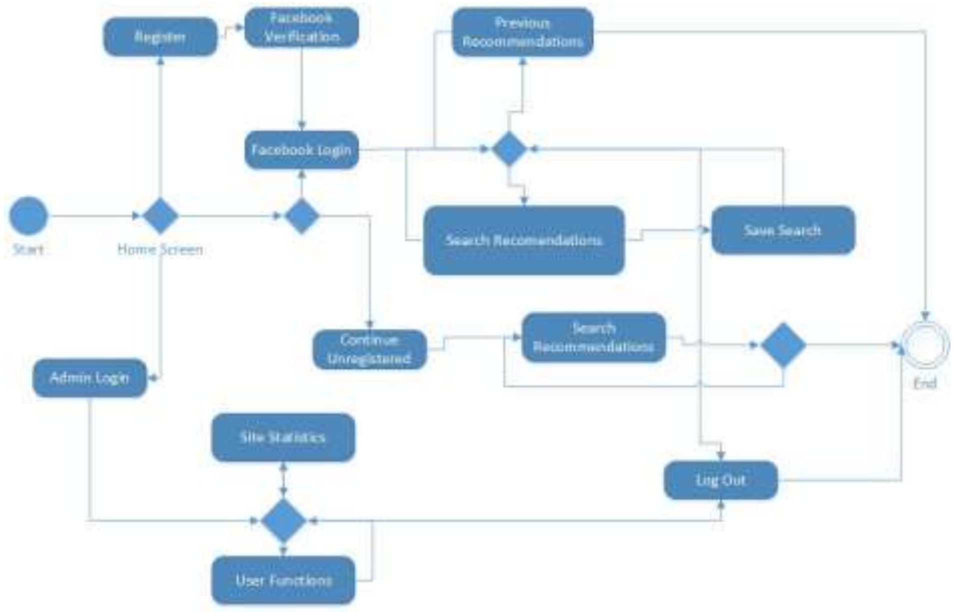
Exceptions:

- The user does not view account information.
- The user is not logged int.

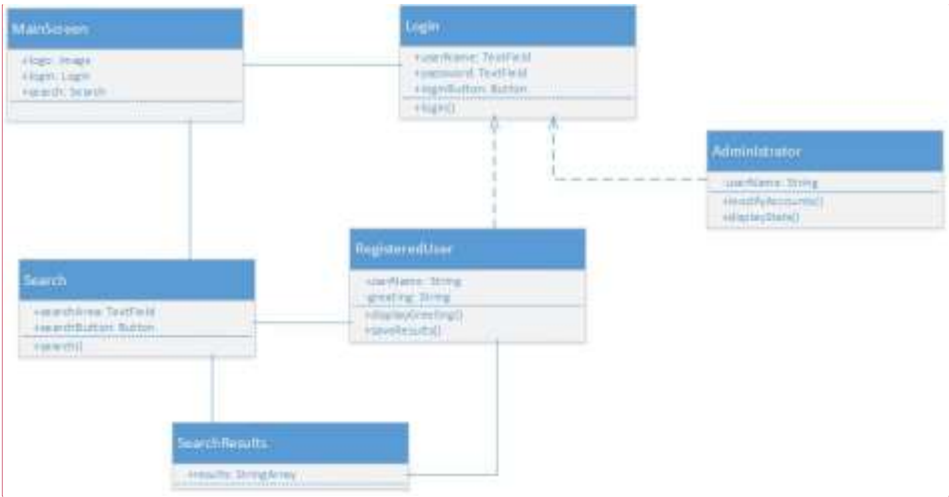
Priority: Low. Must be implemented if the favorite system is created.

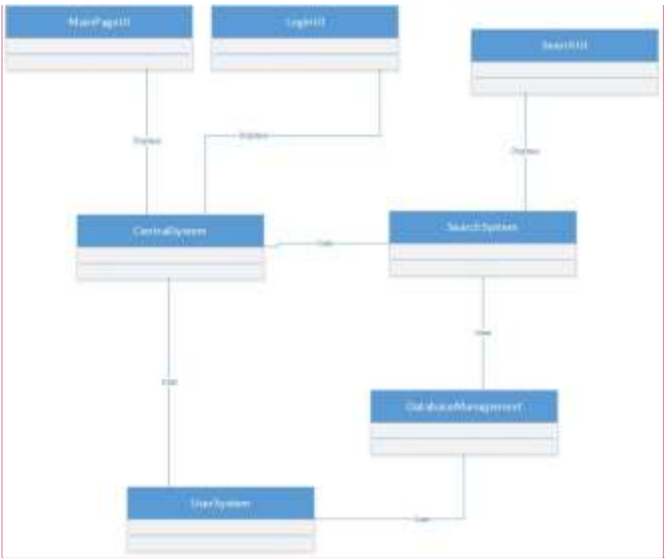
d. Activity Diagrams

Commented [GC16]: Nice.



e. Requirements Class Models





Commented [GC17]: Login, I think, is the LoginScreen. Or is it? I'm confused as to what it is.

Commented [GC18]: Administrator and RegisteredUser share some information. There should be a superclass to hold that shared information.

Commented [GC19]: Shouldn't there be relationships between these objects and the above objects??

f. Data Directory

Administrator – Project members will have access to unique controls allowing them access to the user and data databases as well as control over the website's interface.

Non Registered User – Any person who is using the website's functions but is not logged in.

Registered User - Any person using the website's functions while being logged into an account.

Saved Searches – Searches that a register user has saved to their account for later viewing.

Commented [GC20]: registered

Rating System – User is giving their input on the relevancy of the search results.

g. Limitations & Constraints

Constraints: User's must have an update-to-date internet browser to use all the of website's functions.

Commented [GC21]: Not possessive.

Commented [GC22]: I think you mean up-to-date.

h. Non-functional Requirements

Reliability: The application will be up at all times unless it needs to be taken down for maintenance or external forces such as loss of power.

Usability: The user will be able to use the application with no training or reading required.

Security: Security for the login and passwords of users will be handled by Facebook and Google+ integration.

4. Problems Encountered

- In calculating Function Points, it was unclear what to count at User inputs and what to count as user online queries, additionally it was not clear how to distinguish between External Interfaces and User Online Queries
- Unclear what Logical files are
- Cannot install Visio or project to laptop. Had to use online tools and Microsoft office

Commented [GC23]: Period. Start a new sentence.

Commented [GC24]: External Interfaces are external databases or external programs that your program interacts with. For instance, Facebook is an external interface for your project.

Commented [GC25]: Logical files are files that you create and use within your software.

5. Bibliography