10/02/19

**CSCI 375 Software Engineering**

**System Requirements Specification**

Product name (working title): Melophile

Customer name: Dr. Burge

Team name: The Creative 4

Team members:

* Michaun Pierre
* Jahlil Allen
* Nobel Gebru
* Ryan Davis

[Working title: Melophile]

System Requirements Specification

**Table of Contents**

1. Introduction

* 1. Purpose of This Document
  2. References
  3. Purpose of the Product
  4. Product Scope

2. Functional Requirements

3. Non-Functional Requirements

4. User Interface

5. Deliverables

6. Open Issues

Appendix A – Agreement Between Customer and Contractor

# 

# 1. Introduction

## 1.1 Purpose of This Document

The purpose of this document is to clearly define the requirements of The Creative 4’s software engineering project, and the resulting product.

## 1.2 References

* Wiegers, K. (2019, September 24). Defining Project Scope: Context and Use Case Diagrams. Retrieved from https://www.jamasoftware.com/blog/defining-project-scope-context-use-case-diagrams/

## 1.3 Purpose of the Product

The product allows users to test and share their musical knowledge through a social and competitive challenge experience.

## 1.4 Product Scope

This diagram below illustrates the scope of our music app. Users can be both a challenge maker and receiver once they create an account and login on the app. Depending on whether they’ve created or received a challenge a user will see their current and completed challenges on their home dashboard. All interaction between users occurs through these challenges.



# 2. **Functional Requirements**

Each functional requirement should be represented using a use case.

Refer the reader to the top-level use case/context diagram referred to in Section 1.4. In addition, include separate use case diagrams, where appropriate, for each of the top-level use cases.

In addition to the diagrams, every use case should be documented using the following use case details specification format.

|  |  |  |
| --- | --- | --- |
| **Number** | 1 | |
| **Name** | Sign Up | |
| **Summary** | Creates an account for the user for data persistence. | |
| **Priority** | 5 | |
| **Preconditions** | User must have a working internet connection in order to make a request to the web app. | |
| **Postconditions** | A new user account will be created and logged in the database. | |
| **Primary Actor** | User | |
| **Secondary Actors** | Database | |
| **Trigger** | User clicks the sign up button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks sign up button |
|  | 2 | User enters desired username, password, password confirmation |
|  | 3 | User credentials are logged into database |
|  | 4 | Upon success user is redirected to login page in order to complete login |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | None | |

|  |  |  |
| --- | --- | --- |
| **Number** | 2a | |
| Name | Login | |
| **Summary** | Opens a user account in order to have data persistence. | |
| **Priority** | 5 | |
| **Preconditions** | User must have a working internet connection in order to make a request to the web app. | |
| **Postconditions** | The user will be logged into their account. | |
| **Primary Actor** | User | |
| **Secondary Actors** | Database, Google, Facebook | |
| **Trigger** | User clicks login button. | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks Login button |
|  | 2 | User enters login credentials |
|  | 3 | Check database for the credentials |
|  | 4 | Upon successful confirmation user is redirected to their dashboard |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | User opts to complete 3rd party login **:**  < 3rd Party Login> |
| **Open Issues** | Whether we will allow 3rd party login. | |

|  |  |  |
| --- | --- | --- |
| **Number** | 2b | |
| Name | 3rd Party Login | |
| **Summary** | Opens a user account in order to have data persistence. | |
| **Priority** | 5 | |
| **Preconditions** | User must have a working internet connection in order to make a request to the web app. | |
| **Postconditions** | The user will be logged into their account. | |
| **Primary Actor** | User | |
| **Secondary Actors** | Database, Google, Facebook | |
| **Trigger** | User clicks login button. | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks 3rd party login option |
|  | 2 | User enters 3rd party credentials |
|  | 3 | Upon success user is redirected to their dashboard |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | None | |

|  |  |  |
| --- | --- | --- |
| **Number** | 3 | |
| Name | Create “Name That Song” | |
| **Summary** | Create a name that song challenge for a user. | |
| **Priority** | 5 | |
| **Preconditions** | Must be logged in to an account | |
| **Postconditions** | A name that song challenge will be created and sent to a user | |
| **Primary Actor** | Challenge Maker | |
| **Secondary Actors** | Challenge Receiver | |
| **Trigger** | User clicks the create challenge button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks create challenge button |
|  | 2 | User selects name that song challenge type |
|  | 3 | User selects song and user to send challenge to |
|  | 4 | Upon confirmation log challenge in database, will populate in receiver’s dashboard upon refresh |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | None | |

|  |  |  |
| --- | --- | --- |
| **Number** | 4 | |
| Name | Create “Finish That Lyric” | |
| **Summary** | Create a finish that lyric challenge for a user. | |
| **Priority** | 5 | |
| **Preconditions** | Must be logged in to an account | |
| **Postconditions** | A finish that lyric challenge will be created and sent to a user | |
| **Primary Actor** | Challenge Maker | |
| **Secondary Actors** | Challenge Receiver | |
| **Trigger** | User clicks the create challenge button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks create challenge button |
|  | 2 | User selects finish that lyric challenge type |
|  | 3 | User selects song and user to send challenge to |
|  | 4 | Upon confirmation log challenge in database, will populate in receiver’s dashboard upon refresh |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | None | |

|  |  |  |
| --- | --- | --- |
| **Number** | 5 | |
| Name | View Challenges | |
| **Summary** | Allow user to view current and completed challenges. | |
| **Priority** | 5 | |
| **Preconditions** | A user is logged in and they have a challenge to complete. | |
| **Postconditions** | The user will be able to view their challenges | |
| **Primary Actor** | Challenge Receiver | |
| **Secondary Actors** | None | |
| **Trigger** | User refreshes dashboard | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User refreshes dashboard |
|  | 2 | Refresh makes query to database for all active challenges for the user |
|  | 3 | Results array is displayed on the dashboard screen as a collection of card views |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | User clicks start challenge on “Name that Song” card view **:**  Play “Name That Song” |
|  | 1b | User clicks start challenge on “Finish that Lyric” card view **:**  Play “Finish that Lyric” |
| **Open Issues** | Do we actually have a button to refresh or a swipe/ automatic refresh? | |

|  |  |  |
| --- | --- | --- |
| **Number** | 6 | |
| Name | Play “Name That Song” | |
| **Summary** | Allow user to complete a name that song challenge. | |
| **Priority** | 5 | |
| **Preconditions** | User has an active “Name that Song” challenge to complete | |
| **Postconditions** | User completed “Name that Song” challenge | |
| **Primary Actor** | Challenge Receiver | |
| **Secondary Actors** | None | |
| **Trigger** | User clicks start challenge button on card view | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks start challenge button on card view |
|  | 2 | User is redirected to challenge page |
|  | 3 | Upon second start challenge button click music snippet starts along with challenge timer |
|  | 4 | If possible answer is entered, compare answer likeness to actual answer |
|  | 5 | If likeness is satisfactory give points to the user, update status of challenge to complete, update to show answer, and exit challenge |
|  | 6 | If wrong answer is entered, or time has run out update the challenge to show the answer, set the challenge to completed and exit the challenge |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | Is there going to be a leader board? | |

|  |  |  |
| --- | --- | --- |
| **Number** | 7 | |
| Name | Play “Finish That Lyric” | |
| **Summary** | Allow a user to complete a finish that lyric challenge.. | |
| **Priority** | 5 | |
| **Preconditions** | User has an active “Name that Song” challenge to complete | |
| **Postconditions** | User completed “Name that Song” challenge | |
| **Primary Actor** | Challenge Receiver | |
| **Secondary Actors** | None | |
| **Trigger** | User clicks start challenge button on card view | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks start challenge button on card view |
|  | 2 | User is redirected to challenge page |
|  | 3 | Upon second start challenge button click music snippet starts along with challenge timer |
|  | 4 | If possible answer is entered, compare answer likeness to actual answer |
|  | 5 | If likeness is satisfactory give points to the user, update status of challenge to complete, update to show answer, and exit challenge |
|  | 6 | If wrong answer is entered, or time has run out update the challenge to show the answer, set the challenge to completed and exit the challenge |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | Is there going to be a leader board? | |

|  |  |  |
| --- | --- | --- |
| **Number** | 8 | |
| Name | Update Challenge | |
| **Summary** | Allow a user to update the challenge they created | |
| **Priority** | 4 | |
| **Preconditions** | User has an active challenge that they created | |
| **Postconditions** | User updated desired challenge | |
| **Primary Actor** | Challenge Maker | |
| **Secondary Actors** | Challenge Receiver | |
| **Trigger** | User clicks edit button on active created challenge card view | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks edit button on challenge card view |
|  | 2 | User is redirected to edit challenge page |
|  | 3 | User can change challenge details as desired |
|  | 4 | Upon save, challenge attributes are saved to the database. |
|  | 5 | User redirected back to active created challenges |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | May need a pending challenge state to prevent updating while challenge is being played. | |

|  |  |  |
| --- | --- | --- |
| **Number** | 9 | |
| Name | Delete Challenge | |
| **Summary** | Allow a user to delete a challenge | |
| **Priority** | 4 | |
| **Preconditions** | User has an active challenge | |
| **Postconditions** | User deleted desired challenge | |
| **Primary Actor** | Challenge Receiver | |
| **Secondary Actors** | None | |
| **Trigger** | User clicks delete button on challenge card view | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks delete button on challenge card view |
|  | 2 | User is prompted with a confirmation page |
|  | 3 | Upon confirmation challenge is deleted from database |
|  | 4 | User is redirected back to their active challenges to complete |
| **Extensions** | **Step** | **Branching Action** |
|  | 1a | None |
| **Open Issues** | Should a challenge maker be able to delete the challenge once they’ve made it?  If a challenge receiver keeps deleting a challenge sent by a challenge maker should we prevent them from sending it again, or do nothing? | |

Lastly, write the tests that will be used during system and acceptance testing to verify that each requirement has been met. Note that a single requirement may require multiple tests, so be thorough. It is also possible that a single test verifies more than one requirement. The goal is to come up with the minimum number of test cases that thoroughly test the system. Make sure that the test numbers correspond to the use case numbers.

Test List

1.) Sign up user test

2a.) Login created user test

2b.) Login through Google

2b.) Login through Facebook

3.) Create a name that song challenge test

4.) Create a finish that lyric challenge test

5.) View challenges test

6.) Playing name that song challenge test

7.) Playing finish that lyric challenge test

8.) Update challenge test

9.) Delete challenge test

# 3. **Non-Functional Requirements**

Decide on a standard format for the non-functional requirements (NFRs). Included in the format should be a unique number for each NFR, a priority (1 = lowest, 5 = highest), a clear, concise description, and the test(s) that will be used during system and acceptance testing to verify that the requirement has been met. Make sure that the test case (TESTING Report) numbers correspond to the NFR numbers. Note that you must include a minimum of 10 NFRs specific to product requirements, organizational requirements, and external requirements.

Having a hard time thinking of nonfunctional requirements? Here are some categories and "trigger questions" that may help.

**Trigger Questions**

* User Interface and Human Factors
  + What type of user will be using the system?
  + Will more than one type of user be using the system?
  + What sort of training will be required for each type of user?
  + Is it particularly important that the system be easy to learn?
  + Is it particularly important that users be protected from making errors?
  + What sort of input/output devices for the human interface are available and what are their characteristics?
* Documentation
  + What kind of documentation is required?
  + What audience is to be addressed by each document?
* Hardware Considerations
  + What hardware is the proposed system to be used on?
  + What are the characteristics of the target hardware, including memory size and auxiliary storage space?
* Performance Characteristics
  + Are there any speed, throughput, or response time constraints on the system?
  + Are there size or capacity constraints on the data to be processed by the system?
* Error Handling and Extreme Conditions
  + How should the system respond to input errors?
  + How should the system respond to extreme conditions?
* **System Interfacing**
  + Is input coming from systems outside the proposed system?
  + Is output going to systems outside the proposed system?
  + Are there restrictions on the format or medium that must be used for input or output?
* Quality Issues
  + What are the requirements for reliability?
  + Must the system trap faults?
  + Is there a maximum acceptable time for restarting the system after a failure?
  + What is the acceptable system downtime per 24-hour period?
  + Is it important that the system be portable (able to move to different hardware or operating system environments)?
* System Modifications
  + What parts of the system are likely candidates for later modification?
  + What sorts of modifications are expected?
* Physical Environment
  + Where will the target equipment operate?
  + Will the target equipment be in one or several locations?
  + Will the environmental conditions in any way be out of the ordinary (for example, unusual temperatures, vibration, magnetic fields)?
* Security Issues
  + Must access to any data or the system itself be controlled?
  + Is physical security an issue?
* Resources and Management Issues
  + How often will the system be backed up?
  + Who will be responsible for the back up?
  + Who is responsible for system installation?
  + Who will be responsible for system maintenance?

|  |  |
| --- | --- |
| Non-Functional Questions | **Priorities(1=Lowest, 5=Highest)** |
| Which music API should be used? | 5 |
| What hardware is the system to be used on? | 4 |
| How should we prevent any non-official Apple/Spotify songs? | 5 |
| How can we make the web app user friendly? | 5 |
| How often should we do maintenance? | 3 |
| How should the app react to multiple participants? | 2 |
| Should the app be family friendly? (censoring curse words in some songs) | 2 |
| Are explicit versions of songs allowed? | 2 |
| Leaderboards should be modified later?(More ideas for different types of scores) | 2 |
| Password protection(Preventing break-ins and/or multiple password attempts. | 3 |
| How often shall we perform maintenance on the website | 4 |

# 4. **User Interface**

See “User Interface Design Document for *Melophile* design.”

# 5. **Deliverables**

Provide a list of all deliverable items (that is, all artifacts that you will deliver to the customer). This list will include items such as the product itself (What format? Source code? Executable code? Object code?), documentation, and training resources (if any). Specify when (date) and in what format (e.g., hard copy, CD) each will be delivered. A tabular format works well for this section. We will assume that the deliverable items are as follows:

An electronic copy of the following:

* Systems Requirement Specification
* System Design Document
* User Interface Design Document
* User/Administrator Manual
* The executable program
* Any other software required for installation and execution of the delivered program.
* Url leading to web interface
* link to all github source code

# 6. **Open Issues**

Issues that have been raised and do not yet have a conclusion. These issues will be addressed later in the development process.

**Appendix A – Agreement Between Customer and Contractor**

Place on a separate page. Describe what the customer and your team are agreeing to when both parties sign off on this document. [One paragraph] Include a statement that explains the procedure to be used in case there are future changes to the document. [One paragraph] Provide lines for typed names, signatures, and dates for each team member and the customer. Provide space for customer comments.

**Appendix B – Team Review Sign-off**

Place on a separate page. Provide a brief paragraph stating that all members of the team have reviewed the document and agree on its content and format. Provide lines for typed names, signatures, dates, and comments for each team member. The comment areas are to be used to state any minor points regarding the document that members may not agree with. Note that there cannot be any major points of contention.

**Appendix C – Document Contributions**

Identify how each member contributed to the creation of this document. Include what sections each member worked on and an estimate of the percentage of work they contributed. Remember that each team member must contribute to the writing (includes diagrams) for each document produced.

# Appendix A - Agreement Between the Customer and the Contractor (Developer)

* Feature must be reasonable to make
* Component has to be relevant towards the website
* If the contractor or developer is not authorized or does not know the area of the website that the customer is asking to change then the contractor needs to speak to the original developer who worked on the specific area of interest
* If one customer has multiple request for multiple developers then a work order needs to be put in towards all developers involved on said work order
* If there is a need to have a trouble-shooting fix on a concept then consult with a developer whom worked on the feature that the customer is having problems with
* Customer must submit name and problem, concern, or future contribution in a message before presenting it to the developers and consultants

In case of future changes to the project, both the contributor and the customer must come to terms with the change that they want on the project. A request to change a certain feature or component must be accepted before any implementation or testing. Once the request is accepted, the feature or component must be apart of the function requirements or backlog depending on project workflow. The team will use Agile methodology to incorporate the customer in on the testing phase to see if the quality of the feature or component are satisfactory. If customer admires what the team has done then the feature or component will be ready for launch. Otherwise the team will have to modify the feature or component to meet the customers quality.

Team Name Signatures

1. Print:\_Michaun Pierre\_\_\_\_\_\_\_\_\_\_ Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Print:\_Jah’lil Allen\_\_\_\_\_\_\_\_\_\_\_\_ Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Print:\_Ryan Davis\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
4. Print:\_Nobel Gebru \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Customer

Print:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Comments:

# 

# 

# Appendix B - Team Review Sign-Off

* Every member of the development team must sign off for each given feature
* Must go through a complete preview of said feature with entire team
* Presentations for added features would be necessary for user interface components
* For backend support there needs to be a brief description of the support system

Attention to all team members, you must review and sign off on the review document. I you want to change something before signing off please do so. If any team member would like to address an issue that needs to have everyone's attention please speak up or forever hold your peace. If anyone feels compelled to make further changes please put said changes in the backlog if these are future functional requirements.

1. Name: Michaun Pierre Signature: Date: 10/15/19

Comment:

1. Name: Jah’lil Allen Signature: Date:10/16/19

Comment:

1. Name:Ryan Davis Signature: Date:10/16/19

Comment:

1. Name:Nobel Gebru Signature: Date:10/16/19

Comment:

# Appendix C - Document Contributions

* If a module is made by someone the developer that worked on the original feature and the module for it plus his contributor must sign their names

Contributions that each team member gave to this document:

Michaun Pierre - Introduction, Functional Requirements, Non-Functional Requirements

Nobel Gebru - User Interface

Ryan Davis - Deliverables

Jah’lil Allen - Appendices A-C