Version 0.2

User requirements Document

Individual Track

# REVISION TABLE

07.09.2022

Mariela Gocheva

Student number: 4552458

|  |  |  |
| --- | --- | --- |
| Version | Date | Rationale |
| 0.1 | 7 September 2022 | First Draft |
| 0.2 | 8 September 2022 | More detailed use cases; section 5 updated |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of content**

[REVISION TABLE 0](#_Toc114484541)

[1. INTRODUCTION 3](#_Toc114484542)

[1.1. Purpose 3](#_Toc114484543)

[2. CONSTRAINTS 4](#_Toc114484544)

[3. REQUIREMENTS PRIORITIZATION (MoSCoW method) 5](#_Toc114484545)

[4. NON-FUNCTIONAL REQUIREMENTS 6](#_Toc114484546)

[5. FUNCTIONAL REQUIREMENTS 7](#_Toc114484547)

[5.1. Roles Description 7](#_Toc114484548)

[5.1.1. DJ 7](#_Toc114484549)

[5.1.2. Regular user 7](#_Toc114484550)

[5.2. Web Application Requirements (regarding DJ role) 8](#_Toc114484551)

[5.3. Web Application Requirements (regarding a regular user) 9](#_Toc114484552)

[6. REQUIREMENT TRACEABILITY MATRIX 10](#_Toc114484553)

[7. USE CASES 11](#_Toc114484554)

# INTRODUCTION

## Purpose

This document defines the user requirements for the new music streaming and sharing platform named DJingle. The information stated in this document will be used as the basis for subsequent development activities including design, implementation, testing and post-implementation review.

# CONSTRAINTS

|  |  |
| --- | --- |
|  | **Description** |
| UX | Software solution is expected to be a modern application – it should have a proper UX and user feedback. An appropriate product backlog is also required (should contain user stories, time estimation and acceptance criteria). Other needed required common practices include modern and reliable way of authentication and authorization. |
| Technologies | Back-end – Java  Front-end – React and Java Script  Database - MySql |
| Meetings | Weekly meetings with tutor are mandatory. |
| Version control | FHICT GitLab (<https://git.fhict.nl>).  The project should support continuous integration. |
| Website | The website should run on Luna server. |
| Database | The database should run on Hera server. |
| Submission | Final submission with all deliverables must be submitted on Canvas before the deadline. |

# REQUIREMENTS PRIORITIZATION (MoSCoW method)

Prioritizing user requirements has a vital importance for every project regardless its resources, working conditions and constraints. Prioritization improves time management, which consequently enhances the quality of work. Having said that, I chose to use the MoSCoW method for managing these user requirements because I find this prioritization technique as an excellent match for my needs. The acronym MoSCoW represents four categories of initiatives: must-have, should-have, could-have and won’t-have. This separation provides to all of the stakeholders the opportunity to easily keep track of whether the current progress meets the long-term set goals.

M – Must have

S – Should have

C – Could have

W – Won’t have

# NON-FUNCTIONAL REQUIREMENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Category** | **Requirement Title** | **Requirement Priority** |
| **NFR-01** | System Control | Maintainable and extendable | **M** |
| **NFR-02** | System Control | Bug free system | **M** |
| **NFR-03** | System Security | Secure software | **M** |

# FUNCTIONAL REQUIREMENTS

## Roles Description

### DJ

A DJ has free access to the web application. Using the software system, he’s able to make use of the following functionalities:

* Create, visualize, update and delete playlists for free
* Use another DJ’s playlists in exchange of payment
* Visualize a rank list of the most followed DJs
* Visualize a rank list of the most listened playlists

### Regular user

A regular user has access to the web application only if a surcharged payment is fulfilled. The user can register in the website and then visualize the available playlists and DJs. Moreover, the user is able to choose a playlist and add it to his library via additional payment.

## Web Application Requirements (regarding DJ role)

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Requirement Title** | **Category** | **Requirement Priority** |
| **FR-01** | DJs should be able to create an account | User management | **M** |
| **FR-02** | DJs should be able to log in | User management | **M** |
| **FR-03** | DJs should be able to log out | User management | **M** |
| **FR-04** | DJs should be able to create playlists | Playlist management | **M** |
| **FR-05** | DJs should be able to visualize playlists | Playlist management | **M** |
| **FR-06** | DJs should be able to update playlists’ name | Playlist management | **M** |
| **FR-07** | DJs should be able to add songs to a playlist | Playlist management | **M** |
| **FR-08** | DJs should be able to remove songs from a playlist | Playlist management | **M** |
| **FR-09** | DJs should be able to delete a playlist | Playlist management | **M** |
| **FR-10** | DJs should be able to change their account information | User management | **M** |

## Web Application Requirements (regarding a regular user)

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Requirement Title** | **Category** | **Requirement Priority** |
| **FR-11** | Users should be able to register | User management | **M** |
| **FR-12** | Users should be able to log in | User management | **M** |
| **FR-13** | Users should be able to log out | User management | **M** |
| **FR-14** | Users should be able to visualize playlists | Playlist management | **M** |
| **FR-15** | Users should be able to follow a DJ | User management | **M** |
| **FR-16** | Users should be able to start a playlist after paying for it | Playlist management | **M** |
| **FR-17** | Users should be able to see a rank list with most followed DJs | User management | **M** |
| **FR-18** | Users should be able to visualize a rank list with the most popular playlists | Playlist management | **M** |
| **FR-19** | Users should be able to add a playlist to their library after a successful payment | Playlist management | **M** |
| **FR-20** | Users should be able to edit their account information | User management | **M** |

# REQUIREMENT TRACEABILITY MATRIX

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement ID | Requirement Title | Use case ID | Use case description |
| FR-01 | DJs should be able to create an account | **UC-01** | Creating an account as a DJ |
| FR-02 | DJs should be able to log in | **UC-02** | Logging in as a DJ |
| FR-03 | DJs should be able to log out | **UC-03** | Logging out as a DJ |
| FR-04 | DJs should be able to create playlists | **UC-04** | Creating a playlist |
| FR-05 | DJs should be able to visualize playlists | **UC-05** | Visualizing playlists as a DJ |
| FR-06 | DJs should be able to update playlists’ name | **UC-06** | Updating a playlist’s name |
| FR-07 | DJs should be able to add songs to a playlist | **UC-07** | Adding a song to a playlist |
| FR-08 | DJs should be able to remove songs from a playlist | **UC-08** | Removing a song from a playlist |
| FR-09 | DJs should be able to delete a playlist | **UC-09** | Deleting a playlist |
| FR-10 | DJs should be able to change their account information | **UC-10** | Changing account information as a DJ |
| FR-11 | Users should be able to register | **UC-11** | Registering as a regular user |
| FR-12 | Users should be able to log in | **UC-12** | Logging in as a regular user |
| FR-13 | Users should be able to log out | **UC-13** | Logging out as a regular user |
| FR-14 | Users should be able to visualize playlists | **UC-14** | Visualizing playlists as a regular user |
| FR-15 | Users should be able to follow a DJ | **UC-15** | Following a DJ |
| FR-16 | Users should be able to start a playlist after paying for it | **UC-16** | Start a playlist as a regular user |
| FR-17 | Users should be able to see a rank list for the most followed DJs | **UC-17** | Visualizing a rank list for the most followed DJs |
| FR-18 | Users should be able to visualize a rank list for the most popular playlists | **UC-18** | Visualizing a rank list for the most popular playlists |
| FR-19 | Users should be able to add a playlist to their library after a successful payment | **UC-19** | Adding a playlist to a regular user’s library |
| FR-20 | Users should be able to edit their account information | **UC-20** | Editing account information as a regular user |

# USE CASES

* **UC-01:** Creating an account as a DJ

Actor: DJ

Main Success Scenario:

1. A DJ starts the web application
2. System requests for the required information (first name, last name, email, password)
3. A DJ fills in the fields
4. System checks if the provided data is authentic
5. System proceeds to the login page *{Use case ends.}*

Extensions:

3a: The DJ didn’t fill all of the required information:

1: System displays a missing information message

2: *{Use case resumes at MSS step 2}*

4a: The provided data isn’t authentic

1: System displays an incorrect provided data message

2: *{Use case resumes at MSS step 2}*

* **UC-02:** Logging in as a DJ

Actor: DJ

Pre-condition: DJ should have an account

Main Success Scenario:

1. The DJ starts the software system
2. System requests email and password
3. DJ fills in the fields
4. System checks if the credentials are correct
5. System proceeds to the home page *{Use case ends.}*

Extensions:

3a: DJ didn’t fill all of the required information:

1: System displays a missing password/email information message

2: *{Use case resumes at MSS step 2}*

4a: The provided credentials are incorrect

1: System displays a wrong credentials message

2: *{Use case resumes at MSS step 2}*

* **UC-03:** Logging out as a DJ

Actor: DJ

Pre-condition: DJ must be logged in

Main Success Scenario:

1. DJ proceeds to log out
2. System displays logging in page *{Use case ends.}*

* **UC-04:** Creating a playlist

Actor: DJ

Pre-condition: DJ must be logged in

Main Success Scenario:

1. DJ goes to the feature for creating a playlist
2. System requests information about privacy type, playlist name
3. DJ fills in playlist’s information
4. System shows a preview of the entered data and asks for confirmation
5. DJ confirms the data was correctly entered
6. System shows confirmation that the playlist was successfully created and updates the data in the data base. *{Use case ends.}*

Extensions:

3a: DJ didn’t fill all of the required information:

1: System displays a missing playlist information message

2: *{Use case resumes at MSS step 2}*

5a: DJ doesn’t confirm that the data was correctly entered

1: *{Use case resumes at MSS step 3}*

* **UC-05:** Visualizing playlists as a DJ

Actor: DJ

Pre-condition: DJ must be logged in &

UC-04 must be completed

Main Success Scenario:

1. DJ goes to the home page
2. System displays data only for his playlists *{Use case ends.}*

* **UC-06:** Updating a playlist’s name

Actor: DJ

Pre-condition: DJ must be logged in &

UC-04 must be completed

Main Success Scenario:

1. DJ requests to edit playlist’s name
2. System displays current playlist’s name and requests a new one
3. DJ enters changes
4. System requests confirmation for the changes
5. DJ confirms that the entered data was correct
6. System shows confirmation that the playlist’s name was successfully updated *{Use case ends.}*

Extensions:

3a: DJ didn’t provide a new name:

1: System displays a missing information message

2: *{Use case ends.}*

5a: DJ doesn’t confirm that the data was correctly entered

1: *{Use case resumes at MSS step 3}*

* **UC-07:** Adding a song to a playlist

Actor: DJ

Pre-condition: DJ must be logged in &

UC-04 must be completed

Main Success Scenario:

1. DJ searches for a song
2. DJ proceeds to add the new song to a playlist
3. System visualizes the already existing DJ’s playlists
4. System asks the DJ to choose a playlist
5. DJ chooses the playlist to which he wants to add a song
6. System adds the song to the specified playlist *{Use case ends.}*

Extensions:

3a: DJ has no existing playlists yet

1: System displays message for empty playlist library

2: *{Use case ends.}*

* **UC-08:** Removing a song from a playlist

Actor: DJ

Pre-condition: DJ must be logged in &

UC-04 and UC-07 must be completed

Main Success Scenario:

1. DJ attempts to remove a song from a playlist
2. System requests confirmation
3. DJ confirms that he wants to remove the song from a specific playlist
4. System displays a success message *{Use case ends.}*

Extensions:

3a: DJ doesn’t confirm that he wants to remove the song

1: *{Use case resumes at MSS step 2}*

* **UC-09:** Deleting a playlist

Actor: DJ

Pre-condition: DJ must be logged in &

UC-04 must be completed

Main Success Scenario:

1. DJ attempts to delete a specific playlist
2. System requests confirmation
3. DJ confirms that he wants to delete the specific playlist
4. System displays a success message *{Use case ends.}*

Extensions:

3a: DJ doesn’t confirm that he wants to delete the playlist

1: *{Use case resumes at MSS step 2}*

* **UC-10:** Changing account information as a DJ

Actor: DJ

Pre-condition: UC-01 must be completed &

DJ must be logged in

Main Success Scenario:

1. DJ attempts to update his account information
2. System asks for new information
3. DJ provides the new account information (email, name or password)
4. DJ clicks a button to confirm the changes
5. System displays a success message. *{Use case ends.}*

Extensions:

4a: DJ attempts to confirm the changes without providing new information

1: System displays a missing new information message

2: *{Use case ends.}*