**User Requirements**

***HeardIT***

|  |
| --- |
| **Date : 22/06/2024** |
| **Author : Mihail Vasilev** |

Table of Contents

[**Introduction** 3](#_Toc169969846)

[**Non-functional requirements** 3](#_Toc169969847)

[**Functional requirements** 4](#_Toc169969848)

[**User stories** 5](#_Toc169969849)

[USC-1 Song streaming 5](#_Toc169969850)

[USC-2 Song uploading 5](#_Toc169969851)

[USC-3 Register 6](#_Toc169969852)

[USC-4 Login 6](#_Toc169969853)

[USC-5 Logout 6](#_Toc169969854)

[USC-6 Create comments on songs 7](#_Toc169969855)

[USC-7 Delete account 7](#_Toc169969856)

[**Additional remarks** 7](#_Toc169969857)

# **Introduction**

The purpose of this document is to outline the user requirements of the HeardIT application. Establishing these requirements is an essential part of the creation of the project. This assignment follows the Agile methodology which implies that during the course of the development process, new requirements and user stories can be included.

The requirements are split into two main categories:

* Non-functional requirements
* Functional requirements

In this document I will establish both of categories of requirements and I will set the priority of each of the Functional requirements which will correlate with when each feature will be implemented.

# **Non-functional requirements**

1. Performance
   * Response time: the system should response to user interactions smoothly and without a significant delay
   * Scalability: the platform should be able to handle a growing number of users and content uploads
   * Load Handling: the application should be able to handle large amounts of concurrent loads without significant changes in performance
2. Reliability
   * Uptime: the platform should have a 99.9% uptime to ensure reliable access from the users
   * Fault tolerance: the system should ensure minimal disruptions to its functionalities
   * Data management: data should be stored reliably while being easy to retrieve
3. Security
   * Data encryption: data should be stored safely to prevent any data leaks and data losses
   * Authorization and authentication: secure authentication mechanisms should be implemented to verify users and prevent misuse
4. Scalability
   * Database scalability: the database should scale accordingly to the increase in stored data
   * User number growth: the application should handle an increasing number of users and their interactions
5. Usability
   * UI design: the application should have an intuitive, pleasing and user-friendly interface that promotes the use of the application
6. Maintainability
   * Code maintainability: codebase is well documented, modular and follows the best coding practices
   * Scalable architecture: system architecture should allow for easy maintainability and integration of new features
7. Testing:
   * Comprehensive testing: thorough testing on all features to ensure the quality of the application
   * Load testing: load testing should be conducted to ensure the platform performs under various user loads

# **Functional requirements**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Requirements** | **Priority score** |
| **Song streaming** | * Users should have control over playback options such as play, pause, skip, and volume control. | **1** |
| **Account management** | * Users should be able to create a new account with unique credentials. * Users should be able to update their account information, including username, email, and password. * The system should provide secure login and logout functionality. * Different user types (creator, fan) should have distinct access levels and permissions. | **1** |
| **Upload/Delete tracks** | * Users should be able to upload tracks to the platform. * Uploaded tracks should be stored securely and be accessible for streaming. * Users should have the ability to edit/delete tracks they have uploaded. | **1** |
| **Upload extra information** | * Users should be able to upload tracks to the platform. * Uploaded tracks should be stored securely and be accessible for streaming. * Users should have the ability to edit/delete tracks they have uploaded. | **2** |
| **Advanced search** | * Users should have the ability to perform advanced searches based on various criteria such as artist, genre, album, and keywords. * The search functionality should provide relevant and accurate results quickly. | **2** |
| **Comments section and likes** | * Users should be able to comment on tracks and interact with other users' comments. * Users should be able to like tracks, and the system should display the number of likes for each track. | **3** |
| **Playlist creation** | * Users should be able to create personalized playlists by adding tracks from the platform. * There should be options to name playlists, reorder tracks, and delete playlists. * Users should have the ability to share playlists with other users. | **3** |

In the table above I have established the functional requirements based on each feature. These features have a priority score attached to them. This priority score correlates to the importance of each feature. Depending on that score, I will implement them at different stages of the project. The lower the score, the sooner the feature will be created. For example, I will start with the features with score ***1***. Once they are implemented, I will continue with the features with score ***2*** and so on.

# **User stories**

In this section I will establish the user stories that correspond to each feature and functionality. These user stories will allow me to better establish the intricacies of the functionality of HeardIT.

This part of the document will be continuously updated during the development process.

## USC-1 Song streaming

**Title**

* As a user, I want to have control over playback options such as play, pause, skip, and volume control, so that I can customize my listening experience.

**Acceptance criteria**

* User can start the song
* User can pause the song
* User can skip the song
* User can increase and decrease the volume

**Acceptance tests**

* Manual testing
* Unit testing where appropriate

**Risks and dependencies**

* Dependent on back-end services being available

## USC-2 Song uploading

**Title**

* As a user, I want to be able to upload songs, so that I can share my music.

**Acceptance criteria**

* User can upload a song

**Acceptance tests**

* Manual testing
* Unit testing where appropriate

**Risks and dependencies**

* Dependent on back-end services being available

## USC-3 Register

**Title**

* As a user, I want to be able to register to the HeardIT platform, so that I can have an account on the platform.

**Acceptance criteria**

* User can create an account to HeardIT.

**Acceptance tests**

* Manual testing

**Risks and dependencies**

* Dependent on back-end services being available
* Depends on Auth0 being available

## USC-4 Login

**Title**

* As a user, I want to be able to login to my account so that I can access my data on HeardIT.

**Acceptance criteria**

* User can login into HeardIT.

**Acceptance tests**

* Manual testing

**Risks and dependencies**

* Dependent on back-end services being available
* Depends on Auth0 being available

## USC-5 Logout

**Title**

* As a user, I want to be able to logout of my account.

**Acceptance criteria**

* User can logout from HeardIT.

**Acceptance tests**

* Manual testing

**Risks and dependencies**

* Dependent on back-end services being available
* Depends on Auth0 being available

## USC-6 Create comments on songs

**Title**

* As a user, I want to be able to create comments for a specific song.

**Acceptance criteria**

* User can create comments for songs.

**Acceptance tests**

* Manual testing
* Unit testing where appropriate

**Risks and dependencies**

* Dependent on back-end services being available
* User needs to be logged in order to create comments.

## USC-7 Delete account

**Title**

* As a user, I want to be able to delete my account from HeardIT so that I can delete all of my information from the platform.

**Acceptance criteria**

* User can delete their account and their uploads and comments.

**Acceptance tests**

* Manual testing
* Unit testing where appropriate

**Risks and dependencies**

* Dependent on back-end services being available
* User needs to be logged in order to create comments.

# **Additional remarks**

Due to the nature of this project and the time constraints, I might not be able to finish every feature within the deadlines of this semester. The main reason for this is that the focus of the assignment is to create a project that mainly applies to the Non-functional requirements. This is a project that I would want to work on even after the semester has ended and as such, I decided to include my ideas in this document.