## Chapter 3: Analysis

### 3.1 Introduction

This project aims to reduce the time this company takes to prepare and do the distribution and to correctly create a custom validated report for each artist/producer. Currently the problem they have showed is that, some of the reports come late as there are a lot of songs to process and the records are constantly changing because every day these songs are played, there are errors made by the company’s workers i.e. Some artist/producers don’t get paid properly, calculating the total amount of time played wrong, etc.

The solution this project offers is an interface that manages a database that contains all the information contained of all the records the company needs. This interface will offer some essential functionality.

More functionality will come as the software develops, an idea for this would be store artist/producer emails and as soon as the process is finished they will receive their report of how much they have earned in royalties.

### 3.2 SWOT Analysis

SWOT analysis was conducted to identify key features which could be incorporated into the creation of this system that will help the company improve its performance.

|  |  |
| --- | --- |
| **Strengths**   * Reduce the amount of time taken to finish the process * Bespoke program for the company’s records * Can be used in any operating system that has java on it * Better level of controlling the information on the system | **Weaknesses**   * Staff might need training to use the program * A high standard implementation of the system can be expensive |
| **Opportunities**   * Creating a new system that can be used to manage databases * Allow company to keep a more manageable record keeping framework | **Threats**   * Company will not develop as consequence of not adapting to new technologies * Increase the amount of changes in the projects once feedback is given |

### 3.2.1 SWOT Analysis before build

This SWOT analysis was carried out so a better understanding of what features and improvements need to be integrated in the new system.

### 3.3 Functional and Non-Functional Requirements

After identifying what features need to be added using the SWOT analysis, a list of functional and non-functional requirements was created as considered crucial to the development of the new system that will be incorporated.

### 3.3.1 Functional Requirements

1. Store information
2. Edit information
3. Produce artist Report
4. Search artist information
5. Calculate total
6. Validate entries
7. Data processing
8. Create connection to interface

### 3.3.2 Non-Functional Requirements

1. Maintainability of the System
2. Response time of the system
3. Storage capacity
4. Cost
5. User friendly interface
6. Extendibility of the system
7. Security

### 3.4 MoSCoW Analysis

It is a very valuable technique that this project uses as a tool to manage the resources and time to be focused into the most important aspects of it i.e. main features. At the same time being consulted with the stakeholders

MoSCoW analysis divides its requirements into four categories: Must, Should, Could and Wont. A brief description of the categories are as follows.

* Must: Describes a requirement that must be satisfied in the final solution for the solution to be considered a success.
* Should: Represents a requirement a high-priority item that should be included in the solution if it is possible. This is often a critical requirement but one which can be satisfied in other ways if strictly necessary.
* Could: Describes a requirement, which is considered desirable but not necessary. This will be included if time and resources permit.
* Won’t: Represents a requirement that stakeholders have agreed will not be implemented in a given release, but may be considered in the future.

### 3.4.1 Functional requirements in MoSCoW

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement ID | Function (Verb noun) | Requirements | Prioritisation |
| 1 | Create Artist | User can create artist record | MUST HAVE |
| 2 | Delete Artist | User can delete artist record | MUST HAVE |
| 3 | Update Artist | User can update artist record | MUST HAVE |
| 4 | Validate Artist | System confirms with user to add record | SHOULD HAVE |
| 5 | Data Processing | Create a csv file with the records | SHOULD HAVE |
| 6 | Search Artist | Search artist by either filtering, by ID, name or type | MUST HAVE |
| 7 | Produce Report | Create a report for artist so they are able to see how much money they made per song | MUST HAVE |
| 8 | Create Song | User can create a song record | MUST HAVE |
| 9 | Delete Song | User can delete a song record | MUST HAVE |
| 10 | Update song | User can update a song record | COULD HAVE |
| 11 | Assign song | User can link song to Artists | MUST HAVE |
| 12 | Calculate Royalty | A calculation of the royalties for each Artist are made by the program | MUST HAVE |
| 13 | Search Song | Search a particular song record | COULD HAVE |
| 14 | Display result | User is able to see if the action has been successful | COULD HAVE |

### 3.5 Use Case Diagram

### Figure 3.5.1



### 3.6 Risk Analysis

Any risks are identified here. These risks were considered as they might occur during the use of the new system (see figure 3.4.1)]

### Figure 3.5.1 – Risk Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| Risk number | Risk name | Impact | Action |
| 1 | Data loss from database | Medium | A backup of the system regularly is required |
| 2 | System bugs | Low | Providing a manual for the user and a contact number of the system creator |
| 3 | Fire hazards | High | Must ensure system to not be over heating and follow manufacture advice and keep a record and monitoring the system regularly |
| 4 | New Updates or change the system | Low | Ensure to have up to date back up also make sure that the new updates are compatible with the existing system |