

# Module Interface Specification for Software Engineering

Team 8 – Rhythm Rangers

Ansel Chen

Muhammad Jawad

Mohamad-Hassan Bahsoun

Matthew Baleanu

Ahmed Al-Hayali

January 17, 2025

# 1 Revision History

| Date   | Version | Notes |
|--------|---------|-------|
| Date 1 | 1.0     | Notes |
| Date 2 | 1.1     | Notes |

## 2 Symbols, Abbreviations and Acronyms

See SRS Documentation at [\[give url —SS\]](#)

[\[Also add any additional symbols, abbreviations or acronyms —SS\]](#)

# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Revision History</b>                    | <b>i</b>  |
| <b>2</b> | <b>Symbols, Abbreviations and Acronyms</b> | <b>ii</b> |
| <b>3</b> | <b>Introduction</b>                        | <b>1</b>  |
| <b>4</b> | <b>Notation</b>                            | <b>1</b>  |
| <b>5</b> | <b>Module Decomposition</b>                | <b>1</b>  |
| <b>6</b> | <b>GUI Module</b>                          | <b>3</b>  |
| 6.1      | GUI Module . . . . .                       | 3         |
| 6.2      | Uses . . . . .                             | 3         |
| 6.3      | Syntax . . . . .                           | 3         |
| 6.3.1    | Exported Constants . . . . .               | 3         |
| 6.3.2    | Exported Access Programs . . . . .         | 3         |
| 6.4      | Semantics . . . . .                        | 3         |
| 6.4.1    | State Variables . . . . .                  | 3         |
| 6.4.2    | Environment Variables . . . . .            | 3         |
| 6.4.3    | Assumptions . . . . .                      | 3         |
| 6.4.4    | Access Routine Semantics . . . . .         | 3         |
| 6.4.5    | Local Functions . . . . .                  | 4         |
| <b>7</b> | <b>MIS of Audio File Input Module</b>      | <b>4</b>  |
| 7.1      | Audio File Input Module . . . . .          | 4         |
| 7.2      | Uses . . . . .                             | 4         |
| 7.3      | Syntax . . . . .                           | 4         |
| 7.3.1    | Exported Constants . . . . .               | 4         |
| 7.3.2    | Exported Access Programs . . . . .         | 4         |
| 7.4      | Semantics . . . . .                        | 4         |
| 7.4.1    | State Variables . . . . .                  | 4         |
| 7.4.2    | Environment Variables . . . . .            | 4         |
| 7.4.3    | Assumptions . . . . .                      | 5         |
| 7.4.4    | Access Routine Semantics . . . . .         | 5         |
| 7.4.5    | Local Functions . . . . .                  | 5         |
| <b>8</b> | <b>MIS of Search Query Module</b>          | <b>5</b>  |
| 8.1      | Search Query Module . . . . .              | 5         |
| 8.2      | Uses . . . . .                             | 5         |
| 8.3      | Syntax . . . . .                           | 5         |
| 8.3.1    | Exported Constants . . . . .               | 5         |
| 8.3.2    | Exported Access Programs . . . . .         | 6         |

|           |   |          |
|-----------|---|----------|
| 8.4       | Semantics . . . . .                       | 6        |
| 8.4.1     | State Variables . . . . .                 | 6        |
| 8.4.2     | Environment Variables . . . . .           | 6        |
| 8.4.3     | Assumptions . . . . .                     | 6        |
| 8.4.4     | Access Routine Semantics . . . . .        | 6        |
| 8.4.5     | Local Functions . . . . .                 | 6        |
| <b>9</b>  | <b>MIS of Client Communication Module</b> | <b>6</b> |
| 9.1       | Client Communication Module . . . . .     | 6        |
| 9.2       | Uses . . . . .                            | 6        |
| 9.3       | Syntax . . . . .                          | 6        |
| 9.3.1     | Exported Constants . . . . .              | 6        |
| 9.3.2     | Exported Access Programs . . . . .        | 7        |
| 9.4       | Semantics . . . . .                       | 7        |
| 9.4.1     | State Variables . . . . .                 | 7        |
| 9.4.2     | Environment Variables . . . . .           | 7        |
| 9.4.3     | Assumptions . . . . .                     | 7        |
| 9.4.4     | Access Routine Semantics . . . . .        | 7        |
| 9.4.5     | Local Functions . . . . .                 | 7        |
| <b>10</b> | <b>MIS of Server Communication Module</b> | <b>8</b> |
| 10.1      | Server Communication Module . . . . .     | 8        |
| 10.2      | Uses . . . . .                            | 8        |
| 10.3      | Syntax . . . . .                          | 8        |
| 10.3.1    | Exported Constants . . . . .              | 8        |
| 10.3.2    | Exported Access Programs . . . . .        | 8        |
| 10.4      | Semantics . . . . .                       | 8        |
| 10.4.1    | State Variables . . . . .                 | 8        |
| 10.4.2    | Environment Variables . . . . .           | 8        |
| 10.4.3    | Assumptions . . . . .                     | 8        |
| 10.4.4    | Access Routine Semantics . . . . .        | 9        |
| 10.4.5    | Local Functions . . . . .                 | 9        |
| <b>11</b> | <b>MIS of Driver Module</b>               | <b>9</b> |
| 11.1      | Driver Module . . . . .                   | 9        |
| 11.2      | Uses . . . . .                            | 9        |
| 11.3      | Syntax . . . . .                          | 9        |
| 11.3.1    | Exported Constants . . . . .              | 9        |
| 11.3.2    | Exported Access Programs . . . . .        | 10       |
| 11.4      | Semantics . . . . .                       | 10       |
| 11.4.1    | State Variables . . . . .                 | 10       |
| 11.4.2    | Environment Variables . . . . .           | 10       |
| 11.4.3    | Assumptions . . . . .                     | 10       |

|           |   |           |
|-----------|---|-----------|
| 11.4.4    | Access Routine Semantics . . . . .                      | 10        |
| 11.4.5    | Local Functions . . . . .                               | 10        |
| <b>12</b> | <b>MIS of Tempo (BPM) Feature Extraction Module</b>     | <b>11</b> |
| 12.1      | Tempo (BPM) Feature Extraction Module . . . . .         | 11        |
| 12.2      | Uses . . . . .  | 11        |
| 12.3      | Syntax . . . . .  | 11        |
| 12.3.1    | Exported Constants . . . . .                            | 11        |
| 12.3.2    | Exported Access Programs . . . . .                      | 11        |
| 12.4      | Semantics . . . . .                                     | 11        |
| 12.4.1    | State Variables . . . . .                               | 11        |
| 12.4.2    | Environment Variables . . . . .                         | 11        |
| 12.4.3    | Assumptions . . . . .                                   | 11        |
| 12.4.4    | Access Routine Semantics . . . . .                      | 11        |
| 12.4.5    | Local Functions . . . . .                               | 11        |
| <b>13</b> | <b>MIS of Key and Scale Feature Extraction Module</b>   | <b>12</b> |
| 13.1      | Key and Scale Feature Extraction Module . . . . .       | 12        |
| 13.2      | Uses . . . . .  | 12        |
| 13.3      | Syntax . . . . .  | 12        |
| 13.3.1    | Exported Constants . . . . .                            | 12        |
| 13.3.2    | Exported Access Programs . . . . .                      | 12        |
| 13.4      | Semantics . . . . .                                     | 12        |
| 13.4.1    | State Variables . . . . .                               | 12        |
| 13.4.2    | Environment Variables . . . . .                         | 12        |
| 13.4.3    | Assumptions . . . . .                                   | 12        |
| 13.4.4    | Access Routine Semantics . . . . .                      | 12        |
| 13.4.5    | Local Functions . . . . .                               | 13        |
| <b>14</b> | <b>MIS of Instrument Type Feature Extraction Module</b> | <b>13</b> |
| 14.1      | Instrument Type Feature Extraction Module . . . . .     | 13        |
| 14.2      | Uses . . . . .  | 13        |
| 14.3      | Syntax . . . . .  | 13        |
| 14.3.1    | Exported Constants . . . . .                            | 13        |
| 14.3.2    | Exported Access Programs . . . . .                      | 13        |
| 14.4      | Semantics . . . . .                                     | 13        |
| 14.4.1    | State Variables . . . . .                               | 13        |
| 14.4.2    | Environment Variables . . . . .                         | 13        |
| 14.4.3    | Assumptions . . . . .                                   | 13        |
| 14.4.4    | Access Routine Semantics . . . . .                      | 13        |
| 14.4.5    | Local Functions . . . . .                               | 14        |

|  |           |
|--|-----------|
| <b>15 MIS of Vocal Gender Feature Extraction Module</b>      | <b>14</b> |
| 15.1 MIS of Vocal Gender Feature Extraction Module . . . . . | 14        |
| 15.2 Uses . . . . .  | 14        |
| 15.3 Syntax . . . . .  | 14        |
| 15.3.1 Exported Constants . . . . .                          | 14        |
| 15.3.2 Exported Access Programs . . . . .                    | 14        |
| 15.4 Semantics . . . . .                                     | 14        |
| 15.4.1 State Variables . . . . .                             | 14        |
| 15.4.2 Environment Variables . . . . .                       | 14        |
| 15.4.3 Assumptions . . . . .                                 | 15        |
| 15.4.4 Access Routine Semantics . . . . .                    | 15        |
| 15.4.5 Local Functions . . . . .                             | 15        |
| <b>16 MIS of Dynamic Range Feature Extraction Module</b>     | <b>15</b> |
| 16.1 Dynamic Range Feature Extraction Module . . . . .       | 15        |
| 16.2 Uses . . . . .  | 15        |
| 16.3 Syntax . . . . .  | 15        |
| 16.3.1 Exported Constants . . . . .                          | 15        |
| 16.3.2 Exported Access Programs . . . . .                    | 15        |
| 16.4 Semantics . . . . .                                     | 15        |
| 16.4.1 State Variables . . . . .                             | 15        |
| 16.4.2 Environment Variables . . . . .                       | 16        |
| 16.4.3 Assumptions . . . . .                                 | 16        |
| 16.4.4 Access Routine Semantics . . . . .                    | 16        |
| 16.4.5 Local Functions . . . . .                             | 16        |
| <b>17 MIS of Instrumentalness Feature Extraction Module</b>  | <b>16</b> |
| 17.1 Instrumentalness Feature Extraction Module . . . . .    | 16        |
| 17.2 Uses . . . . .  | 16        |
| 17.3 Syntax . . . . .  | 16        |
| 17.3.1 Exported Constants . . . . .                          | 16        |
| 17.3.2 Exported Access Programs . . . . .                    | 17        |
| 17.4 Semantics . . . . .                                     | 17        |
| 17.4.1 State Variables . . . . .                             | 17        |
| 17.4.2 Environment Variables . . . . .                       | 17        |
| 17.4.3 Assumptions . . . . .                                 | 17        |
| 17.4.4 Access Routine Semantics . . . . .                    | 17        |
| 17.4.5 Local Functions . . . . .                             | 17        |
| <b>18 MIS of Contour Feature Extraction Module</b>           | <b>17</b> |
| 18.1 Contour Feature Extraction Module . . . . .             | 17        |
| 18.2 Uses . . . . .  | 17        |
| 18.3 Syntax . . . . .  | 18        |

|           |  |           |
|-----------|--|-----------|
| 18.3.1    | Exported Constants                             | 18        |
| 18.3.2    | Exported Access Programs                       | 18        |
| 18.4      | Semantics                                      | 18        |
| 18.4.1    | State Variables                                | 18        |
| 18.4.2    | Environment Variables                          | 18        |
| 18.4.3    | Assumptions                                    | 18        |
| 18.4.4    | Access Routine Semantics                       | 18        |
| 18.4.5    | Local Functions                                | 18        |
| <b>19</b> | <b>MIS of Mood Feature Extraction Module</b>   | <b>18</b> |
| 19.1      | Mood Feature Extraction Module                 | 18        |
| 19.2      | Uses   | 18        |
| 19.3      | Syntax   | 19        |
| 19.3.1    | Exported Constants                             | 19        |
| 19.3.2    | Exported Access Programs                       | 19        |
| 19.4      | Semantics                                      | 19        |
| 19.4.1    | State Variables                                | 19        |
| 19.4.2    | Environment Variables                          | 19        |
| 19.4.3    | Assumptions                                    | 19        |
| 19.4.4    | Access Routine Semantics                       | 19        |
| 19.4.5    | Local Functions                                | 19        |
| <b>20</b> | <b>MIS of Recommendation Module</b>            | <b>19</b> |
| 20.1      | Recommendation Module                          | 19        |
| 20.2      | Uses   | 19        |
| 20.3      | Syntax   | 20        |
| 20.3.1    | Exported Constants                             | 20        |
| 20.3.2    | Exported Access Programs                       | 20        |
| 20.4      | Semantics                                      | 20        |
| 20.4.1    | State Variables                                | 20        |
| 20.4.2    | Environment Variables                          | 20        |
| 20.4.3    | Assumptions                                    | 20        |
| 20.4.4    | Access Routine Semantics                       | 21        |
| 20.4.5    | Local Functions                                | 21        |
| <b>21</b> | <b>MIS of Program Results Interface Module</b> | <b>21</b> |
| 21.1      | Program Results Interface Module               | 21        |
| 21.2      | Uses   | 21        |
| 21.3      | Syntax   | 21        |
| 21.3.1    | Exported Constants                             | 21        |
| 21.3.2    | Exported Access Programs                       | 21        |
| 21.4      | Semantics                                      | 21        |
| 21.4.1    | State Variables                                | 21        |



|           |                                    |           |
|-----------|------------------------------------|-----------|
| 21.4.2    | Environment Variables . . . . .    | 21        |
| 21.4.3    | Assumptions . . . . .              | 22        |
| 21.4.4    | Access Routine Semantics . . . . . | 22        |
| 21.4.5    | Local Functions . . . . .          | 22        |
| <b>22</b> | <b>Appendix</b>                    | <b>24</b> |

### 3 Introduction

The following document details the Module Interface Specifications for [Fill in your project name and description —SS]

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at .... [provide the url for your repo —SS]

### 4 Notation

[You should describe your notation. You can use what is below as a starting point. —SS]

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol  $:=$  is used for a multiple assignment statement and conditional rules follow the form  $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \dots | c_n \Rightarrow r_n)$ .

The following table summarizes the primitive data types used by Software Engineering.

| Data Type      | Notation     | Description   |
|----------------|--------------|---|
| character      | char         | a single symbol or digit  |
| integer        | $\mathbb{Z}$ | a number without a fractional component<br>in $(-\infty, \infty)$ |
| natural number | $\mathbb{N}$ | a number without a fractional component<br>in $[1, \infty)$       |
| real           | $\mathbb{R}$ | any number in $(-\infty, \infty)$                                 |

The specification of Software Engineering uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Software Engineering uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

### 5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

| Level 1           | Level 2  |
|-------------------|--|
| Hardware-Hiding   |  |
| Behaviour-Hiding  | GUI Module<br>Audio File Input Module<br>Search Query Module<br>Client Communication Module<br>Server Communication Module<br>Driver Module<br>Tempo (BPM) Feature Extraction Module<br>Key and Scale Feature Extraction Module<br>Instrument Type Feature Extraction Module<br>Vocal Gender Feature Extraction Module<br>Dynamic Range Feature Extraction Module<br>Instrumentalness Feature Extraction Module<br>Contour Feature Extraction Module<br>Mood Feature Extraction Module<br>Recommendation Module<br>Program Results Interface |
| Software Decision | Database<br>Spotify API<br>Deezer API<br>Genre Feature Module  |

Table 1: Module Hierarchy

## 6 GUI Module

### 6.1 GUI Module

### 6.2 Uses

- First-Match Text Field Input Module
- URL Input module
- Audio File Input Module
- Spotify Query Search & Select

### 6.3 Syntax

#### 6.3.1 Exported Constants

N/A

#### 6.3.2 Exported Access Programs

| Name                  | In   | Out                                      | Exceptions |
|-----------------------|--|--|------------|
| Consolidate<br>Inputs | Up to 4 collection(s)<br>of reference(s)<br>track(s) | Merged collection of<br>track references | -          |

### 6.4 Semantics

#### 6.4.1 State Variables

- Data type of the collection of track reference(s)

#### 6.4.2 Environment Variables

N/A

#### 6.4.3 Assumptions

N/A

#### 6.4.4 Access Routine Semantics

consolidate\_inputs():

- output: parses the user input and returns the songs that are sent to be processed

### 6.4.5 Local Functions

- `parse_wav_file(file)`

—

- `parse_url(url)`

—

- `parse_text(text)`

—

## 7 MIS of Audio File Input Module

### 7.1 Audio File Input Module

User inputs an audio file to the system to analyze.

### 7.2 Uses

N/A

### 7.3 Syntax

#### 7.3.1 Exported Constants

N/A

#### 7.3.2 Exported Access Programs

| Name                  | In         | Out                             | Exceptions        |
|-----------------------|------------|---------------------------------|-------------------|
| On Input Button Press | Audio File | Collection of song reference(s) | Invalid File Type |

### 7.4 Semantics

#### 7.4.1 State Variables

- Collection of track reference(s)

#### 7.4.2 Environment Variables

N/A

### 7.4.3 Assumptions

- User has a properly named Audio File.
- User audio file input is actually a song.

### 7.4.4 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

[A module without environment variables or state variables is unlikely to have a state transition. In this case a state transition can only occur if the module is changing the state of another module. —SS]

[Modules rarely have both a transition and an output. In most cases you will have one or the other. —SS]

### 7.4.5 Local Functions

[As appropriate —SS] [These functions are for the purpose of specification. They are not necessarily something that is going to be implemented explicitly. Even if they are implemented, they are not exported; they only have local scope. —SS]

## 8 MIS of Search Query Module

### 8.1 Search Query Module

searchQuery

### 8.2 Uses

- GUI Module
- Client Communication Module

### 8.3 Syntax

#### 8.3.1 Exported Constants

N/A

### 8.3.2 Exported Access Programs

| Name        | In         | Out           | Exceptions |
|-------------|------------|---------------|------------|
| searchQuery | input_text | Spotify Query | -          |

## 8.4 Semantics

### 8.4.1 State Variables

- user\_sq\_input: stores the query being processed

### 8.4.2 Environment Variables

- Spotify Client ID
- Spotify Client Secret

### 8.4.3 Assumptions

N/A

### 8.4.4 Access Routine Semantics

- transition: Takes the text input and/or Spotify ID from the GUI Module, and builds the query to be sent to the Client Communication Module

### 8.4.5 Local Functions

N/A

## 9 MIS of Client Communication Module

### 9.1 Client Communication Module

User inputs a song and that is turned into a spotify search query where the top 10 matches are available for user to select

### 9.2 Uses

N/A

### 9.3 Syntax

#### 9.3.1 Exported Constants

N/A

### 9.3.2 Exported Access Programs

| Name                    | In             | Out                                      | Exceptions |
|-------------------------|----------------|--|------------|
| Search Query Request    | text input     | top 10 matches from spotify query search | N/A        |
| Output result selection | user selection | Collection containing track reference    | N/A        |

## 9.4 Semantics

### 9.4.1 State Variables

- Collection containing track reference

### 9.4.2 Environment Variables

- Spotify Client ID
- Spotify Client Secret

### 9.4.3 Assumptions

N/A

### 9.4.4 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

[A module without environment variables or state variables is unlikely to have a state transition. In this case a state transition can only occur if the module is changing the state of another module. —SS]

[Modules rarely have both a transition and an output. In most cases you will have one or the other. —SS]

### 9.4.5 Local Functions

[As appropriate —SS] [These functions are for the purpose of specification. They are not necessarily something that is going to be implemented explicitly. Even if they are implemented, they are not exported; they only have local scope. —SS]



## 10 MIS of Server Communication Module

### 10.1 Server Communication Module

User inputs a song and that is turned into a spotify search query where the top 10 matches are available for user to select

### 10.2 Uses

N/A

### 10.3 Syntax

#### 10.3.1 Exported Constants

N/A

#### 10.3.2 Exported Access Programs

| Name                    | In             | Out                                      | Exceptions |
|-------------------------|----------------|--|------------|
| Search Query Request    | text input     | top 10 matches from spotify query search | N/A        |
| Output result selection | user selection | Collection containing track reference    | N/A        |

### 10.4 Semantics

#### 10.4.1 State Variables

- Collection containing track reference

#### 10.4.2 Environment Variables

- Spotify Client ID
- Spotify Client Secret

#### 10.4.3 Assumptions

N/A

#### 10.4.4 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

[A module without environment variables or state variables is unlikely to have a state transition. In this case a state transition can only occur if the module is changing the state of another module. —SS]

[Modules rarely have both a transition and an output. In most cases you will have one or the other. —SS]

#### 10.4.5 Local Functions

[As appropriate —SS] [These functions are for the purpose of specification. They are not necessarily something that is going to be implemented explicitly. Even if they are implemented, they are not exported; they only have local scope. —SS]

## 11 MIS of Driver Module

### 11.1 Driver Module

User inputs a song and that is turned into a spotify search query where the top 10 matches are available for user to select

### 11.2 Uses

N/A

### 11.3 Syntax

#### 11.3.1 Exported Constants

N/A

### 11.3.2 Exported Access Programs

| Name                    | In             | Out                                      | Exceptions |
|-------------------------|----------------|--|------------|
| Search Query Request    | text input     | top 10 matches from spotify query search | N/A        |
| Output result selection | user selection | Collection containing track reference    | N/A        |

## 11.4 Semantics

### 11.4.1 State Variables

- Collection containing track reference

### 11.4.2 Environment Variables

- Spotify Client ID
- Spotify Client Secret

### 11.4.3 Assumptions

N/A

### 11.4.4 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

[A module without environment variables or state variables is unlikely to have a state transition. In this case a state transition can only occur if the module is changing the state of another module. —SS]

[Modules rarely have both a transition and an output. In most cases you will have one or the other. —SS]

### 11.4.5 Local Functions

[As appropriate —SS] [These functions are for the purpose of specification. They are not necessarily something that is going to be implemented explicitly. Even if they are implemented, they are not exported; they only have local scope. —SS]

## 12 MIS of Tempo (BPM) Feature Extraction Module

### 12.1 Tempo (BPM) Feature Extraction Module

#### 12.2 Uses

N/A

#### 12.3 Syntax

##### 12.3.1 Exported Constants

N/A

##### 12.3.2 Exported Access Programs

| Name             | In                                | Out                         | Exceptions |
|------------------|-----------------------------------|-----------------------------|------------|
| Extract<br>Tempo | Audio time series<br>(np.ndarray) | Song Tempo $\in \mathbb{R}$ | N/A        |

#### 12.4 Semantics

##### 12.4.1 State Variables

N/A

##### 12.4.2 Environment Variables

N/A

##### 12.4.3 Assumptions

Valid audio file with coherent song information.

##### 12.4.4 Access Routine Semantics

ExtractTempo():

- transition: N/A
- output: `SongTempo := ExtractTempo(Audio_Time_Series)`
- exception: N/A

##### 12.4.5 Local Functions

N/A

## 13 MIS of Key and Scale Feature Extraction Module

### 13.1 Key and Scale Feature Extraction Module

### 13.2 Uses

N/A

### 13.3 Syntax

#### 13.3.1 Exported Constants

N/A

#### 13.3.2 Exported Access Programs

| Name                    | In                                  | Out                                | Exceptions |
|-------------------------|-------------------------------------|------------------------------------|------------|
| Extract<br>Key<br>Scale | Audio time series<br>& (np.ndarray) | Song Key, Scale $\in \mathbb{Z}^2$ | N/A        |

### 13.4 Semantics

#### 13.4.1 State Variables

N/A

#### 13.4.2 Environment Variables

N/A

#### 13.4.3 Assumptions

Valid audio file with coherent song information.

#### 13.4.4 Access Routine Semantics

Extract\_Key\_Scale():

- transition: N/A
- output: SongKey, SongScale := Extract\_Key\_Scale(Audio\_Time\_Series)
- exception: N/A

### 13.4.5 Local Functions

N/A

## 14 MIS of Instrument Type Feature Extraction Module

### 14.1 Instrument Type Feature Extraction Module

#### 14.2 Uses

N/A

### 14.3 Syntax

#### 14.3.1 Exported Constants

N/A

#### 14.3.2 Exported Access Programs

| Name                          | In                                | Out                                | Exceptions |
|-------------------------------|-----------------------------------|------------------------------------|------------|
| Extract<br>Instrument<br>Type | Audio time series<br>(np.ndarray) | Instrument Type $\in \mathbb{Z}^k$ | N/A        |

### 14.4 Semantics

#### 14.4.1 State Variables

N/A

#### 14.4.2 Environment Variables

N/A

#### 14.4.3 Assumptions

Valid audio file with coherent song information.

#### 14.4.4 Access Routine Semantics

Extract\_Instrument\_Type():

- transition: N/A

- output: `InstrumentType := Extract_Instrument_Type(Audio.Time.Series)`
- exception: **N/A**

#### 14.4.5 Local Functions

**N/A**

## 15 MIS of Vocal Gender Feature Extraction Module

### 15.1 MIS of Vocal Gender Feature Extraction Module

This feature seeks to quantify whether the voices features in the inputted audio file are largely more feminine or masculine sounding. This is represented by a float with a range between 0 and 1 where 0 means only "masculine" sound signatures are contained and 1 means only "feminine" sounds, where values in-between represent a blend.

### 15.2 Uses

**N/A**

### 15.3 Syntax

#### 15.3.1 Exported Constants

**N/A**

#### 15.3.2 Exported Access Programs

| Name                    | In   | Out                           | Exceptions |
|-------------------------|--|-------------------------------|------------|
| Extract<br>Vocal Gender | Audio time series<br>( <code>np.ndarray</code> ) | Vocal Gender $\in \mathbb{R}$ | <b>N/A</b> |

### 15.4 Semantics

#### 15.4.1 State Variables

**N/A**

#### 15.4.2 Environment Variables

**N/A**

### 15.4.3 Assumptions

Valid audio file with coherent song information.

### 15.4.4 Access Routine Semantics

`Extract_Vocal_Gender()`:

- transition: N/A
- output: `VocalGender := Extract_Vocal_Gender(Audio_Time_Series)`
- exception: N/A

### 15.4.5 Local Functions

N/A

## 16 MIS of Dynamic Range Feature Extraction Module

### 16.1 Dynamic Range Feature Extraction Module

Feature extracts the range of sounds (difference between peak and trough) of the audio signal.

### 16.2 Uses

N/A

### 16.3 Syntax

#### 16.3.1 Exported Constants

N/A

#### 16.3.2 Exported Access Programs

| Name                        | In   | Out                                       | Exceptions |
|-----------------------------|--|---|------------|
| Extract<br>Dynamic<br>Range | Audio time series<br>( <code>np.ndarray</code> ) | Dynamic Range (decibels) $\in \mathbb{R}$ | N/A        |

### 16.4 Semantics

#### 16.4.1 State Variables

N/A



### 16.4.2 Environment Variables

N/A

### 16.4.3 Assumptions

Valid audio file with coherent song information.

### 16.4.4 Access Routine Semantics

`Extract_Dynamic_Range()`:

- transition: N/A
- output: `DynamicRange := Extract_Dynamic_Range(Audio_Time_Series)`
- exception: N/A

### 16.4.5 Local Functions

N/A

## 17 MIS of Instrumentalness Feature Extraction Module

### 17.1 Instrumentalness Feature Extraction Module

Extracts the how prominent instrumental sounds are within the song. Represented by a float variable where the range is between 0 and 1, where higher values mean more instrumental sounds and lower means less. Eg, 0 would mean an acapella piece of music, 1 would be something that purely features instruments.

### 17.2 Uses

N/A

### 17.3 Syntax

#### 17.3.1 Exported Constants

N/A

### 17.3.2 Exported Access Programs

| Name                     | In   | Out                               | Exceptions |
|--------------------------|--|-----------------------------------|------------|
| Extract Instrumentalness | Audio time series<br>( <code>np.ndarray</code> ) | Instrumentalness $\in \mathbb{R}$ | N/A        |

## 17.4 Semantics

### 17.4.1 State Variables

N/A

### 17.4.2 Environment Variables

N/A

### 17.4.3 Assumptions

Valid audio file with coherent song information.

### 17.4.4 Access Routine Semantics

`Extract_Instrumentalness()`:

- transition: N/A
- output: `Instrumentalness := Extract_Instrumentalness(Audio_Time_Series)`
- exception: N/A

### 17.4.5 Local Functions

N/A

## 18 MIS of Contour Feature Extraction Module

### 18.1 Contour Feature Extraction Module

### 18.2 Uses

N/A

## 18.3 Syntax

### 18.3.1 Exported Constants

N/A

### 18.3.2 Exported Access Programs

| Name    | In                | Out    | Exceptions |
|---------|-------------------|--------|------------|
| Extract | Audio time series | output | N/A        |
| Melodic | (np.ndarray)      |        |            |
| Contour |                   |        |            |

## 18.4 Semantics

### 18.4.1 State Variables

N/A

### 18.4.2 Environment Variables

N/A

### 18.4.3 Assumptions

Valid audio file with coherent song information.

### 18.4.4 Access Routine Semantics

Extract\_Melodic\_Contour():

- transition: N/A
- output: `variable name := Extract_Melodic_Contour(Audio_Time_Series)`
- exception: N/A

### 18.4.5 Local Functions

N/A

## 19 MIS of Mood Feature Extraction Module

### 19.1 Mood Feature Extraction Module

### 19.2 Uses

N/A

## 19.3 Syntax

### 19.3.1 Exported Constants

N/A

### 19.3.2 Exported Access Programs

| Name    | In                | Out                   | Exceptions |
|---------|-------------------|-----------------------|------------|
| Extract | Audio time series | Mood $\in \mathbb{Z}$ | N/A        |
| Mood    | (np.ndarray)      |                       |            |

## 19.4 Semantics

### 19.4.1 State Variables

N/A

### 19.4.2 Environment Variables

N/A

### 19.4.3 Assumptions

Valid audio file with coherent song information.

### 19.4.4 Access Routine Semantics

Extract\_Mood():

- transition: N/A
- output: Mood := Extract\_Mood(Audio\_Time\_Series)
- exception: N/A

### 19.4.5 Local Functions

N/A

## 20 MIS of Recommendation Module

### 20.1 Recommendation Module

### 20.2 Uses

- Tempo (BPM) Feature Extraction Module

- Key and Scale Feature Extraction Module
- Instrument Type Feature Extraction Module
- Vocal Gender Feature Extraction Module
- Dynamic Range Feature Extraction Module
- Instrumentalness Feature Extraction Module
- Contour Feature Extraction Module
- Mood Feature Extraction Module
- Driver Module
- Spotify API

## 20.3 Syntax

### 20.3.1 Exported Constants

N/A

### 20.3.2 Exported Access Programs

| Name                      | In                   | Out                | Exceptions |
|---------------------------|----------------------|--------------------|------------|
| Generate                  | Song_Features        | Rec_Tracks         | N/A        |
| Recom-<br>menda-<br>tions | np.ndarray[Features] | np.ndarray[Tracks] |            |

## 20.4 Semantics

### 20.4.1 State Variables

N/A

### 20.4.2 Environment Variables

N/A

### 20.4.3 Assumptions

N/A

#### 20.4.4 Access Routine Semantics

GenerateRecommendations():

- transition: N/A
- output: Recommended\_Songs := GenerateRecommendations(Song\_Features)
- exception: N/A

#### 20.4.5 Local Functions

N/A

## 21 MIS of Program Results Interface Module

### 21.1 Program Results Interface Module

#### 21.2 Uses

- Recommendation Module

#### 21.3 Syntax

##### 21.3.1 Exported Constants

N/A

##### 21.3.2 Exported Access Programs

| Name                         | In                               | Out          | Exceptions |
|------------------------------|----------------------------------|--------------|------------|
| Generate<br>Spotify<br>Embed | Rec_Tracks<br>np.ndarray[Tracks] | Tracks_Embed | N/A        |

#### 21.4 Semantics

##### 21.4.1 State Variables

N/A

##### 21.4.2 Environment Variables

N/A

### 21.4.3 Assumptions

N/A

### 21.4.4 Access Routine Semantics

$[\text{accessProg} \text{---SS}]()$ :

- transition:  $[\text{if appropriate} \text{---SS}]$
- output:  $[\text{if appropriate} \text{---SS}]$
- exception:  $[\text{if appropriate} \text{---SS}]$

### 21.4.5 Local Functions

N/A

## References

- Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.
- Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL <http://citeseer.ist.psu.edu/428727.html>.



## 22 Appendix

[Extra information if required —SS]

## Appendix — Reflection

[Not required for CAS 741 projects —SS]

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design.

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing “what you think the evaluator wants to hear.”

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. Which of your design decisions stemmed from speaking to your client(s) or a proxy (e.g. your peers, stakeholders, potential users)? For those that were not, why, and where did they come from?
4. While creating the design doc, what parts of your other documents (e.g. requirements, hazard analysis, etc), if any, needed to be changed, and why?
5. What are the limitations of your solution? Put another way, given unlimited resources, what could you do to make the project better? (LO\_ProbSolutions)
6. Give a brief overview of other design solutions you considered. What are the benefits and tradeoffs of those other designs compared with the chosen design? From all the potential options, why did you select the documented design? (LO\_Explores)