CSC 431

Music Therapy

System Architecture Specification (SAS)

Team Number 15

Hiba Farhan	Scrum Master
Cameron VanDyke	Web Developer
Ayesha Bakshi	Web Developer

Version History

Version	Date	Author(s)	Change Comments
1.0.0	March 25, 2022	Hiba Farhan, Ayesha Bakshi, Cameron VanDyke	The first draft of the SAS document was created.

Table of Contents

1. System Analysis 5
    1.1 System Overview 5
    1.2 System Diagram 5
    1.3 Actor Identification 5-6
    1.4 Design Rationale 6
        1.4.1 Architectural Style 6
        1.4.2 Design Pattern(s) 6
        1.4.3 Framework 6
2 Functional Design 7
    2.1 Sequence Diagram 7
3. Structural Design 8
4. Behavioral design 8

Table of Figures

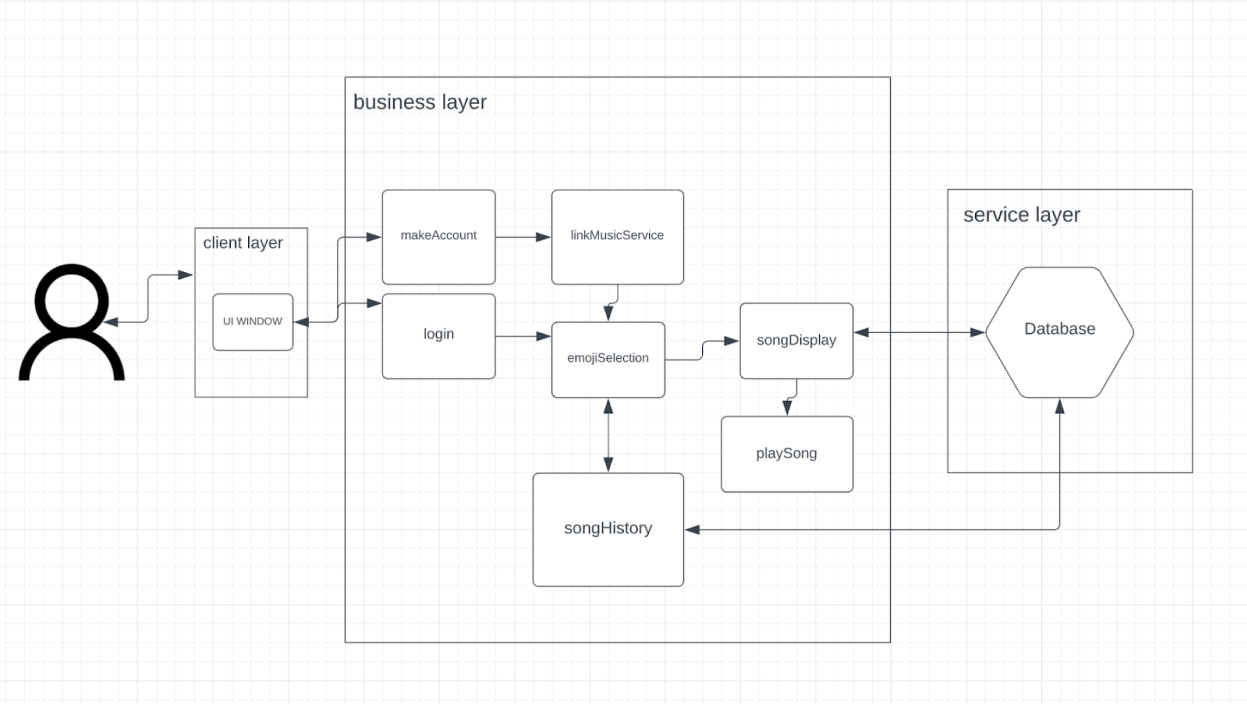
1.2 System Diagram 5
2.1 Sequence Diagram 7
3 Structural Design 8

1. System Analysis

1.1 System Overview

- This document describes the various parts that will be developed in our website titled Music Therapy. This system uses a three-tier architecture that includes the client layer, a business layer, and a service layer. The client server connects the user to the network. The business layer gives the website logic which includes linking your music streaming service, emoji selection (dependent on your mood), giving the user the appropriate song, and finally saving the song history. The service layer will filter through said music streaming service to create a database that stores the songs.
- This system has three essential components. The UI interface is what the user will be interacting with. This layer will connect to the business layer. The business logic layer has components like making your account, selecting and logging into your streaming service, selecting the emoji to then give the user the appropriate song. The song selection and user information will be stored in our database.

1.2 System Diagram



1.3 Actor Identification

1. Users: Users will create or log into their Music Therapy account to generate a song to listen to based on their current mood or view their history of mood and past recommendations. The user initiates the interaction between the user and system and therefore triggers the system to run.
   - New members (Human Users)
   - Registered members (Human Users)

- Web App Developers (Human Users)

2. Database Administrator (External System): Provides a database of users' Spotify account information. The database interacts with the third-party streaming service in order to generate a personalized song recommendation or their account history.
3. Time (System Clock)

1.4 Design Rationale

1.4.1 Architectural Style

Our design will utilize a 3-tier architecture, which consists of a middle tier of business logic between a database-centric and client-server architecture, which fits our system well.

1. User/client layer: User selects the emoji based on their emotion
2. Business layer: Includes the logic for song recommendations.
3. Service layer: Stores data (song recommendation history) and authenticates through third party applications (Spotify and Apple Music).

1.4.2 Design Pattern(s)

1. Factory Method
   - Both the new user and registered user will inherent from a parent which will initialize classes for both users
   - The subclasses that differentiate the new users and the registered users will override the specific initializations to make the two users different
2. Adapter
   - The adapter design would serve to allow communication between the business layer and the service layer in our three-tier architecture style.

1.4.3 Framework

Our team will be making use of the Replit IDE, which gives us access to languages such as HTML, CSS, and Javascript.

Why Replit: Replit works as an easily accessible cloud-based IDE that should allow us to make our website responsive on both mobile and desktop.

Why HTML: We will make use of HTML to structure our website and to write the content on our website.

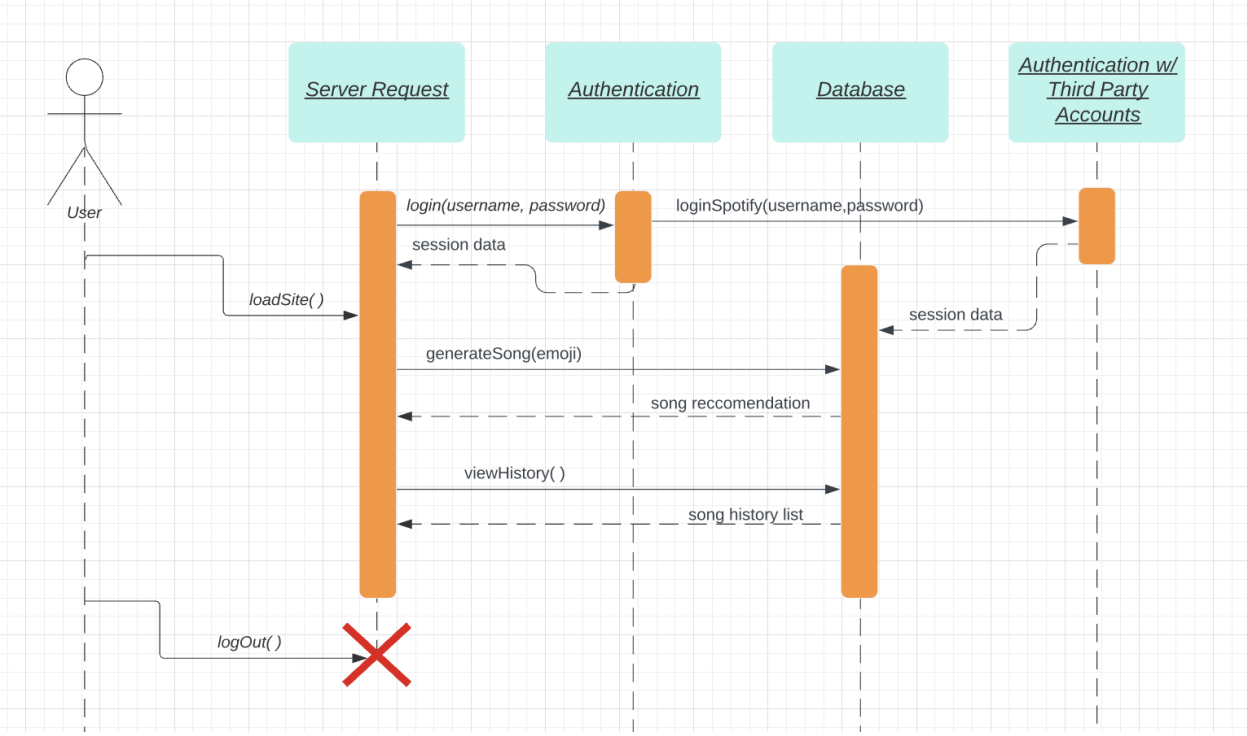
Why CSS: We will make use of CSS to style our website.

Why JavaScript: We will use JavaScript to make our website interactive.

Rationale behind framework: Our team has prior experience with working with these softwares, which is why we chose these applications.

2. Functional Design

2.1 Sequence Diagram



- When a user arrives at our site for the first time, they must create an account with MusicTherapy then continue to log in with their Spotify account.
- After logging in with their Spotify account, relevant information (saved songs, albums, artists, and playlists) is returned to the MusicTherapy database.
- When the user selects an emoji, the generateSong() class sends this information to the database which returns a song recommendation based on the user's current music library.
- When the user chooses to view their history, the viewHistory() class accesses and returns the list of songs they have generated in the past.

3. Structural Design