**Software Requirements and Design Document**

**For**

**Group <10>**

Version 1.0

**Authors**:

Philip Sablan

Aditi Anandjiwala

Jacob Holleman

Josiah Walters

Krenn Intharaksa

# 

# 

# **1.** **Overview (5 points)**

*Give a general overview of the system in 1-2 paragraphs (similar to the one in the project proposal).*

JukeBoxd is a website that allows users to create accounts and rate music albums, singles, or artists. Users will be able to rate these with a “star” system (such as 3 stars out of 5). Additionally, users will be able to add personal comments or reviews on music that they have rated. JukeBoxd will grab most of its music data, specifically what albums and artists are available to rate, from Spotify’s artist and music lists.

# **2.** **Functional Requirements (10 points)**

*List the* ***functional requirements*** *in sentences identified by numbers and for each requirement state if it is of high, medium, or low priority. Each functional requirement is something that the system shall do. Include all the details required such that there can be no misinterpretations of the requirements when read. Be very specific about what the system needs to do (not how, just what). You may provide a brief design rationale for any requirement which you feel requires explanation for how and/or why the requirement was derived.*

1.) Users shall be able to create a unique account with username and password (High)

2.) Users shall be able to log in to their already created accounts with correct username and password entry (High)

3.) Users shall be able to search for music to rate on the site (High)

4.) Users shall be able to rate and review music and have these ratings and reviews visible on their account pages (High)

5.) Users shall be able to view the ratings of other users on their account pages (Medium)

6.) Users shall be able to search for other users in order to view their ratings (Medium)

7.) Users shall be able to see average ratings and others’ reviews on music that they have searched for on the site (Medium)

8.) Users shall be able to provide an additional overall “like/dislike” denomination supplementing their rating and review (Medium)

9.) Users shall be able to make their account private so as to limit who can view their ratings and reviews (Low)

10.) Users shall be able to discuss music, artists, and other related topics on user forums(Low) (This is a good feature to add for flavor and usability but not 100% necessary for site function)

11.) Users shall be provided with a link to music they are reviewing/searching for that takes allows them to play the album/track on a streaming service such as spotify (Low)

Non-functional Requirements (10 points)

*List the* ***non-functional requirements*** *of the system (any requirement referring to a property of the system, such as security, safety, software quality, performance, reliability, etc.) You may provide a brief rationale for any requirement which you feel requires explanation as to how and/or why the requirement was derived.*

1.) Users’ account and password privacy will be safe from others

2.) User reviews may not include hyperlinks to domains that are not JukeBoxd or a music streaming service

3.) During times of normal site traffic, queries to site will be consistent and quick

4.) JukeBoxd should always be available to users

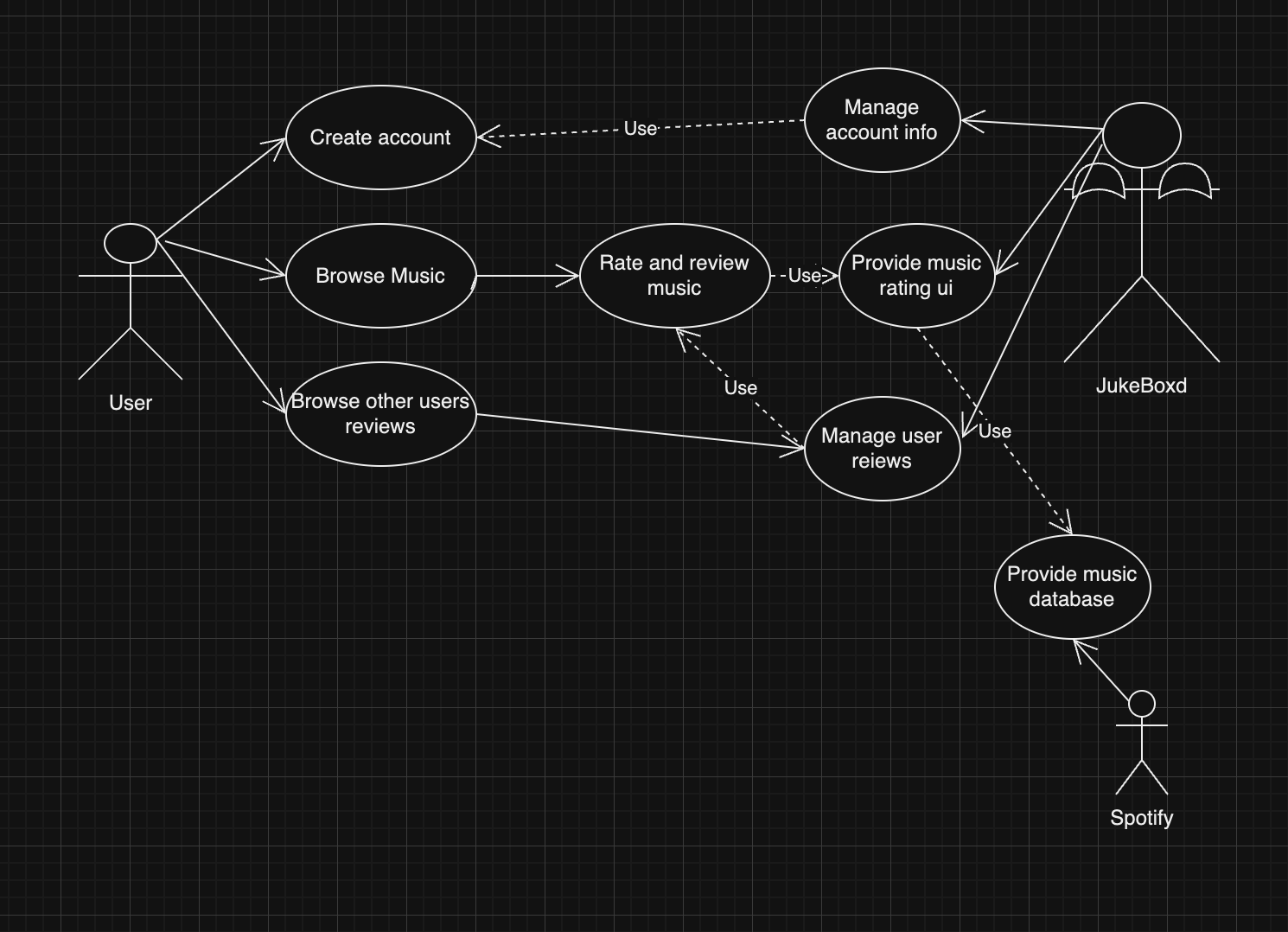
5.) JukeBoxd UI should be intuitive for new users to understand and use

6.) Account authentication must be in place such that users will only be able to create or modify reviews/ratings corresponding to their signed in account.

7.) Code and other relevant documentation shall be neat and appropriately sectioned, such that other group members may easily understand all parts of the project

Use Case Diagram (10 points)

*This section presents the* ***use case diagram*** *and the* ***textual descriptions*** *of the use cases for the system under development. The use case diagram should contain all the use cases and relationships between them needed to describe the functionality to be developed. If you discover new use cases between two increments, update the diagram for your future increments.*

***Textual descriptions of use cases****: For the first increment, the textual descriptions for the use cases are not required. However, the textual descriptions for all use cases discovered for your system are required for the second and third iterations.*

# **3.** **Class Diagram and/or Sequence Diagrams (15 points)**

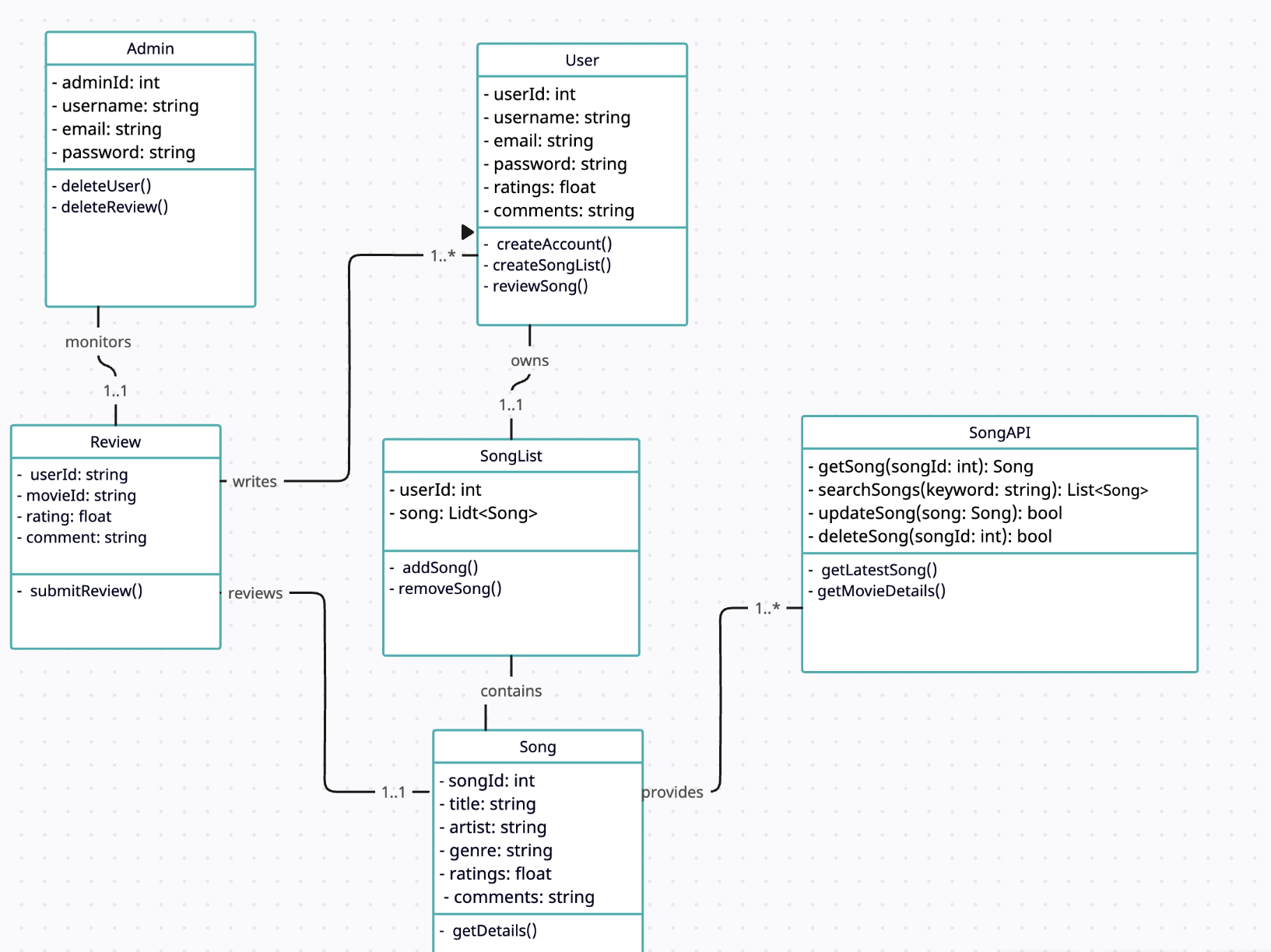
*This section presents a high-level overview of the anticipated system architecture using a* ***class******diagram*** *and/or* ***sequence diagrams****.*

*If the main* ***paradigm*** *used in your project is* ***Object Oriented*** *(i.e., you have classes or something that acts similar to classes in your system), then draw the* ***Class Diagram******of the entire system and Sequence Diagrams for the three (3) most important use cases in your system.***

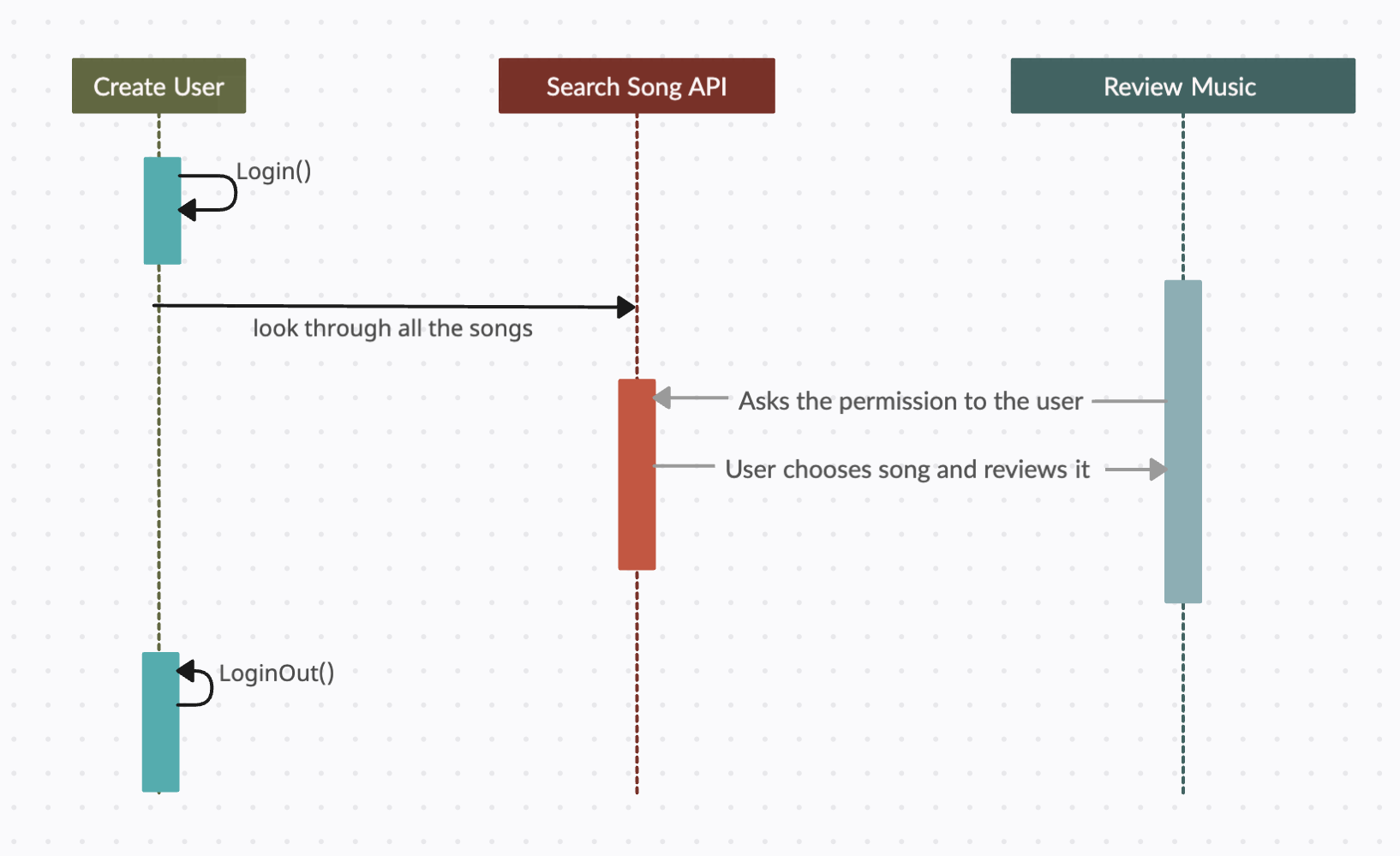
*If the main* ***paradigm*** *in your system is* ***not Object Oriented*** *(i.e., you* ***do not*** *have classes**or anything similar to classes in your system) then only draw* ***Sequence Diagrams****,* ***but for all the use cases of your system.*** *In this case, we will use a modified version of Sequence Diagrams, where instead of objects, the lifelines will represent the functions in the system involved in the action sequence.*

***Class Diagrams*** *show the* ***fundamental objects/classes*** *that must be modeled with the system to satisfy its requirements and* ***the relationships*** *between them. Each class rectangle on the diagram* ***must also include the attributes and the methods of the class*** *(they can be refined between increments). All the* ***relationships between classes and their multiplicity*** *must be shown on the class diagram.*

*A* ***Sequence Diagram*** *simply depicts* ***interaction******between objects*** *(or* ***functions -*** *in our case - for non-OOP systems) in a sequential order, i.e. the order in which these interactions take place. Sequence diagrams describe how and in what order the objects in a system function.*

Class Diagram- 

Sequence Diagrams-



# **4.** **Operating Environment (5 points)**

*Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.*

We are using React version 18.2 to run the website, Bootstrap version 5.3.2, and Adobe CSS tools 4.3.3. Website has been run on Macbook and Windows laptops. Currently using VisualCode in conjunction with GitHub to store and work on code.

# **5.** **Assumptions and Dependencies (5 points)**

*List any assumed factors (as opposed to known facts) that could affect the requirements stated in this document. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project.*

The project is reliant on a database of music, specifically music that is hosted on spotify. As is currently planned, the only music that will be available to rate will be what is posted on spotify. Thus, changes to this database will affect what music is available and access to it is important for the development of JukeBoxd. As of current JukeBoxd is being developed on localhost, and external server details are yet to be determined.