Keyboard Buddies

Earworm

Software Design Document

Name (s): Alvaro Lopez-Romero
Mitchell Mercer
Nathalie Claire Tivar
Rodolfo Rivera
Charles Quebral

Version: 1.0
Date: 20 November 2022

Supervisor: Professor Edmund Dantes
Computer Science Department
California State University, Northridge

Revision History

Version	By	Change Description	Date
1.0	All team members	Original Document	11/19/2022
1.1	Mitchell Mercer, Natalie Claire Tivar, Charles Quebral	Updated 4.2 Sequence Diagram, and 6.0 Requirements Matrix.	11/29/2022
1.2			
1.3			

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Purpose

The purpose of this document is to describe the architecture and system design for Earworm, a senior capstone that seeks to connect individuals and artists with like minded individuals based on criteria such as music taste, sex, age, and location. This document is for the Keyboard Buddies team, Professor Edmund Dantes and anyone who will be involved in this project.

1.2 Scope

EarWorm will be available to any individual, over the age of eighteen, to find and collaborate with individuals of similar music taste and garner semi-professional and personal relationships.

1.3 Overview

This document serves to illustrate the overall system design and architecture of the EarWorm service. In particular this document will lay out the specific implementation of the front and back end implementation.

1.4 Reference Materials

- SDD_Template.DOCX
- Web api. Spotify for Developers
- Spring Boot Reference Documentation

1.5 Definitions and Acronyms

- User - The person who utilizes our web application/services
- UML - Unified Modeling Language
- UI - User Interface
- ML - Machine Learning
- MySQL - Database management system
- JSON - Programming language
- OO - Object Oriented
- API - Application Programming Interface

2.0 SYSTEM OVERVIEW

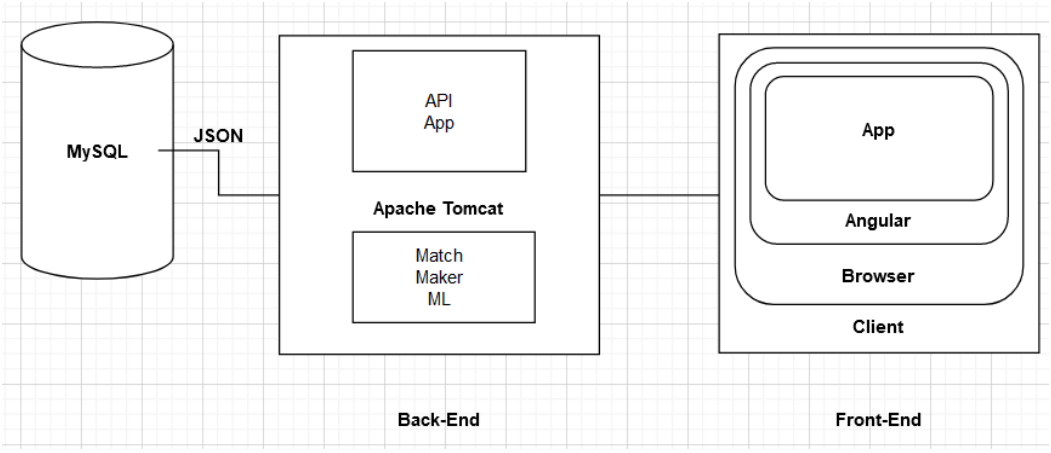

Figure 1

The above diagram illustrates the architecture for the development of the Earworm service. The backend of this service will be responsible for accessing, modifying, and utilizing a database full of individual users and formulating an assumption on what two users may be interested in meeting as well as forming an assumption of a user's music taste.

This service will utilize tomcat as it is a lightweight and well documented framework. Additionally, Apache is very user friendly and allows quick revision and modification.

Finally these design choices as illustrated above will lend itself to the potential development of a mobile application in the future.

3.0 SYSTEM ARCHITECTURE

3.1 Architectural Design

The architecture of EarWorm shall centralize around interactions which occur within the backend of EarWorm. In addition, the backend shall be split into subcategories based on subsystems. These systems shall include matchmaking and instant messaging. Both of these subsystems will use Apache Tomcat for network functionality to perform their appropriate operations. Furthermore to the user this process will be extremely simple. using a web browser they shall be presented with our website which will then offer them a simple and straightforward user experience. Our Backend API will keep a database up to date with a list of current users, and continually update these users with an assigned “Music Preference”. This music preference value will be used to connect users of similar tastes together using ranked lists and logic. This “Music Preference” will be assigned using a separate

machine learning module.

3.2 Interface Design

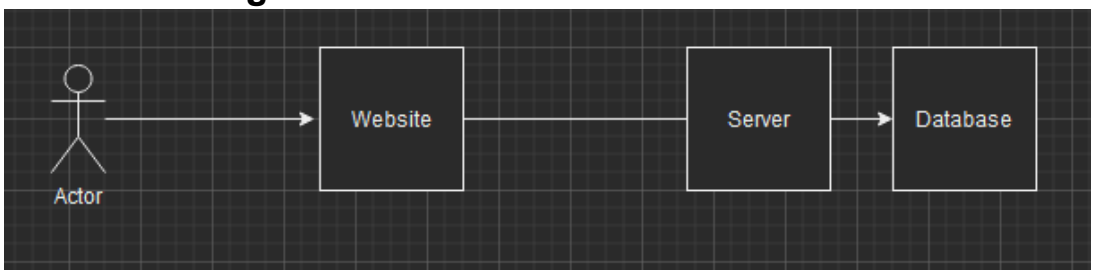

Figure 2

3.3 Decomposition Description

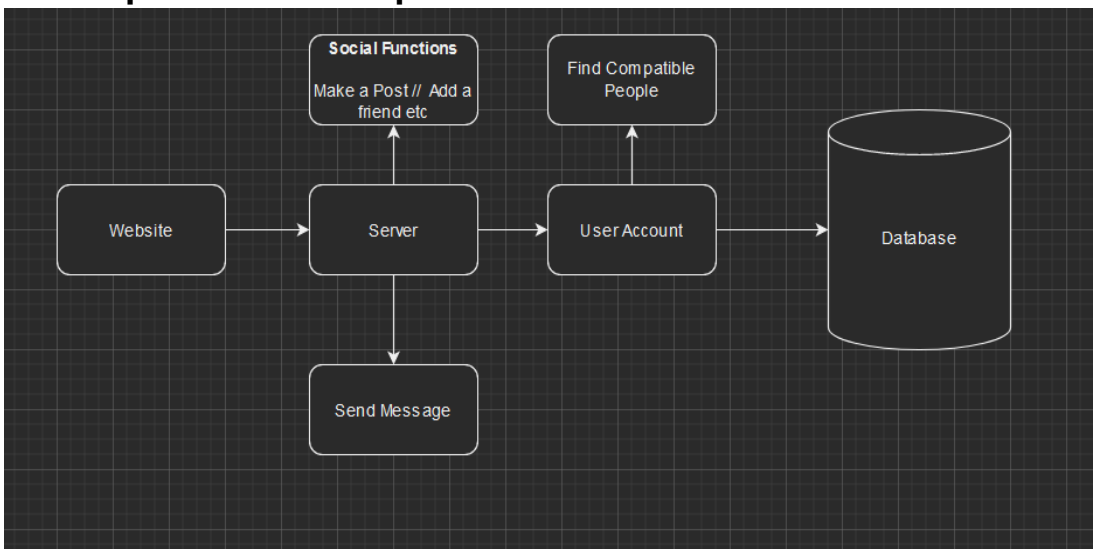

Figure 3

The Application shall use a web browser on the client machine to access the server via the internet. At this point this connection to our server will facilitate access to social media functions such as finding and making connections, adding friends, blocking users, making posts, and sending messages. Additionally this entry point is where user credentials are required to access the site or an account must be created.

4.0 COMPONENT DESIGN/DETAILED DESIGN

4.1 UML Diagram

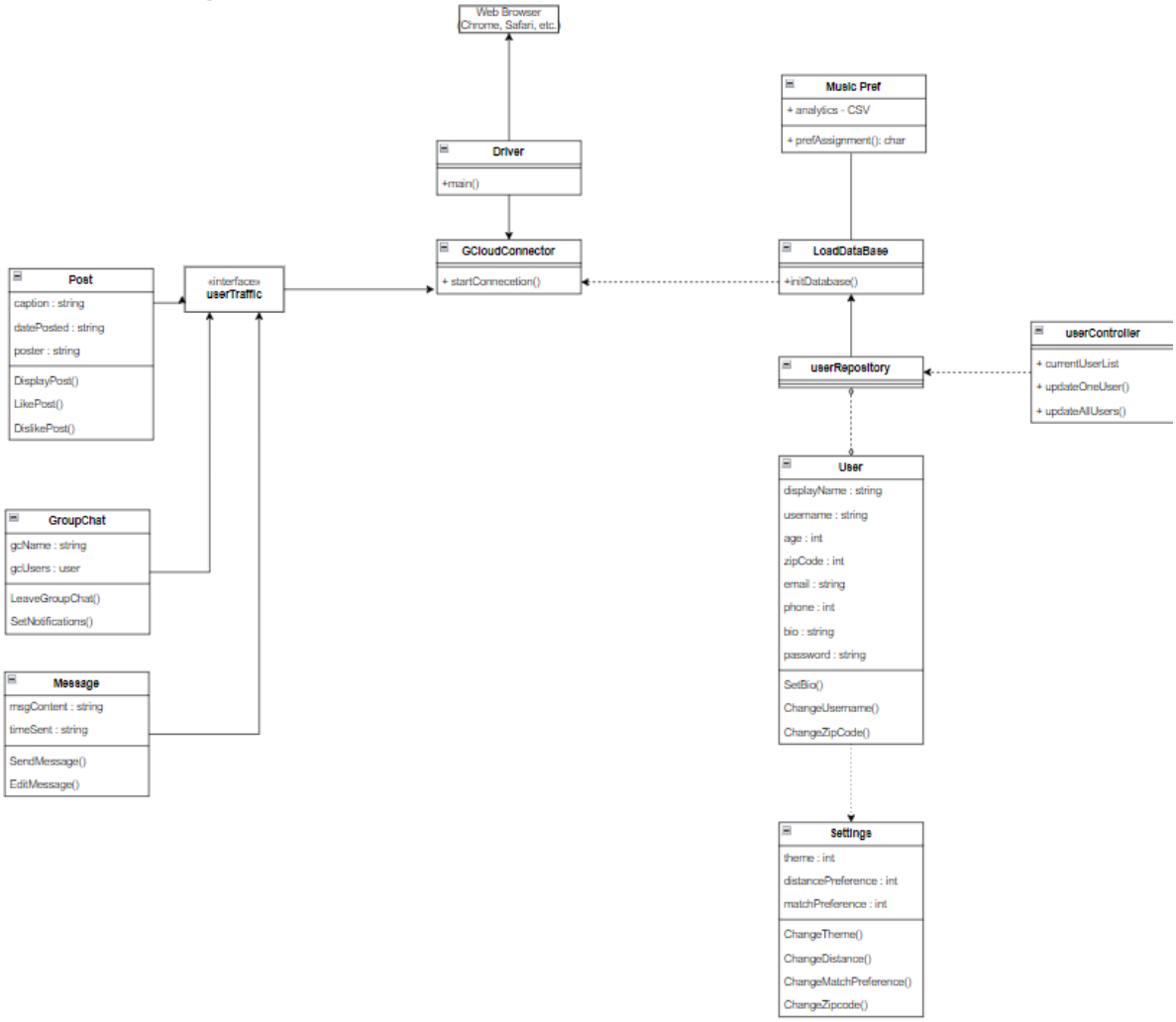

Figure 4

4.2 Sequence Diagram

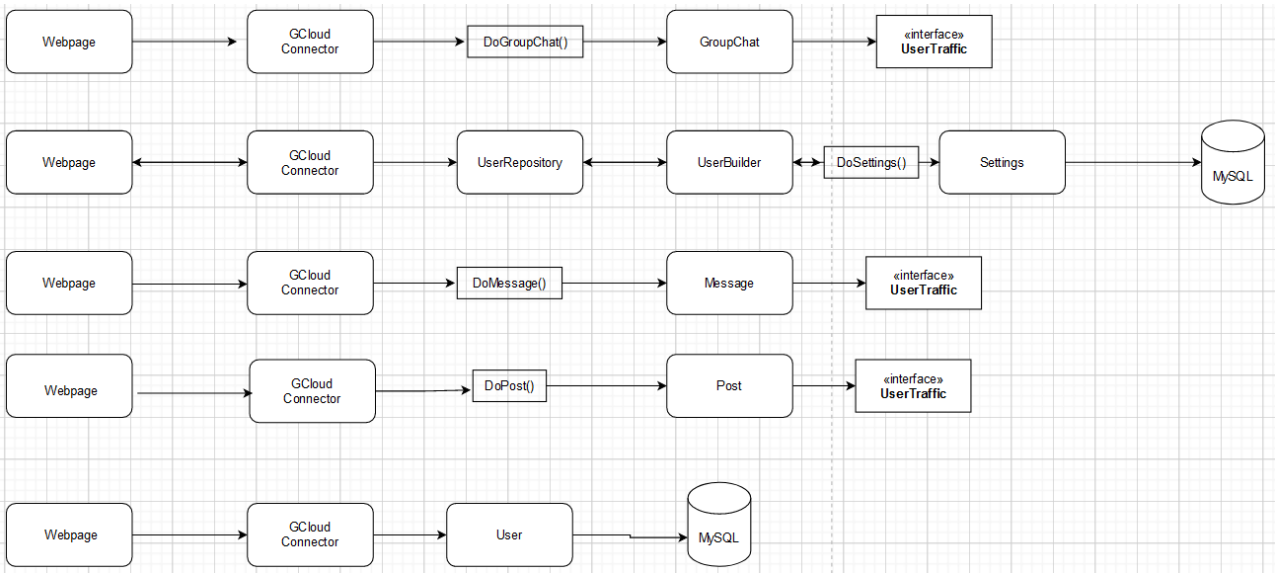

Figure 5

4.3 Database Schema

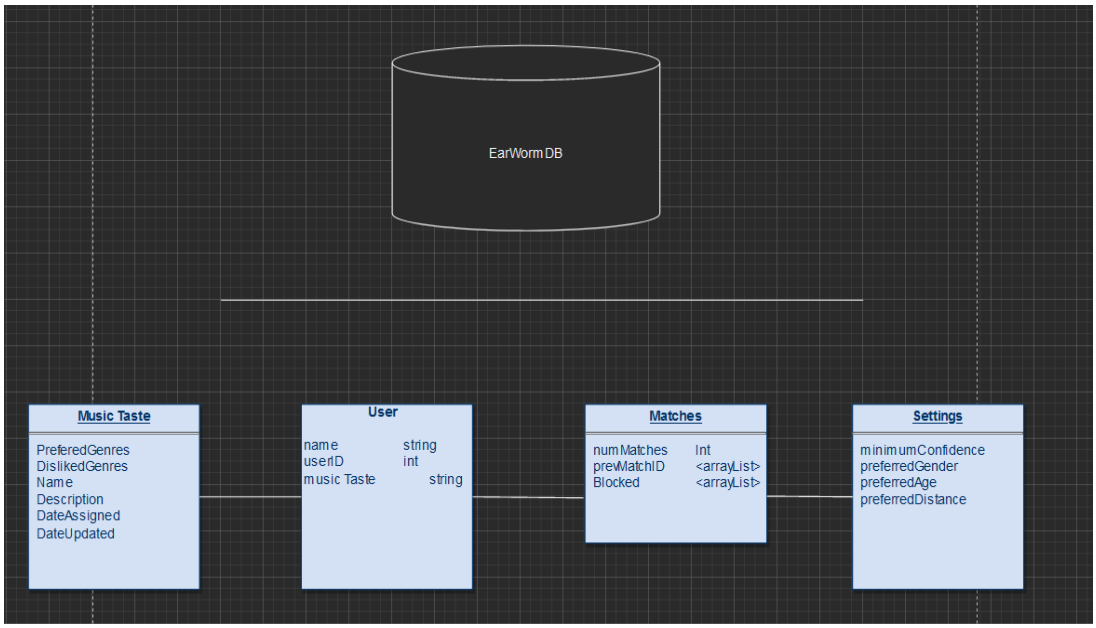

Figure 6

5.0 HUMAN INTERFACE DESIGN

5.1 Overview of User Interface

When visiting the website, users will be greeted by the intial landing page (figure 7). The website will be asking whether the user wants to sign up or to log in. New users will need to sign up; The Sign Up button will route the users to the sign up page where the website will ask for their information (name, location, age, music preferences) to get them started. Returning users can log back in and will be routed to the homepage. The footer includes necessary links and information about the website.

Figure 7- Landing Page (Wireframe)

Once the users are in the home page (Figure 2). The information they provided will be displayed on their profile (name, profile photo, location, music preferences, and social media links). A list of people with the same preferences will be presented to the user. The user can add them as a friend or message them. There will also be a home button, message button and a setting button. Home button will route the user back to the home page. Message button will show the user's inbox. The settings button will show the settings of the application as well as the log out button.

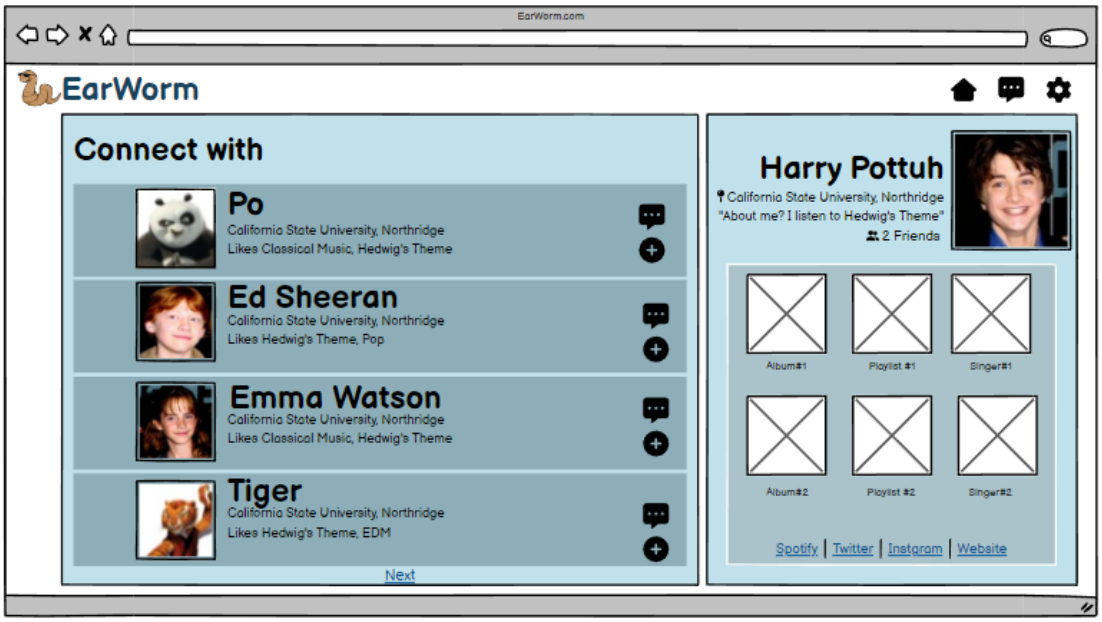

Figure 2 - Home Page (Wireframe)

6.0 REQUIREMENTS MATRIX

Provide a cross reference that traces components and data structures to the requirements in your software requirements specification (SRS) document.

Use a tabular format to show which system components satisfy each of the functional requirements from the SRS. Refer to the functional requirements by the numbers/codes that you gave them in the SRS. This paragraph contains the traceability from each software unit in this SDD to the SRS that was previously developed.

Sample Text:

FR 001	Machine Learning Module	3.2 SRS	Music Preferences
FR 002	Compatibility	3.2 SRS	User Preferences
FR 003	Performance	3.3 SRS	Spotify API
FR 004	Logical Database	3.4 SRS	Unique Users
SSA 001	Security	3.6.1 SRS	Credentials prerequisite
SSA 002	Reliability	3.6.4 SRS	App availability
SSA 003	Scalability	3.6.5 SRS	SQL Database
SSA 004	Usability	3.6.6 SRS	User

7.0 APPENDICES

Spring Boot Documentation - Phillip Webb, D. S. (n.d.). Spring Boot Reference Documentation.Retrieved from https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/

Spotify API - Web api. Spotify for Developers. (n.d.). Retrieved November 17, 2022, from https://developer.spotify.com/documentation/web-api/