Computer Engineering and Computer Science
491A Senior Project Product Proposal

Academic Term: Fall 2023

Team Name: The Decision Tree

Team Members: Aiden Hock, Jacob Phillips, Nathan Wolski, Noah Daniels, Kihambo Muhumuza, Diego Garcia

Team Leader: Aiden Hock

Date: 9/11/2023

Product Name: Kara-Goke

Table of Contents

Product Outline:

Product’s Value/Purpose:

The “Kara-Goke” Karaoke party planner aims to connect singing friend groups to each other anytime, anywhere. The goal of “Kara-Goke” is to bring singers together and revolutionize karaoke in a fun, viable, and painless process. With “Kara-Goke” you no longer must drive to a karaoke bar and hope to see your friends. Simply set up a “Kara-Goke” session in the app, invite your friends, customize your “Kara-Goke” game mode, and enjoy an exciting singing session with your friends anywhere you want! “Kara-Goke”’s value comes from its ubiquitous karaoke experience, highlighting our game-changing song voting system, real time pitch-analysis, tons of unique game modes, and real in-game highlights users can share amongst each other!

Vision:

Our vision for “Kara-Goke” is to create a meaningful user experience predicated around having fun, competing with your friends, and singing anywhere you want with anyone you want. Our app revolutionizes traditional karaoke conventions. We allow you to seamlessly integrate your own playlists, personalize your own unique singing challenges, vote on which song plays next, without ever having to step into a karaoke bar! Most of all, we want users to hold onto their unforgettable singing moments with their friends. That is why “Kara-Goke” would be incomplete without a highlight reel. This would allow users to keep their favorite memories from their funniest sessions as well as compile all the times they were “Pitch Perfect!” in their songs.

Features:

Kihambo

Medium to High Complexity Feature (Phase 1):

- **UI Host/Guest Dashboard:**
  - Develop a dashboard for hosts to manage the "Kara-Goke" session and for guests to experience and participate in as well.
  - This will be the home screen for the application, and to do this we'll need to design a user interface.
  - Will be user- friendly, so design will be simple enough to start a session
  - There will also be an option to add a profile picture and create a username that the user wants to go by

Low to Medium Complexity Feature (Phase 1):

- **Karaoke Profile Manager**:
  - Create a system for users to create profiles with customization features involving profile pics, usernames, their bio.
  - To do this, we'll need a system that allows the user to make an online profile using their email and/ or phone number.
  - Alternatively, we can give them the option to sign in as a guest, but if they choose this option, they won't be able to make upgrades and perks or collect points that they can use towards their account.
  - There will also be an option to let local users find you so that they can start a session in a LAN party

Aiden

Medium to High Complexity Feature (Phase 1):

- **Song catalog integration:**
  - This feature allows users to import their liked songs or playlists from Spotify, Apple Music, or Youtube to their Karagoke sessions or, by default, search any song on Youtube.
  - Implementation of playlist song selection will require us to use an API and develop a process that allows users to sync their music accounts with Karagoke.
  - Additionally, we will have to incorporate proper user authentication for users to securely sign into their music accounts and sync their music data to the KaraGoke app.

    - Finally, we will have to keep user’s song libraries stored on the KaraGoke app as well as develop a method to ensure these libraries are up to date with their external music profiles, allowing them the ability to pick and choose what songs they’d like to add to their KaraGoke music profile.
    - Implementation of the MVP will require us to implement Youtube’s robust song catalog as our default user song selection music streaming service first.

Low to Medium Complexity Feature (Phase 1):

- **Lyrics Display:**
    - Implement a HUD that is able to communicate to the user what lyrics are currently being played.
    - Integrate HUD into the UI so it can easily be minimized and expanded.
    - Depending on the selected song, this feature will analyze song meta-data to give us the lyrics, tempo, and points of singing emphasis.
    - Integrate lyric synchronization and highlight the current word being sung at the appropriate timings.
    - This will require us to create a dynamic sing-a-long lyric rendering tool that will identify and portray when lyrics are being sang and/or emphasized.
    - Test for lyric synchronization algorithm accuracy.
    - Implementation of the MVP will require us to develop a lyric synchronization algorithm and integrate a HUD into our UI.

Nathan

Medium to High Complexity Feature (Phase 1):

- **User pitch analysis scoring:**
    - Create a scoring system that grades the user’s performance.
    - Uses a pitch detection algorithm to compare to the song data.
    - Analyzes user’s pitch in combination with their timing to compute the score of their karaoke performance

Low to Medium Complexity Feature (Phase 1):

- **Feedback and Tips:**
    - Users would be able to see a breakdown of their score.
    - This would include the parts of the song that the user did not score well on
    - This will give the user the ability to improve by practicing where they messed up.

Noah

Medium to High Complexity Feature (Phase 1):

- **Inviting friends via share:**
  - This SOA feature lets users invite friends to their hosted karaoke party by sending invite links over the cloud.
  - Following the architecture we will be able to securely send back and forth packets of data as invite links and friends will be able to connect, chat, and play our game to their heart's content.

Low to Medium Complexity Feature (Phase 1):

- **Karaoke Party Reminders:**
  - This feature will be written in node.js
  - reminds users who have previously been notified or set up a karaoke session that it will be starting soon
  - This gives them enough prep time (sleep function) to warm up their voice and do whatever they need to do to hit those high notes.

Jacob

Medium to High Complexity Feature (Phase 1):

- **Real-time user song-queue voting system:**
  - Implement a song voting system that gives users a list of options of potential next songs and allows them to vote on which song is up next turn.
  - Potential next songs is determined using playlists from API Integration with Spotify/Apple Music
  - Users will be able to view their votes live in the voting phase with a dynamic chart including percentages and colors for each song.
  - Up until the end of the voting phase, each user will be able to change their answer.
  - When the vote finishes, the host will confirm the song and push it to the queue.

Low to Medium Complexity Feature (Phase 1):

- **Queue Manager:**
  - Use a queue data structure for a queue management system to allow the party host to sufficiently manage song rotations and allow guests to make song recommendations.

    - There will be a host-exclusive queue manager UI for user-friendly and intuitive queue management.
    - The queue will be updated after each vote ends if the vote creates a final unanimous decision.
    - If the vote ends in a tie, it is up to the host's discretion to add which song between the two to the queue.

Diego

Medium to High Complexity Feature (Phase 1):

- **Game modes:** Develop classic and creative karaoke game modes such as duets or thematic singing challenges for more fun.
    - Users will have the option to choose between different modes when creating a lobby or changing modes between sessions.
    - Modes include differing numbers of users (duet, quartet) or provide different themes or genres (such as pop, rock, or horror)

Low to Medium Complexity Feature (Phase 1):

- **Karaoke Party Chat:** party members will be able to communicate with each other through text chat during a live song session or through personal direct messages.
    - Users will be able to communicate with each other during games or in lobbies with a simple text chat.
    - Users will be able to "friend" each other and send messages while not in games.

User Types:

- **Root Admin (Super Admin):**
    - Responsibilities: Initialize general karaoke service, manage parties/party settings, manage user accounts, configure system settings, access karaoke session audio analytics.
- **Admin Delegates (Admin):**
    - Responsibilities: Controlling song selection queue, resetting passwords, general user management, assigning/removing permission sets, moderation of party chat, managing parties.
- **Normal Users:**
    - Responsibilities: Host karaoke sessions, invite friends to party, vote for songs, chat with others, have fun with karaoke, sing.

Target Audience:

- The target audience is anyone, and everyone who loves karaoke based games and a more immersive experience with friends and family. This is far more than just a karaoke application, it is for a community of people who want to expand on just plain old karaoke.

Scope:

- The scope of the web application involves creating a CDN that will be able to provide a variety of music gaming features. The CDN will have to be integrated so that users can connect with other users to improve the app experience.

Minimum Viable Product:

- User Registration
    - User authentication
    - text / email code for 2 factor authentication

- UI Host/Guest Dashboard
    - Simple and user- friendly so that setting up a session is easy
    - Profile picture option, otherwise the app will just be a solid color background
    -
- Karaoke Profile Manager
- Song catalog with API Integration with Spotify/Apple Music
    - User authentication
    - External library updater
- Karaoke Lyrics Display
    - Integrate HUD into UI
    - Identify song meta-data and integrate lyrics
    - Lyric synchronization rendering tool
- Party chat w/ hosting and inviting functions
- Voting system for determining songs/queues
- User pitch analysis scoring
    - Scoring system
    - Pitch detection algorithm
    - Combine pitch and timing to form basis of score
- Feedback and Tips
    - Breakdown of score
    - Include poor performance areas
- Inviting friends via share
- Karaoke Party Reminders
- Live song-queue voting system manager
- Game modes
- Karaoke Party Chat
    - Message storage/cataloging
    - Ensure adequate/dynamic number of simultaneous lobbies

Business terms:

- CDN: Content Delivery Network

Technical terms:

- LSTM NN: Long Term Short Term Neural Network (memory based algorithm)
- UI: User interface
- Queue Data Structure: A list in which all additions to the list are made at one end, and all deletions from the list are made at the other end.
- Meta-Data: data that describes other data
- HUD: Heads up display
- Pitch Detection Algorithm: An algorithm used to estimate the pitch of an audio clip
- SOA: Service Oriented Architecture

Resources:

- https://github.com/aidenhock/491A-Senior-Project-.git
- Music files in backend: https://github.com/vitejs/vite/issues/7778
- https://stackoverflow.com/questions/75739371/what-api-does-spotify-youtube-music-apple-music-use-to-integrate-with-google-h
- https://developer.spotify.com/documentation/web-api
- https://www.geeksforgeeks.org/cpp-program-to-create-an-interface/
- https://www.geeksforgeeks.org/queue-data-structure/