3324 – Object Oriented Programming

Music Livestream Bot

04/24/2023

Team M

Team Member Role – Ryan Grieb

Team Member Role – Anthony Tran

Team Member Role - Ryan Seidel

Presentation Outline

- Project Vision
- Project Planning
- Project Requirements
- Product Design
- Project Implementation
- Product Testing
- Lessons Learned
- Product Launch / Demonstration
- Questions

Project Vision

- Project Vision:
  - Team M decided to create a GUI application using Java Swing in order to create a simple interface for the user to be able to livestream music and change background pictures.
- Include your main stakeholders and user categories.
  - Users, Music Artist and labels, Business Partners, Developers, Project Managers, and Investors
- What are the strengths of your product?
  - User Friendly
- Why should we buy or use your product?
  - Easy to use and user friendly
  - Fully Automate a Livestream

Project Planning

- Provide relevant details of your project plan.
  - Create a user-friendly and engaging interface.
  - Provide high-quality audio and video streams for users.
- Provide your project communications and teamwork approach.
  - Clear Common Goal, Regularly Communication, Shared Responsibility
- What was the most important risk in the project? How did it turn out?
  - Technical Risk: Making sure the project meets ALL technical requirements
  - Looks amazing and works fantastically!

Project Requirements

- Main Use Cases:
  - Music Manager
  - Live stream
  - Song skip
  - Playlist Import

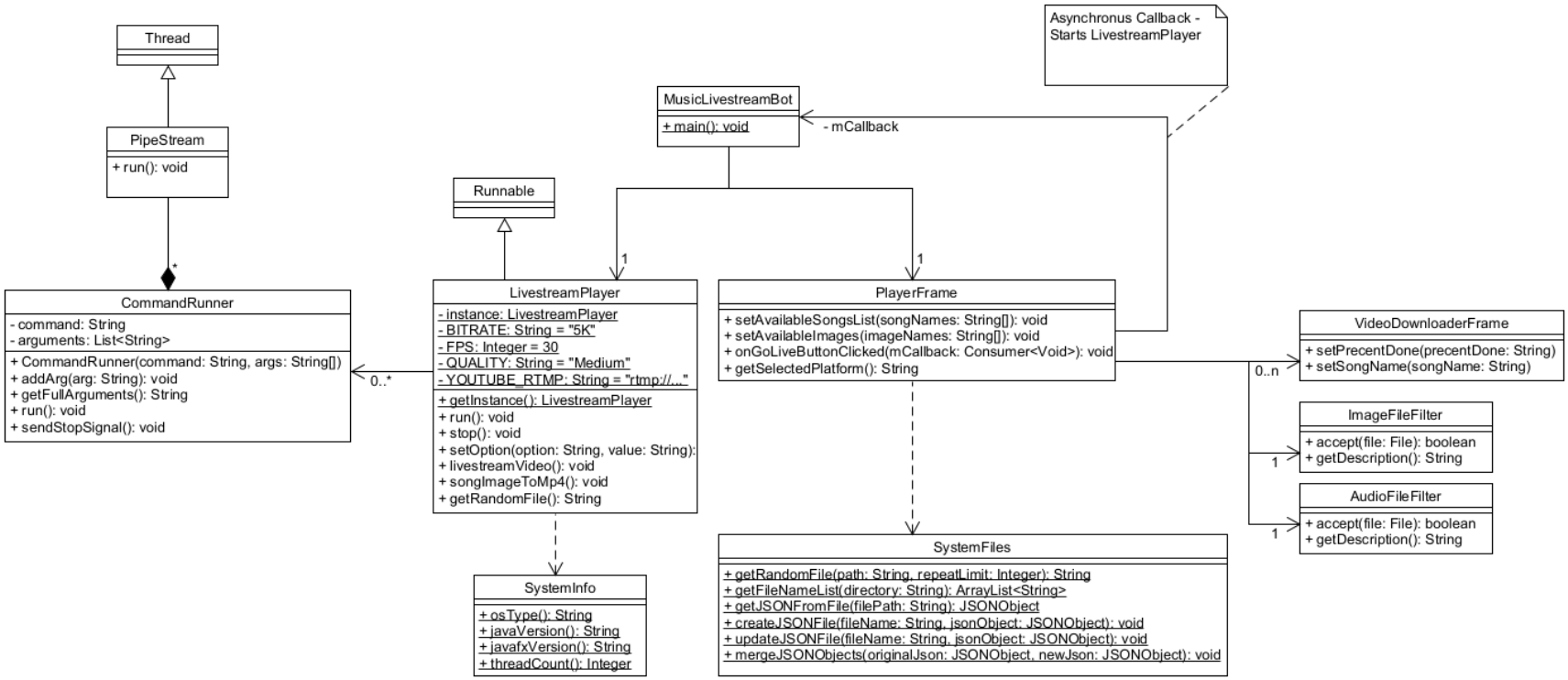
Project Requirements

- Main Requirements:
  - Software Requirement: The system shall play MP3 music streams in real-time, with minimal latency and buffering issues.
  - Software Requirement: The system shall allow users to control the livestream. It should also support advanced features such as start, stop, go to back to a song, go to the next song.
  - Hardware Requirement: The system shall have good connection.
  - Quality Requirement: The system shall have a simple user interface.
  - Quality Requirement: The system shall be tested to ensure efficient updates and bug fixes.

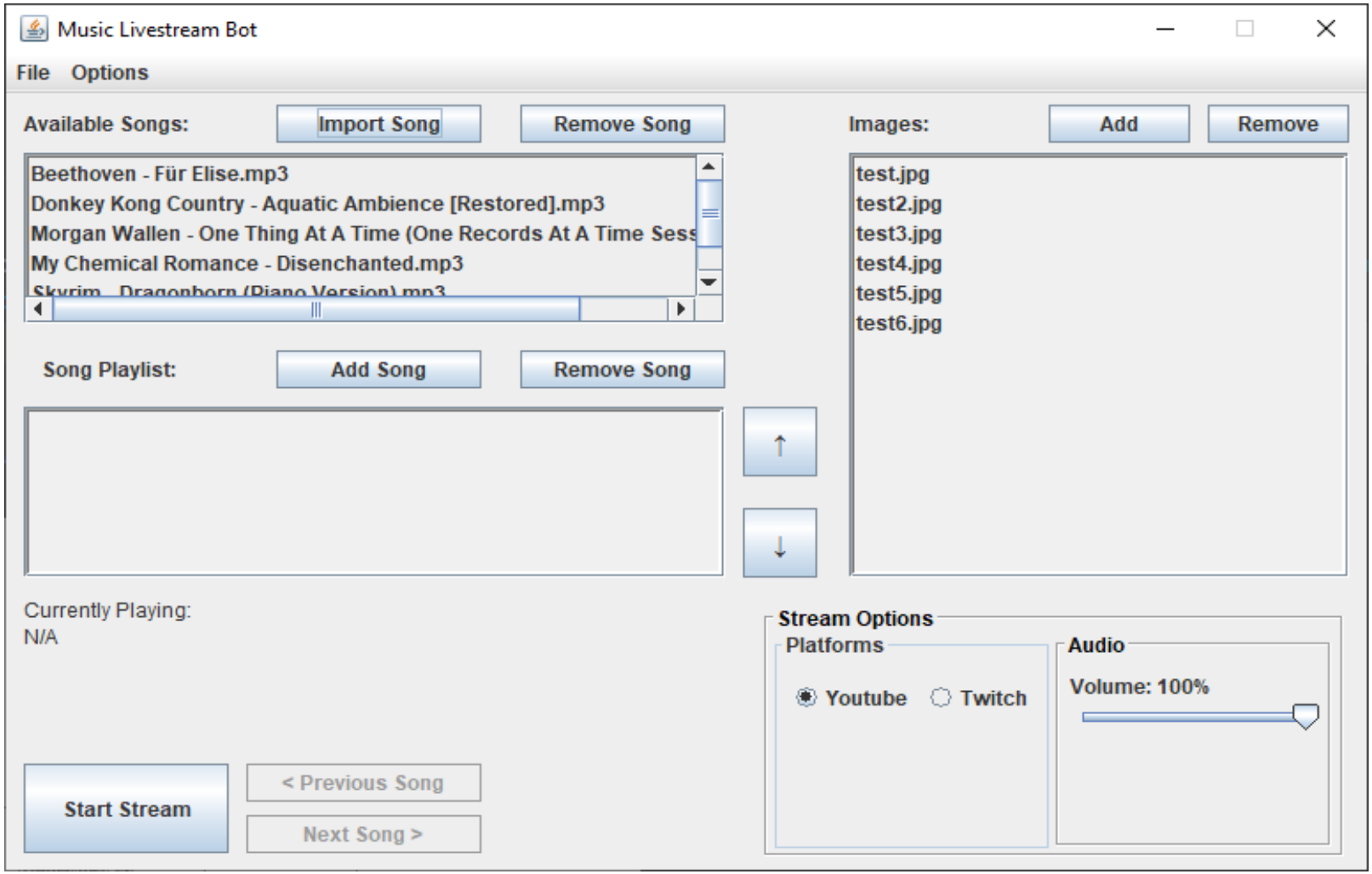
Product Design

- Main Design Decisions:
  - Live Streaming
  - User Interface Design
  - Architecture
- Main Constraints:
  - Compatibility
  - Monitoring

Product Design



Product Design



Project Implementation

- How did you achieve teamwork during product implementation
  - The group scheduled weekly meetings in the discord to discuss the state of the project.
- Team Roles:
  - Ryan Grieb
    - Software Developer / Requirements Engineer
  - Ryan Seidel
    - Software Tester /Quality Assurance Manager
  - Anthony Tran
    - Project Manager / Product Analyst

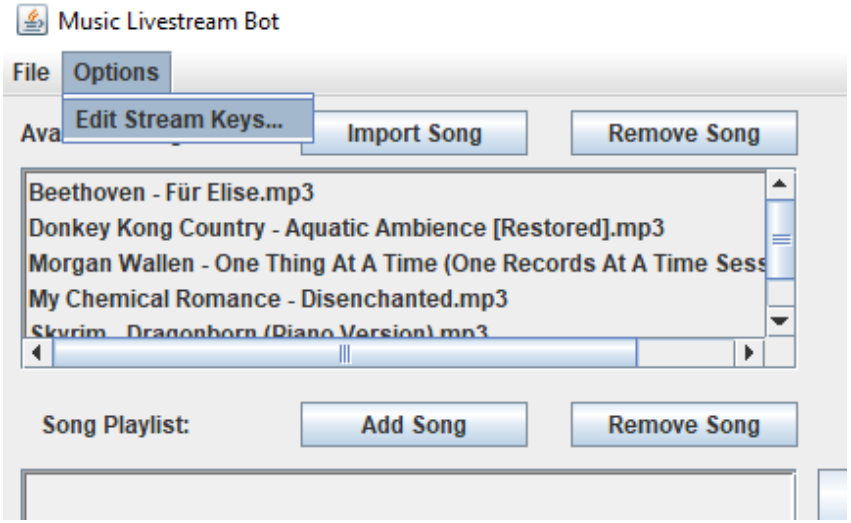
Product Testing

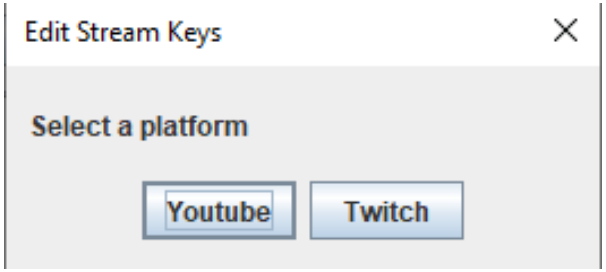
- Highlight your most important test cases.
  - Live Stream Option, Live Stream Audio, Song Skip
- How long do the tests take?
  - 4-8 hours
- What challenges did you face during testing?
  - Debugging Multithreading Code
  - Time Constraints
  - Finding Bugs
- How successful are your testing activities?
  - Most features worked properly and were successful.

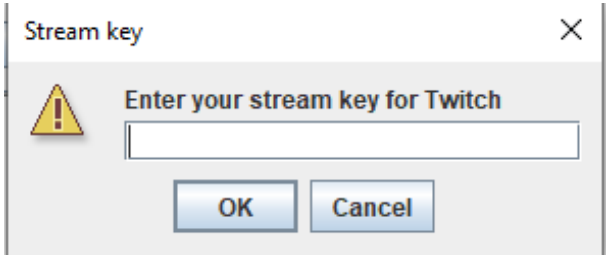
Lessons Learned

- What was the most challenging aspect of your team project?
  - Scheduling Conflicts: Just making sure everybody can contribute at the right time.
- What was motivating and demotivating?
  - Hearing it work for the first time with the live stream. (motivating)
  - Trying to find the documentations for the project dependencies (demotivating)
- What would you have done differently?
  - Set proper project milestones
- What are the things that you consider a success in this project?
  - The program fully automates a music livestream.
- What are the things that you should have improved and why?
  - Improving the look & feel of the UI. The default swing theme gives the project an armature feel.

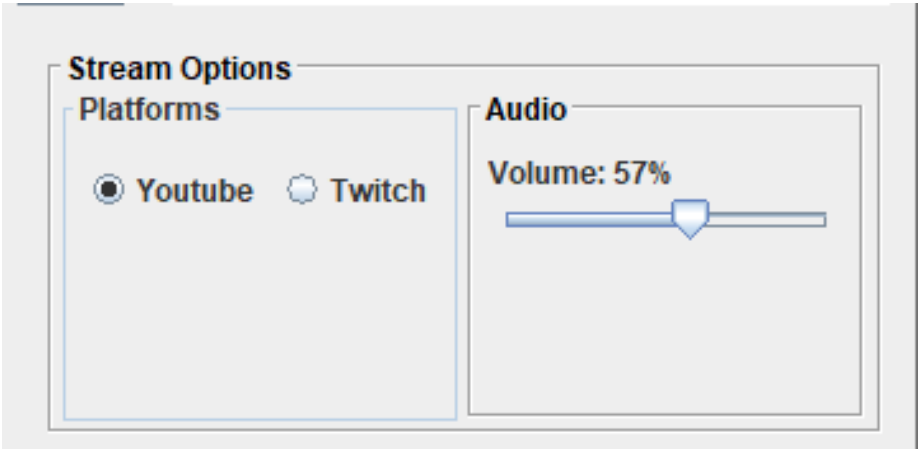
Product Features



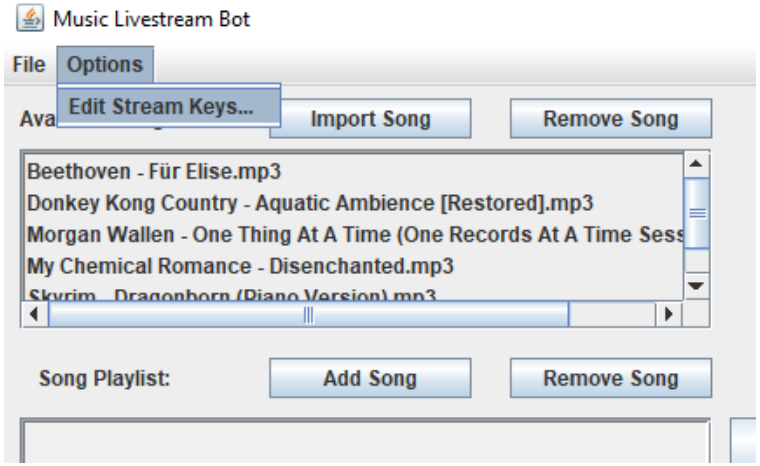


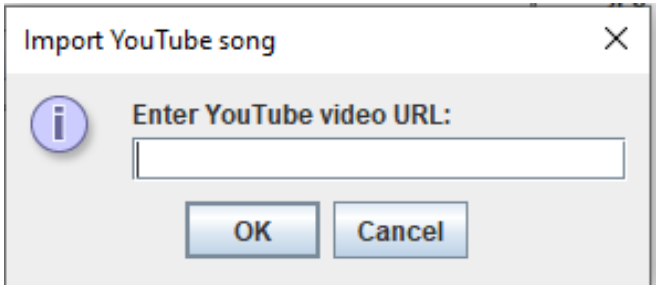


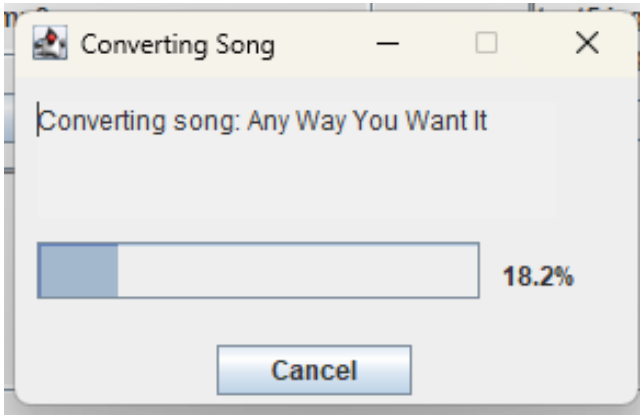
Product Features



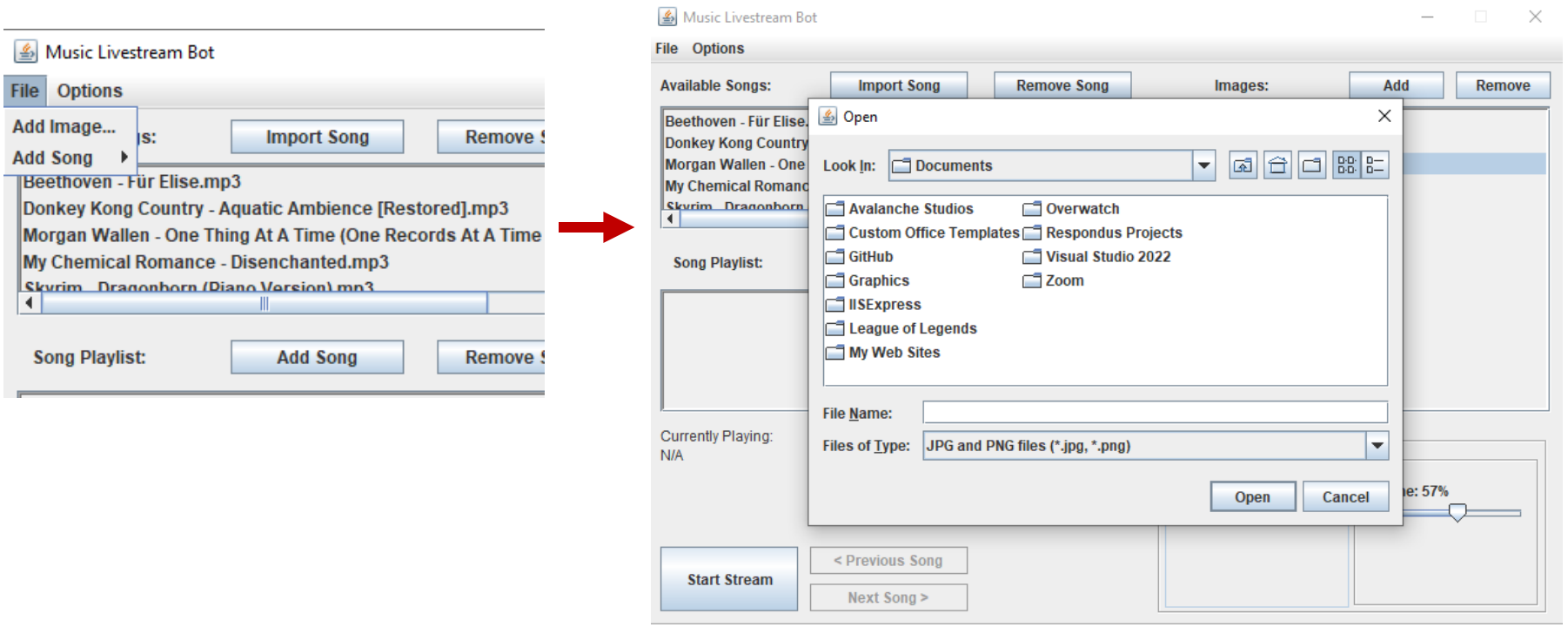
Product Features



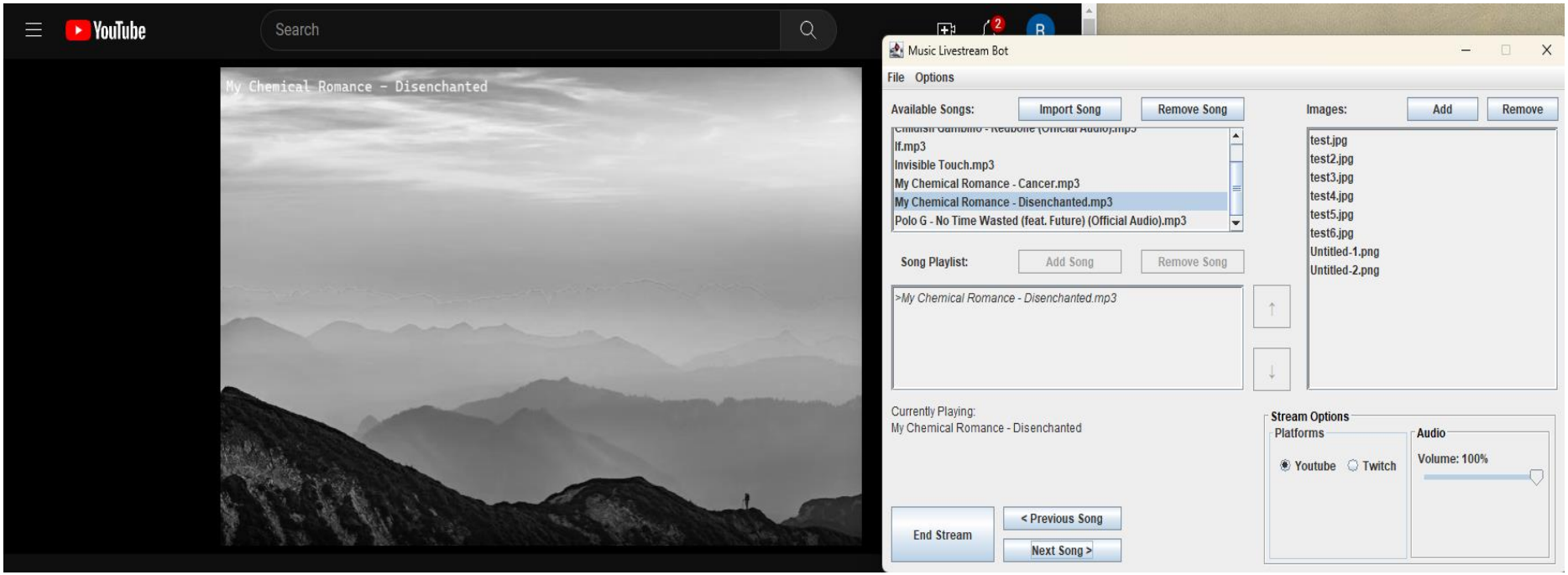




Product Features



Product Features



Product Launch

- Provide a demonstration of your product for 5-10 minutes.

Team M

Music Livestream Bot
Vision Document

Version 3.0

Document Preparation

Name	Role	Approval (Signature)	Approval Date
Ryan Grieb	Editor	Ryan M. Grieb	02/09/2023
Anthony Tran	Product Manager	Anthony Tran	02/10/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	02/10/2023
Ryan Seidel	Software Tester	Ryan Seidel	02/10/2023

Document Approvals

Name	Role	Approval (Signature)	Approval Date
Anthony Tran	Project Design Manager	Anthony Tran	02/10/2023
Wyatt Wilson	Project Configuration Manager	Wyatt Wilson	02/10/2023
Ryan Seidel	Project Test Manager	Ryan Seidel	02/10/2023
Ryan Seidel	Quality Assurance Manager	Ryan Seidel	02/10/2023
Ryan Grieb	Project Technical Manager	*Ryan M. Grieb*	02/09/2023
Anthony Tran	Project Manager	Anthony Tran	02/10/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	02/10/2023

Revision History

Date	Version	Description	Author
02/09/23	1.0	Initial document creation.	Ryan Grieb
02/10/23	1.0	Check for grammars	Ryan Seidel
03/24/2023	2.0	Updated the tables	Anthony Tran
03/24/2023	2.0	Updated the features	Ryan Seidel
04/25/2023	3.0	Finalizing the document	Ryan Seidel

Table of Contents

Project Vision Document

1. Introduction

There are many ways to stream music in the present day. Currently, there are many applications such as YouTube or Spotify which play music. On these platforms there are livestreams, which can play a wide variety of songs. Usually, these livestreams have a video or image accompanied by a song playing in the background. This is where our application comes into play. Currently, there is not a user-friendly way to set up these streams. Normally, you would need some technical background to broadcast music like the existing streams do already. Further, this document will detail the requirements and mission of this software project.

1.1 Purpose of the Document

This document collects the thoughts and ideas of Team M working on the Music Livestream Bot into a single document. All information about the program's direction is to be written here, and any issues that arise can be documented in one place. As time progresses, we can keep track of the features and requirements of this document and make changes accordingly.

1.2 Scope of the Document

This document contains information regarding the positioning of the project, stakeholder and user information, product overview and features. Code and documentation will not be presented in this document.

1.3 References

- Project Proposal
- Project Plan Version 2.0 April 25, 2023
- Project Software Requirements Specification Version 2.0 April 25, 2023
- Project Design Documents Version 2.0 April 25, 2023
- Project Test Documents Version 2.0 April 25, 2023

1.4 Definitions, Acronyms, and Abbreviations

Term	Abbreviation / Acronym	Definition
Graphical User Interface	GUI	A visual way of interacting with a computer using items such as buttons and textboxes.
Real-Time Messaging Protocol Secure	RTMPS	A communication protocol for streaming audio, video, and data over the Internet.
Fast Forward Moving Picture Experts Group	FFMPEG	A free and open-source software project that offers many tools for video and audio processing.
Application Programming Interface	API	A set of protocols, routines, and tools that allow different software applications to communicate and exchange information with each other.

2. Positioning

2.1 Business Opportunity

There is always a market for software that makes tasks easier for the end-user. In this case, our software helps non-tech savvy YouTubers/Twitch streamers who want to stream music and related images of their choice.

2.2 Problem Statement

The problem of	No obvious way to stream music and images in a streamlined manner with a GUI.
affects	YouTube/Twitch users and creators.
the impact of which is	A limited selection of high-quality music streams to choose from.
a successful solution would be	Lowering the difficulty of creating a music livestream through an intuitive GUI application.

2.3 Product Position Statement

For	YouTube/Twitch content creators.
Who	I want to livestream music.
The (product name)	Music Livestream Bot
That	Provides a streamlined way to set up a music livestream without any prior technical knowledge.
Unlike	PyLivestream, which has no GUI and needs external dependencies such as Python.
Our product	Will be a cross-platform program, which has an easy-to-use GUI to guide the user through the process. All required dependencies like FFMPEG can be packaged with the program on platforms like Windows.

3. Stakeholder and User Descriptions

3.1 Stakeholder Summary

Name	Description	Responsibilities
YouTube Company	The child-company of Google which has a direct role of managing the YouTube platform.	Ensures the YouTube platform is maintained and runs properly. Specifically, the RTMPS links to stream video.
End-users:	These are the individuals who will be using the software on a regular basis to perform specific tasks.	Provide feedback to developers that may be used for product enhancement.

3.2 User Summary

Name	Description	Responsibilities	Stakeholder
Content Consumer	Would watch the end-result of our product.	Consume content.	
Content Creator	The users of our program.	Produce/secure songs and images to display for the livestream. Maintain and moderate livestream.	

3.3 User Environment

The main user environment for the program is the desktop computer. This is due to the requirement of local storage for mp3 files, and image aggregation. The software is cross-platform on the desktop, so the user can use our program on Linux, Windows, or MACOS. Our software heavily utilizes FFMPEG, which is a command line program that can process video for the music livestream. We integrate by packaging the FFMPEG software as an external dependency on Windows, which is downloaded along with our software. On other platforms such as Linux and MACOS the user needs to install FFMPEG through their respective package manager. Though this is subject to change in the future. In the future this can become a web application, where all files are managed through a centralized server. Overall, the application is intended for a single user, who should use the program less than 20 minutes to set up a successful live stream.

3.4 Summary of Key Stakeholder or User Needs

- Need 1: Integration with Music Streaming Services
    - Reason: To play music, the music bot needs to be integrated with music streaming services like Spotify or SoundCloud.
    - Current solution: Not specified in the proposal.
    - Desired solution: The stakeholder or user may want the music bot to be able to access and play music from one or more popular music streaming services.
- Need 2: Finding Random Images Based on Genre
    - Reason: The music bot needs to display a random image based on the genre of the song being played.
    - Current solution: Not specified in the proposal.
    - Desired solution: The stakeholder or user may want the music bot to be able to find and display a random image based on the genre of the song being played.
- Need 3: User Song Requests
    - Reason: Users may want to request specific songs to be played during the livestream.
    - Current solution: Not specified in the proposal.
    - Desired solution: The stakeholder or user may want the music bot to include a feature where users can make song requests via chat during the livestream.

3.5 Alternatives and Competition

4. Product Overview

This section provides a high-level view of the product capabilities, interfaces to other applications, and system configurations. This section usually consists of two subsections, as follows:

- Product perspective
- Assumptions and dependencies

4.1 Product Perspective

The product perspective for the music bot focuses on its capabilities and functionalities. The music bot aims to automate livestreams on YouTube by choosing random songs based on a specified genre and displaying a random image based on the genre of the song or a specified input. The music bot is expected to play music from popular music streaming services, allow users to make song requests via chat, and include chat moderation features.

4.2 Assumptions and Dependencies

Assumption 1: The music bot has access to music streaming services like Spotify or SoundCloud.

Assumption 2: The music bot has the necessary API (Application Programming Interface) access to generate random images based on the genre of the song.

Dependency 1: The music bot has been integrated with YouTube or a tool like OBS to livestream itself.

Dependency 2: The music bot has a reliable and fast internet connection to access music and display images.

4.3 Cost and Pricing

The Music Livestream Bot is intended to be freeware. Certain features of the program can include a separate paid version of the program. Pricing of this premium version should range from $5-$20, respectively.

4.4 Licensing and Installation

Licensing:

The music bot will require proper licensing to access music streaming services and use APIs (Application Programming Interface) to generate images. The licensing agreements with the music streaming services and image generation API provider will need to be reviewed to ensure compliance with their terms and conditions.

Installation:

1. Choose your operating system: Before starting the installation process, ensure that you have the appropriate version of the music bot software for your operating system (Windows, Mac, or Linux).
2. Download the installation package: Visit the music bot's official website or an authorized download site to download the installation package for your chosen operating system.
3. Run the installer: Locate the downloaded installation package and double-click on it to run the installer. If prompted, you may need to allow the installer to make changes to your system.
4. Follow the installation wizard: The installation wizard will guide you through the installation process. Make sure to read each step carefully and provide the requested information. This may include agreeing to terms and conditions, choosing an installation directory, and selecting the desired components to be installed.
5. Configure network licensing (optional): If you plan to use the music bot with multiple users or to support multiple streams simultaneously, you may need to configure network licensing during the installation process. Follow the prompts provided by the installation wizard to do so.
6. Set up security measures (optional): To protect your music bot from unauthorized access, you may need to set up password protection and serialization during the installation process. Follow the prompts provided by the installation wizard to configure these security measures.
7. Complete the installation: Once you have followed all the steps in the installation wizard, click on the "Finish" or "Install" button to complete the installation process. The installer may require you to restart your device to finalize the installation.
8. Launch the music bot: After the installation is complete and your device has restarted (if required), locate the newly installed music bot application on your device and double-click on it to launch the software. You can now start using the music bot to play and manage your music collection.

5. Product Features

5.1 Feature 1 – Music Manager

Ability to choose a random song in our playlist. Should have the ability to prevent the chance of choosing the same song randomly, up to a certain number of songs.

5.2 Feature 2 – Image Hub

Like the Music Manager, just with images. This would be implemented before the Image Generator feature.

5.3 Feature 3 – Image Manager

Leverage emerging AI (Artificial Intelligence) technologies such as DALLE-2 and generate unique images to livestream. We would need to integrate their APIs into our application. Allow the end-user to prompt which images they would like to see.

5.4 Feature 4 – Webcaster

Ability to stream the video to a RTMPS link, such as the ones provided on YouTube.

5.5 Feature 5 – Text Overlay

Display informational text over the provided image on the livestream. This text can contain the song name, and author.

5.6 Feature 6 – Song Skip

Can be implemented through the program GUI, in which we provide the option to skip to the next song in the playlist.

5.7 Feature 7 – Playlist Creator

Graphical interface for user-made playlists. User can add and remove mp3 files for their playlists

5.8 Feature 8 – Playlist Import

Import playlists from YouTube utilizing external programs such as YouTube-dl.

5.9 Feature 9 – Audio Controls

Control audio on the song played, such as volume, pause, play. This can include a scrubber to control where in the song we play.

5.10 Feature 10 – Chat Control

Allow users who watch the stream have options to control the livestream through the chat. This can include song requests, or skips.

6. Constraints

Copyright issues of playing certain songs will arise. YouTube scans all videos that violate copyright and includes livestreams. Our users might be at-risk of playing copyrighted music when they are selecting songs to play. We might be required to provide a disclaimer, stating that we are not responsible for any legal issues regarding copyright claims.

7. Quality Ranges

Performance:

The music bot should have a quick response time, allowing users to select songs and display images in a timely manner.

The music bot should have minimal latency when playing music and live streaming to YouTube or OBS.

The music bot should be able to play music and livestream simultaneously without sacrificing performance.

Robustness:

The music bot should be able to handle sudden spikes in traffic or changes in network conditions without crashing or producing errors.

The music bot should be able to recover gracefully from unexpected failures or system crashes.

Fault Tolerance:

The music bot should be designed to handle unexpected failures, such as network outages, without affecting its operation.

The music bot should have robust error handling and reporting capabilities to help diagnose and resolve issues.

Usability:

The music bot should have a user-friendly interface that allows users to easily select songs and images, make song requests, and moderate chat.

The music bot should be intuitive and easy to use, with minimal learning time required for users to get started.

Similar Characteristics:

The music bot should be secure, with robust security measures in place to prevent unauthorized access and data breaches.

The music bot should be scalable, allowing it to support an increasing number of users and streams as needed.

The music bot should be maintainable, with clear documentation and an architecture that allows developers to add new features or fix bugs easily.

8. Precedence and Priority

Choose random songs based on genre: This feature is the core of the music bot and is critical for the application to function as intended. This feature should be given a high priority.

Display random image based on genre: This feature adds visual appeal to the music bot and is essential for the user experience. This feature should be given a high priority.

Livestream to YouTube: This feature is essential for the music bot to reach its target audience and is a key differentiator from existing solutions. This feature should be given a high priority.

User song requests in chat: This feature adds interactivity to the music bot and can help to increase engagement and user satisfaction. This feature should be given a medium priority.

Chat moderation: This feature is important for ensuring the chat remains appropriate and respectful and can help to maintain the quality of the user experience. This feature should be given low priority, as it can be added later if necessary.

Team M

Music Livestream Bot
Project Plan Document

Version 3.0

Document Preparation

Name	Role	Approval (Signature)	Approval Date
Ryan Grieb	Editor	Ryan M. Grieb	02/17/2023
Anthony Tran	Product Manager	Anthony Tran	02/17/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	02/17/2023
Ryan Seidel	Software Tester	Ryan Seidel	02/17/2023

Document Approvals

Name	Role	Approval (Signature)	Approval Date
Anthony Tran	Project Design Manager	Anthony Tran	02/17/2023
Wyatt Wilson	Project Configuration Manager	Wyatt Wilson	02/17/2023
Ryan Seidel	Project Test Manager	Ryan Seidel	02/17/2023
Ryan Seidel	Quality Assurance Manager	Ryan Seidel	02/17/2023
Ryan Grieb	Project Technical Manager	Ryan M. Grieb	02/17/2023
Anthony Tran	Project Manager	Anthony Tran	02/17/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	02/17/2023

Revision History

Date	Version	Description	Author
02/17/2023	1.1	Revising	Anthony Tran
03/31/2023	2.0	Revision for iteration 2	Ryan Seidel
04/25/2023	3.0	Finalizing the document	Ryan Seidel

Table of Contents

Project Plan

1. Introduction

1.1 Purpose of the Document

The purpose of this document is to coordinate the activities of Team M. It helps focus on the milestones that need to be made for the project. This document includes any Risks that may involve Team M team to help strengthen as a team.

1.2 Scope of the Document

The scope of this project plan document is to provide guidance and coordination for Team M in achieving their project goals and deliverables. The document outlines the business, product, and quality goals, the key stakeholders, and their needs. The project milestones and schedule are also included to help track the project's progress. This document contains a work breakdown structure (WBS) to outline the different work packages and responsibilities of each team member. Additionally, this document highlights potential risks that may impact the project's success and provides approaches to mitigate them. The scope of this document includes all project deliverables, such as the music livestream bot, user documentation, marketing materials, technical documentation, training materials, and project report. The communication channels and project members' roles and responsibilities are also outlined in this document. Finally, the appendix contains definitions, acronyms, and abbreviations used throughout the document.

1.3 References

- Project Proposal
- 'Team M Project Vision Document Version 2.0 April 25, 2023
  - Project Software Requirements Specification Version 2.0 April 25, 2023
  - Project Design Documents Version 2.0 April 25, 2023
  - Project Test Documents Version 2.0 April 25, 2023
-

1.4 Definitions, Acronyms, and Abbreviations

See Appendix A.

2. Project Goals

2.1 Business Goals

1. Increase revenue by 25% in the first year of operation.
2. Build a loyal fanbase of at least 10,000 active users within the first six months.
3. Establish partnerships with at least three music-related brands within the first year.
4. Expand the music livestream bot to other platforms within the first year.

2.2 Product Goals

1. Create a user-friendly and engaging interface for the livestream music bot.
2. Provide high-quality audio and video streams for users.
3. Implement features such as customized chat commands and alerts to enhance user experience.
4. Implement features such as customized chat commands and alerts to enhance user experience.
5. Offer multiple payment options for users to support the music livestream bot. to support the music livestream bot.

2.3 Quality Goals

1. Ensure the livestream music bot is accessible and functional on a variety of devices and internet speeds.
2. Maintain uptime of the music livestream bot to a minimum of 99%.
3. Provide prompt and efficient customer support to users.
4. Continuously gather user feedback and make improvements to the music, livestream bot based on user input.

3. Project Stakeholders and Stakes

3.1 Summary of Key Stakeholder or User Needs

1. **Users:** The users of the music livestream bot are the primary stakeholders. Their stakes include having access to high-quality audio and video streams, engaging features, and the ability to support the bot in numerous ways.

2. **Music artists and labels:** Music artists and labels are also stakeholders in the project. Their stakes include having a platform to reach a wider audience and the ability to monetize their music through the music livestream bot.

3. **Business partners:** Business partners such as sponsors and advertisers have a stake in the project as well. Their stakes include having exposure to the music livestream bot's audience and the potential for increased sales and revenue.

4. **Developers and designers:** The developers and designers of the music livestream bot have a stake in the project's success as well. Their stakes include the satisfaction of creating a successful product and potentially receiving recognition or compensation for their work.

4. **Project managers and investors:** Project managers and investors also have a stake in the project's success. Their stakes include the potential for increased profits and a successful project outcome.\

5. Project Budget

The Team M project will require all 4 team members to be active for the project and contribute several hours (3-4 hrs.) a week for coding, planning, and team meetings.

6. Project Milestones and Schedule

Project Start Date: 2/3/2023

Project End Date: 4/26/2023

Project Milestone	Date	Deliverable
Problem Analysis, Vision Creation, Project Planning	2/10/2023 2/17/2023	Product Vision Document Project Plan
Requirements Analysis	3/31/2023	Software Requirements Specification (SRS)
Product Design	4/7/2023	Software Design Document (SDD)
Product Implementation	4/14/2023	Software Code
Product Test	4/12/2023	Software Test Document (STD)
Final Product Deliverable	4/22/2023	All Documentation and Code
Product Launch	4/26/2023	Project Presentation

7. Project Communications

Effective communication is critical for the success of any project. One of the primary communication channels will be through direct messaging in the group chat on Message and Discord. The group chat will also allow the team to discuss any issues or roadblock they may be facing and find solutions together. The format of these meetings may vary depending on the team’s location and availability and can include virtual or in-person options.

8. Project Members and Roles

Project Role	Name	Responsibilities
Project Manager	Anthony Tran	In charge of leading the development and management of the product
Project Technical Manager	Ryan Grieb	Overseeing the technical aspects of the project, including ensuring that the software meets technical specifications and requirements, and working closely with the team to ensure that the project is delivered on time and within budget.
Project Design Manager	Ryan Seidel	In charge of making sure that the code is best code and meets all the requirements for customer's satisfaction
Project Test Manager	Ryan Seidel	Ensuring that the project meets all quality assurance standards, creating and executing test plans, documenting results, and reporting any issues or defects to the team.
Project Engineer	Ryan Grieb	Overseeing the technical aspects of the project, including ensuring that the software meets technical specifications and requirements, and working closely with the team to ensure that the project is delivered on time and within budget.

9. Project Work Breakdown Structure (WBS)

Work Package Number	Work Package Name	Responsible Team Member	Definition
WP-1	Bot Architecture	Project Engineer	The Project Engineer is responsible for designing the architecture of the live music bot, including selecting the programming language, framework, and other technologies that will be used to develop the bot.
WP-2	Feature Design	Project Design Manager	The Project Design Manager is responsible for designing the features of the live music bot, such as song selection, playback controls, and playlist management. This work package involves creating user stories, wireframes, and other design artifacts that will guide the development of the bot.

WP-3	Development	Project Technical Manager	The Project Technical Manager is responsible for overseeing the development of the live music bot, including coding, testing, and debugging. This work package involves breaking down the bot's features into smaller, manageable tasks and assigning them to team members.
WP-4	Quality Assurance	Project Test Manager	The Project Test Manager is responsible for ensuring that the live music bot meets all quality assurance standards, including testing all features and functionality, reporting any defects or bugs, and verifying that all requirements have been met.
WP-5	Deployment	Project Manager	The Project Manager is responsible for deploying the live music bot to its target environment, whether that be a public-facing server or a private network. This work package involves configuring the necessary infrastructure, setting up authentication and access controls, and verifying that the bot is running as expected.

10. Project Deliverables

Deliverable Number	Deliverable Name
D-1	Music Livestream Bot
D-2	User Documentation
D-3	Marketing Materials
D-4	Technical Documentation
D-5	Training Materials
D-6	Project Report

11. Project Risks

Risk Number	Risk	Risk Reduction Approach
R-1	Time Management	Keeping up with tasks and making sure that all work is turned in correctly.
R-2	Implementation: There will be hurdles integrating our product with programs like discord and OBS.	OBS and discord provides documentation on how to use and integrate other applications with their products, as well as there being plenty of other people who have done similar projects and posted tutorials or just share their experiences online.

R-3	Technical Issues: there may be technical issues that arise during the development or implementation of the music livestream bot. These may include server crashes, streaming delays, or software bugs.	Assure the presence of a robust technical team that can quickly address and resolve any issues that may arise.
R-4	Security Breaches: As the livestream music bot will likely involve the use of sensitive user data such as payment information, there is a risk of security breaches.	Implement strong security measures and protocols to ensure the safety of user data.
R-5	Competition: There may be other music livestream bots or similar services that emerge in the market, which can potentially impact the success of the project.	Regularly monitor competitor activity and make strategic adjustments to the music livestream bot as needed to stay competitive.
R-6	User adoption: There is a risk that the music livestream bot may not be adopted by a large enough user base to make the project financially viable.	Conduct thorough market research to identify potential users and develop targeted marketing strategies to reach them.
R-7	Legal and regulatory compliance: The music livestream bot must comply with various legal and regulatory requirements, such as copyright laws and payment processing regulations.	Have a legal team that can ensure compliance with all relevant laws and regulations.

12. Appendix A – Definitions, Acronyms, and Abbreviations

Term	Abbreviation / Acronym	Definition
Application Programming Interface	API	An Application Programming Interface (API) is a set of protocols, routines, and tools for building software applications that specify how software components should interact with each other.
Database	DB	A Database (DB) is an organized collection of data stored in a computer system.
Graphical User Interface	GUI	Graphical User Interface (GUI) is a type of user interface that allows users to interact with an application or system through visual elements such as icons, menus, and buttons.
BOT	BOT	A bot, short for robot, is a software application that performs automated tasks on behalf of users or other software applications.
Model View Controller	MVC	MVC: Model-View-Controller. A software design pattern that separates the representation of data from the user's interaction with it.
Software Requirements Specification	SRC	Software Requirements Specification. System software requirements are the specific software and hardware requirements necessary for a software application or program to run on a computer system.
Work Package	WP	A Work Package is a unit of work that is manageable and can be completed within a specific timeframe by one or more team members.
MP3	MP3	MP3 is a digital audio file format that is commonly used for storing and playing music.
Livestream	N/A	A Livestream is a live broadcast of an event over the internet, allowing viewers to watch in real-time.
Image Generator	N/A	An Image Generator is a tool or software that creates images automatically or based on specific parameters or rules.
UI	UI	The point of interaction between a user and a software application.
UX	UX	The overall experience a user has while interacting with a software application.

Team M

Music Livestream Bot
Software Requirements Specifications (SRS)

Version 3.0

Document Preparation

Name	Role	Approval (Signature)	Approval Date
Ryan Grieb	Editor	Ryan Grieb	2/24/2023
Anthony Tran	Product Manager	Anthony Tran	2/24/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	2/24/2023
Ryan Seidel	Software Tester	Ryan Seidel	2/24/2023

Document Approvals

Name	Role	Approval (Signature)	Approval Date
Anthony Tran	Project Design Manager	Anthony Tran	2/24/2023
Wyatt Wilson	Project Configuration Manager	Wyatt Wilson	2/24/2023
Ryan Seidel	Project Test Manager	Ryan Seidel	2/24/2023
Ryan Seidel	Project Risk Manager	Ryan Seidel	2/24/2023
Ryan Grieb	Project Technical Manager	Ryan Grieb	2/24/2023
Anthony Tran	Product Manager	Anthony Tran	2/24/2023
Wyatt Wilson	Requirements Engineer	Wyatt Wilson	2/24/2023

Revision History

Date	Version	Description	Author
03/31/2023	2.0	Revision for iteration 2	Ryan Seidel
04/25/2023	3.0	Finalizing the document	Ryan Seidel

Table of Contents

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose of the Document

The purpose of this document is to define the functional and non-functional requirements of the Music Livestream Bot software. The document is aimed at the development team, project stakeholders, and other interested parties who require a comprehensive understanding of the bot's functionality, design, and technical specifications. The main objective is to ensure that the bot meets the needs and expectations of its users, stakeholders, and the project team, while operating with optimal efficiency and effectiveness.

1.2 Scope of the Document

This document outlines what the Music Livestream Bot can do and what it can't do. It's like drawing a line that separates what we're supposed to build and what we're not supposed to build. We're focusing on the bot's functions, like playing and searching for music, chatting with other users, and other related things. The document describes how the bot should perform and how it should be built, including any technical details and limitations.

But this document doesn't cover other things that are part of building the bot, like managing the project, testing the bot, making it available to users, or keeping it running after it's done. It also doesn't talk about any legal or regulatory issues that might come up.

1.3 References

- Project Team M Proposal Version 2.0 April 25, 2023
- Project Plan Version 2.0 April 25, 2023
- Project Design Documents Version 2.0 April 25, 2023
- Project Test Documents Version 2.0 April 25, 2023

1.4 Definitions, Acronyms, and Abbreviations

- API: Application Programming Interface. A set of protocols and tools used for building software applications.
- BOT: A bot, short for robot, is a software application that performs automated tasks on behalf of users or other software applications.
- DB: Database. A structured set of data held in a computer, typically accessed by software applications.
- GUI: Graphical User Interface. A visual way of interacting with a software application using graphical elements such as windows, icons, and buttons.
- LIVESTREAM - Livestream refers to the act of broadcasting real-time video or audio content over the internet.
- MVC: Model-View-Controller. A software design pattern that separates the representation of data from the user's interaction with it.
- SRS: Software Requirements Specification. System software requirements are the specific software and hardware requirements necessary for a software application or program to run on a computer system.
- UI: User Interface. The point of interaction between a user and a software application.
- UX: User Experience. The overall experience a user has while interacting with a software application.
- Image-Generator: An Image Generator is a tool or software that creates images automatically or based on specific parameters or rules.
- WP: Work Package. A Work Package is a unit of work that is manageable and can be completed within a specific timeframe by one or more team members.

2. Product Scope

The scope of the Music Livestream Bot product is to provide a platform for users to livestream music, search for music, and interact with other users. The bot will allow users to be able to listen to high quality streams and share their favorite music with others.

The bot should be designed to provide an engaging and interactive experience for users.

The following are out of scope for this product:

- Integration with third-party applications or services that are not specified in the requirements.
- The development of any hardware or software components not directly related to the Music Livestream Bot product.
- The creation of any additional features or functionality not specified in this document.

3. Product Users

3.1 Users

Product User Type	User Characteristics
Common User	Plays music, listen to livestreams and share links to the bot with others
Admin	A designated user who has access to the bot's administrative functions
Age	People aged 10 years and up interested have control over the music they listen to.
Music Enthusiast	These users are passionate about music and want to listen to their favorite songs and artists in high quality, while also engaging with other music enthusiasts.
DJs and Music Curators	These users are music professionals who want to automate their music streaming tasks and curate high-quality playlists for their audience.
Music Venue Owners and Event Promoters	These users are professionals in the music industry, such as music venue owners and event promoters, who want to automate their music streaming tasks and engage with their audience in real-time.
Music Educators	These users are music educators who want to automate their music streaming tasks and provide high-quality and engaging music education experiences for their students.
User tasks	Develop and test new features, troubleshoot technical issues

4. Use Cases

Use Case ID	Use Case Name	Brief Description
UC-1	Music Manager	This use case allows users to choose music for livestream.
UC-2	Image Hub	This use case allows user to choose image for livestream
UC-3	Image Manager	This use case allows user to generate unique images for livestream
UC-4	Live stream	This use case allows user to have the ability to stream to a RTMPS link
UC-5	Text Overlay	This use case allows user to overlay text of what the song name is and author
UC-6	Song Skip	This use case allows user to skip the next song in playlist
UC-7	Playlist Creator	This use case allows user to add or remove mp3 files for their playlist

UC-8	Playlist Import	This use case allows user to import from YouTube utilizing external programs such as Youtube-dl
UC-9	Audio Controls	This use case allows user to control volume, pause or play
UC-10	Chat Controls	This use case allows user to song requests or skip

5. Functional Requirements

5.1 Software Requirements

Requirement ID	Requirement Priority (High/Medium/Low)	Requirement
Req-Func-Sw-1	Medium	The system shall only allow authorized users to login the system.
Req-Func-Sw-2	High	The system saw provide a high-quality music stream
Req-Func-Sw-3	High	The system shall provide users' ability to vote for songs/skip
Req-Func-Sw-4	High	The system shall exchange information with the city weather services systems.
Req-Func-Sw-5	Medium	The system shall generate autonomous pictures based on song selection
Req-Func-Sw-6	High	The system shall play MP3 music streams in real-time, with minimal latency and buffering issues.
Req-Func-Sw-7	Medium	The system shall manage playlists, including adding, removing, and shuffling songs. It should also be able to support user-generated playlists.
Req-Func-Sw-8	Medium	The system shall allow users to interact with the livestream through chat and comments, including song requests, feedback, and reactions.
Req-Func-Sw-9	Medium	The system shall allow users to control the livestream, including start, pause, and stop. It should also support advanced features such as skip, rewind, and fast-forward.
Req-Func-Sw-10	Medium	The system shall allow users to customize their music listening experience, including choosing the bitrate, equalizer settings, and visual effects.

5.2 Hardware Requirements

Requirement ID	Requirement Priority (High/Medium/Low)	Requirement
Req-Func-Hw-1	Medium	The system requires at least a dual-core CPU with a clock speed of 2.0 GHz or higher is recommended. The bot may require more processing power depending on the number of concurrent users and music streams.

Req-Func-Hw-2	Medium	The system requires at least 4 GB of RAM is recommended to ensure that the bot can handle a large number of concurrent users and music streams. The more RAM available, the better the bot's performance will be.
Req-Func-Hw-3	Medium	At least 50 GB of free storage space is recommended to store the music files, playlists, and other data required by the bot. The storage space requirement may vary depending on the number of music files and playlists.
Req-Func-Hw-4	High	A stable and reliable internet connection is required to ensure that the bot can stream music in real-time without buffering or other issues.
Req-Func-Hw-5	High	A sound card or audio interface is required to ensure that the bot can play music streams in high quality.
Req-Func-Hw-6	High	The bot should be installed on a dedicated server to ensure that it can handle a large number of concurrent users and music streams

6. Quality Requirements

Requirement ID	**Requirement Priority** (High/Medium/Low)	**Requirement**
Req-Qual-Simp-1	High	The Music Live Bot shall have a simple and intuitive user interface for easy navigation.
Req-Qual-Sec-1	High	The Music Live Bot shall implement secure user authentication to prevent unauthorized access to the system.
Req-Qual-Main-3	Medium	The Music Live Bot shall be designed with maintainability in mind to ensure efficient updates and bug fixes.
Req-Qual-Perf-3	High	The Music Live Bot shall be optimized for high performance to ensure seamless streaming and user experience.
Req-Func-Sw-1	High	The Music Live Bot shall allow users to choose a random song from the playlist and prevent the chance of choosing the same song randomly up to a certain number of songs.
Req-Func-Sw-2	High	The Music Live Bot shall include an Image Hub feature, allowing users to view a library of images before the Image Generator feature.

Req-Func-Sw-3	High	The Music Live Bot shall include a Webcaster feature, allowing users to live stream video through a RTMPS link.
Req-Func-Sw-4	Medium	The Music Live Bot shall include an Image Manager feature, leveraging emerging AI technologies such as DALLE-2, and generating unique images to livestream. This feature shall integrate with the API's of the AI technology providers and allow end-users to select which images to display.
Req-Func-Sw-5	Medium	The Music Live Bot shall include a Song Skip feature, allowing users to skip to the next song in the playlist through the program GUI.
Req-Func-Sw-6	Medium	The Music Live Bot shall include a Playlist Creator feature, providing a graphical interface for users to create their own playlists and add or remove mp3 files.
Req-Func-Sw-7	Medium	The Music Live Bot shall include a Playlist Import feature, allowing users to import playlists from YouTube using external programs such as YouTube-dl.
Req-Func-Sw-8	Medium	The Music Live Bot shall include an Audio Controls feature, allowing users to control audio on the song played, such as volume, pause, play, and scrubber.
Req-Func-Sw-9	Medium	The Music Live Bot shall include a Chat Control feature, allowing users who watch the stream to control the livestream through the chat, including song requests and skips.
Req-Func-Sw-10	Medium	Display informational text over the provided image on the livestream. This text can contain the song name, and author.

7. Appendix A – Requirements Traceability

Requirement ID	Related Requirements	Test Case ID
Req-Func-Sw-1	Music Manager	Test-Case-1
Req-Func-Sw-2	Image Hub	Test-Case-2
Req-Func-Sw-3	Webcaster	Test-Case-3
Req-Func-Sw-4	Image Manager	Test-Case-4
Req-Func-Sw-5	Song Skip	Test-Case-5

Req-Func-Sw-6	Playlist Creator	Test-Case-6
Req-Func-Sw-7	Playlist import	Test-Case-7
Req-Func-Sw-8	Audio Controls	Test-Case-8
Req-Func-Sw-9	Chat Control	Test-Case-9
Req-Func-Sw-10	Text Overlay	Test-Case-10

8. Appendix B – Use Case 1 – Music Manager

Use Case ID	UC-1	
Use Case Name	Creating the Music Manager	
Brief Description	User will be able to listen to their music of choice	
Frequency of Use	High – 100%	
Priority	High	
Current Version	1.0	
Date of First Version	02/24/2023	
Date of Last Version	N/A	
Created By	High	
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The system is currently being developed and will have more qualities in the future.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Livestream Bot and that bot has a reservation system in place.	
Postconditions	The user has successfully made a reservation for the Music Livestream Bot	
Trigger	The user will be able to see the songs that they are listening to when playing the stream.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User requests song to be played	2. System searches for song name against database of music

	5. User wants to add a song to the queue 9. Song ends and more songs are the queue	3. If there is an item in the database with the same song name as one searched, it will return the mp3 file to be played along with an accompanying photo, else it will return nothing, and an error message will be shown. 4. If an mp3 file is found, it will be played alongside a photo on the application. 6. Song name will be searched for against database 7. if an item in the database with the same song name is found the database will return an mp3 file and a photo, if not it will return nothing and an error message will be displayed. 8. If song name is valid, the accompanying mp3 file and photo will be put in a queue data structure to be played after all the songs in front of it have been played. 10. The mp3 file and photo first in the queue will be given to the application to play the next song and display its photo while the queue data structure will pop the object containing the photo and mp3 file.
Alternate Scenarios	**User Actions**	**System Actions**
	9.a. Song ends and queue is empty	10.a.. System will issue a message alerting the user that there are no more songs in the queue.
Additional Notes (Constraints, etc.)		

9. Appendix B – Use Case 2 – Image Hub

Use Case ID	UC-2
Use Case Name	Accessing the Image Hub
Brief Description	User will be able to listen to their music of choice
Frequency of Use	High – 100%

Priority	High	
Current Version	1.0	
Date of First Version	03/31/2023	
Date of Last Version	N/A	
Created By	High	
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The Image Hub contains a variety of images for users to choose from	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Live Bot and has navigated to the Image Hub section	
Postconditions	The user has successfully accessed and selected an image from the Image Hub	
Trigger	The user wants to select an image to display during the live stream	
Main Success Scenario	**User Actions**	**System Actions**
	1. User navigates to the Image Hub section of the application	2. System displays the available images in the Image Hub
	3. User selects an image from the Image Hub	4. System retrieves the selected image from the Image Hub and displays it during the live stream.
	5. User confirms that the selected image is being displayed during the live stream.	6. System continues to display the selected image until the user selects a new one or the live stream ends.
Alternate Scenarios	**User Actions**	**System Actions**
	7.a. Image not selected	8.a. System will issue a message alerting the user that there is no image.
Additional Notes (Constraints, etc.)		

10. Appendix B – Use Case 3 – Webcaster

Use Case ID	UC-3	
Use Case Name	Webcasting the Livestream	
Brief Description	The Music Live Bot will be able to stream the video to a RTMPS link, such as the ones provided on YouTube/Twitch.	
Frequency of Use	High – 100%	
Priority	High	
Current Version	1.0	
Date of First Version	03/31/2023	
Date of Last Version	N/A	
Created By	High	
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The system is currently being developed and will have more qualities in the future. The Music Live Bot has a stable internet connection.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Livestream Bot and a stable internet connection. Postconditions	
Postconditions	The user has successfully streamed the Music Live Bot to a RTMPS link.	
Trigger	The user wants to stream the Music Live Bot to a RTMPS link.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User navigates to the Webcaster section of the application 2. User inputs the RTMPS link where the live stream should be webcasted 3. User confirms that the live stream is being webcasted to the specified RTMPS link 5. System initiates the webcasting of the live stream to the specified RTMPS link	4. System displays the Webcaster section of the application

		6. System continues to webcast the live stream to the specified RTMPS link until the user stops the webcasting process.
Alternate Scenarios	**User Actions**	**System Actions**
	7.a. User decides to stop webcasting the live stream	8.a. System stops the webcasting process and returns to the normal live streaming mode.
Additional Notes (Constraints, etc.)		

11. Appendix B – Use Case 4 – Image Manager

Use Case ID	UC-4	
Use Case Name	Image Manager	
Brief Description	The user can use an AI-powered Image Manager to generate unique images for live streaming and request specific images to display.	
Frequency of Use	High – 100%	
Priority	High	
Current Version	1.0	
Date of First Version	03/01/2023	
Date of Last Version	N/A	
Created By	High	
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The AI-powered Image Manager has been integrated into the Music Livestream Bot.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Livestream Bot. The AI-powered Image Manager has been integrated into the Music Livestream Bot.	
Postconditions	The user has successfully generated and displayed unique images during live streaming.	
Trigger	The user selects the "Generate Image" button.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User selects the "Generate Image" button.	

	2. User prompts the AI to generate an image with certain parameters. 3. User confirms that the generated image meets their expectations. 6. System displays the generated image to the user.	4. System sends the parameters to the AI through its API. 5. AI generates an image and returns it to the system through its API.
Alternate Scenarios	**User Actions**	**System Actions**
	7.a. User prompts the AI to generate a new image with different parameters. 9.a. AI generates a new image and returns it to the system through its API.	8.a. System sends the new parameters to the AI through its API. 10.a. System displays the new generated image to the user.
Additional Notes (Constraints, etc.)		

12. Appendix B – Use Case 5 – Song Skip

Use Case ID	UC-5
Use Case Name	Song Skip
Brief Description	The user is able to skip to the next song in the playlist during live streaming
Frequency of Use	High - 90%
Priority	High
Current Version	1.0
Date of First Version	03/31/2023
Date of Last Version	N/A
Created By	High
Last Update By	
Approved By	Ryan Grieb

Assumptions	The Music Live Bot is currently running and a playlist has been selected for live streaming	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Live Bot and that bot has a playlist playing The user has successfully logged in to the Music Live Bot	
Postconditions	The user has successfully skipped to the next song in the playlist	
Trigger	The user wants to skip to the next song in the playlist during live streaming	
Main Success Scenario	**User Actions**	**System Actions**
	1. User clicks the "Skip" button to skip to the next song in the playlist 2. The Music Live Bot skips to the next song in the playlist 3. The Music Live Bot plays the next song in the playlist	4. The Music Live Bot receives the "Skip" command from the user 5. The Music Live Bot skips to the next song in the playlist 6. The Music Live Bot plays the next song in the playlist
Alternate Scenarios	**User Actions**	**System Actions**
	N/A	
Additional Notes (Constraints, etc.)		

13. Appendix B – Use Case 6 – Playlist Creator

Use Case ID	UC-6
Use Case Name	Playlist Creator
Brief Description	The user creates a playlist by adding or removing mp3 files for their playlists.

Frequency of Use	Medium – 50%	
Priority	Medium	
Current Version	1.0	
Date of First Version	03/01/2023	
Date of Last Version	N/A	
Created By	High	
Last Update By		
Approved By	Ryan Seidel	
Assumptions	The system is currently being developed and will have more qualities in the future.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Livestream Bot and the application has a playlist creation feature.	
Postconditions	The user has successfully created a playlist.	
Trigger	The user wants to create a new playlist.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User navigates to the Playlist Creator section of the application. 2. User clicks on the “New Playlist” button. 3. User adds mp3 files to the new playlist. 4. User removes mp3 files from the new playlist. 5. User saves the new playlist. 6. User confirms that the new playlist has been successfully created.	7. System displays the available mp3 files. 8. System creates a new empty playlist. 9. System adds the selected mp3 files to the new playlist. 10. System removes the selected mp3 files from the new playlist. 11. System saves the new playlist. 12. System displays a confirmation message to the user.

Alternate Scenarios	**User Actions**	**System Actions**
	1.a. User decides to cancel the new playlist creation process.	2.a. System cancels the creation of the new playlist and returns the user to the main menu.
Additional Notes (Constraints, etc.)		

14. Appendix B – Use Case 7 – Playlist Import

Use Case ID	UC-7	
Use Case Name	Playlist Import	
Brief Description	User imports playlists from YouTube and Twitch to be played on the Music Live Bot	
Frequency of Use	Medium - 50%	
Priority	Medium	
Current Version	1.0	
Date of First Version	02/24/2023	
Date of Last Version	N/A	
Created By		
Last Update By		
Approved By	Ryan Seidel	
Assumptions	The user has access to the Music Live Bot and a stable internet connection	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has an account on YouTube or Twitch and has created a playlist to import	
Postconditions	The user's playlist has been imported and is ready to be played on the Music Live Bot	
Trigger	The user wants to import a playlist from YouTube or Twitch to the Music Live Bot	
Main Success Scenario	**User Actions**	**System Actions**

	1. User selects the "Import Playlist" button. 2. User enters the URL of the playlist from YouTube or Twitch.	3. System retrieves the playlist from the YouTube or Twitch API. 4. System extracts the song titles and artist names from the playlist. 5. System searches the database for mp3 files that match the song titles and artist names. 6. System adds the mp3 files to the Music Live Bot's playlist in the correct order. 7. System displays a message confirming the playlist import was successful.
Alternate Scenarios	**User Actions**	**System Actions**
	5.a. System does not find a match for a song title and artist name.	5.a. System displays a message indicating the missing song and suggests the user add it to the Music Live Bot's playlist manually.
Additional Notes (Constraints, etc.)		

15. Appendix B – Use Case 8 – Audio Controls

Use Case ID	UC-8
Use Case Name	Audio Controls
Brief Description	The user can control the audio while the song is being played.
Frequency of Use	High - 90%
Priority	High
Current Version	1.0

Date of First Version	03/01/2023	
Date of Last Version	N/A	
Created By		
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The system is currently being developed and will have more qualities in the future.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has access to the Music Live Bot and is currently playing a song.	
Postconditions	The user has successfully controlled audio on the song played.	
Trigger	The user wants to control audio on the song played.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User wants to control the audio on the current song played. 2. User adjusts the volume of the current song. 4. User pauses the current song. 5. User resumes the current song after pausing it. 6. User drags the scrubber to the desired point in the song.	3. System receives the user's request to control the audio on the song played. 2. System searches for song name against database of music 7. System adjusts the volume of the current song based on the user's input. 8.. System pauses the current song based on the user's input. 9. System resumes the current song after it was paused based on the user's input. 10. System plays the song from the point where the user dragged the scrubber.
Alternate Scenarios	**User Actions**	**System Actions**
	N/A	N/A

Additional Notes (Constraints, etc.)	

16. Appendix B – Use Case 9 – Chat Controls

Use Case ID	UC-9	
Use Case Name	Chat Controls	
Brief Description	This use case describes the scenario in which a user interacts with the chat feature to control the live stream.	
Frequency of Use	High – 70%	
Priority	High	
Current Version	1.0	
Date of First Version	03/01/2023	
Date of Last Version	N/A	
Created By		
Last Update By		
Approved By	Ryan Grieb	
Assumptions	The user has a stable internet connection and access to the Music Live Bot.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	The user has logged in and has access to the live stream.	
Postconditions	The user has successfully used the chat feature to control the live stream.	
Trigger	The user wants to make a request or suggestion for the live stream.	
Main Success Scenario	**User Actions**	**System Actions**
	1. User types in chat message requesting a song or skip.	2. The message is sent to the system. 3. The system acknowledges the message and adds it to the queue. 4.The song or skip is played/displayed during the live stream.
Alternate Scenarios	**User Actions**	**System Actions**
	N/A	N/A

Additional Notes (Constraints, etc.)	

17. Appendix B – Use Case 10 – Text Overlay

Use Case ID	UC-10	
Use Case Name	Text Overlay	
Brief Description	Allows users to display text overlays on the music stream.	
Frequency of Use	Medium – 50%	
Priority	Medium	
Current Version	1.0	
Date of First Version	02/24/2023	
Date of Last Version	N/A	
Created By		
Last Update By	Anthony Tran	
Approved By	Anthony Tran	
Assumptions	The system is currently being developed and will have more qualities in the future.	
Primary Actor	User	
Secondary Actor/s	N/A	
Preconditions	User is logged in and has access to the music stream interface.	
Postconditions	Text overlay is displayed on the music stream.	
Trigger		
Main Success Scenario	**User Actions**	**System Actions**
	1. User selects "Text Overlay." 2. User inputs text and customization options. 3. User confirms the overlay.	4. System processes the text and customization options. 5. System renders the text overlay on the music stream. 6. System displays a confirmation message.
Alternate Scenarios	**User Actions**	**System Actions**
	1. Song ends and queue is empty. User selects "Text Overlay." 2. User inputs invalid text or customization options. 3. User confirms the overlay	

		4. System detects invalid text or customization options. 5. System displays an error message. 6. System prompts the user to correct the issue.
Additional Notes (Constraints, etc.)		

Team M

Music Livestream Bot
Software Design Document (SDD)

Version 3.0

Document Preparation

Name	Role	Approval (Signature)	Approval Date
Ryan Grieb	Editor	Ryan Grieb	3/03//2023
Anthony Tran	Product Manager	Anthony Tran	3/03/2023
Ryan Seidel	Software Tester	Ryan Seidel	3/03/2023

Document Approvals

Name	Role	Approval (Signature)	Approval Date
Anthony Tran	Project Design Manager	Anthony Tran	3/03/2023
Ryan Seidel	Project Configuration Manager	Ryan Seidel	3/03/2023
Ryan Seidel	Project Test Manager	Ryan Seidel	3/03/2023
Ryan Seidel	Project Risk Manager	Ryan Seidel	3/03/2023
Ryan Grieb	Project Technical Manager	Ryan Grieb	3/03/2023
Anthony Tran	Product Manager	Anthony Tran	3/03/2023
Anthony Tran	Requirements Engineer	Anthony Tran	3/03/2023

Revision History

Date	Version	Description	Author
03/03/2023	1.1	Initial version	Team M
04/07/2023	2.0	Revision	Team M
04/25/2023	3.0	Finalizing the document	Ryan Seidel

Table of Contents

Software Design (SDD)

1. Introduction

1.1 Purpose of the Document

This document serves to provide a detailed software design for the Music Livestream Bot project. Its goal is to present a comprehensive overview of the system's architecture, design, and implementation, including key decisions, limitations, modules, and user interface considerations. By serving as a guide for the development team and stakeholders, this document will facilitate the project's successful development, testing, deployment, and maintenance.

1.2 Scope of the Document

This document outlines the system's architecture, design patterns, and user interface, and includes guidelines for testing, deployment, and maintenance. The document is intended for the development team and stakeholders, covering all aspects of the software design and implementation, from conceptualization to deployment and maintenance. Its purpose is to ensure that the project's development is efficient and successful, providing a comprehensive overview of the system's functionalities, limitations, and design considerations.

1.3 References

- Team Project Proposal
- Project Vision Document
- Project Plan Document
- Software Requirements Specifications (SRS) Document

1.4 Definitions, Acronyms, and Abbreviations

See Appendix A.

2. Product Scope

The Music Livestream Bot API will provide a set of programming interfaces for developers to integrate live streaming, audience voting, and chat room functionalities into their own applications. The API will support scheduling of events and payment processing for artists and their audiences. Out of scope for this product are features related to music production, such as recording or editing. The API is designed to be scalable and flexible to accommodate future expansion based on user feedback and needs, and will be documented and supported for ease of use by developers.

3. System Design Decisions and Constraints

1. Audience Members:

- Who view and interact on the platform
- Have the ability to search and interact with features

2. Administrator:

- Users who manage the platform and ensure its smooth operation
- Manage/monitors the livestreams and perform system maintenance tasks

3.1 System Design Decisions

System Design Decision ID	Date	Design Decision	Reason

DD-1	3/03/2023	Architecture	The design decision for the system development is to use a client-server architecture. This will be complemented with a web-based user interface, which will provide users with an easy-to-use platform for accessing the system..
DD-2	3/03/2023	Live Streaming	The design decision for the system is to incorporate a third-party video streaming service. The purpose of this is to enable the system to provide high-quality live streaming capabilities to its users..
DD-4	3/03/2023	User Interface Design	The system's user interface will be designed to be intuitive, easy to navigate, and visually appealing, ensuring a positive user experience.
DD-5	3/03/2023	System Performance	The system will be designed with performance in mind, ensuring that it can handle a large number of concurrent users and traffic without compromising performance.
DD-6	3/03/2023	System Maintenance	The system will include tools and processes for maintenance, monitoring, and debugging to ensure smooth operation and quick resolution of any issues that arise.
DD-7	3/03/2023	Testing	The system will undergo rigorous testing throughout the development process to ensure that it meets all requirements and functions as intended.

3.2 System Design Constraints

System Design Constraint ID	Date	Constraint	Reason
DC-1	3/03/2023	Security	The API must comply with industry-standard security practices to ensure that user data and payment information is protected.

DC-2	3/03/2023	Compatibility	The API must be compatible with modern programming languages and frameworks to ensure ease of use and integration.
DC-3	3/03/2023	Scalability	The API must be designed to handle a large number of requests and traffic without compromising performance.
DC-4	3/03/2023	Documentation	The API must be thoroughly documented, including API endpoints, request/response formats, and error codes, to ensure ease of use and integration by developers.
DC-5	3/03/2023	Backward Compatibility	The API must maintain backward compatibility with previous versions to ensure that existing integrations do not break when updates are made.
DC-6	3/03/2023	Rate Limiting	The API must implement rate limiting to prevent abuse and ensure fair usage by developers.
DC-7	3/03/2023	Monitoring	The API must include monitoring and logging tools to track usage, identify issues, and ensure smooth operation.
DC-8	3/03/2023	Testing	The API must undergo rigorous testing throughout development to ensure that it meets all requirements and functions as intended.

4. Software Architectural Design

The software architecture of the Music Livestream Bot is described in this section.

5. Software Design - Modules/Classes

Module/Class Name	Module Short Name (If there is one)	Short Description of the Module/Class
CommandRunner	CR	Executes system commands
LivestreamPlayer	LP	The class plays the music video with sound/quality
MusicLivestreamBot	MLB	Takes the URL link and play the music playlist

PlaylistDownloader	PD	Downloads the playlist
SystemInfo	SI	Gets the system information that the api may need

5.1 CommandRunner

The CommandRunner (CR) class is used to execute system commands in a Java application. This module features a constructor that takes a command and arguments, and a run() method that executes the command using a ProcessBuilder.

5.2 LivestreamPlayer

The LivestreamPlayer (LP) class is designed to play music videos with high-quality sound. This class is used to stream and play music videos with reliable performance. Overall, the LivestreamPlayer class simplifies the process of streaming and playing music videos in a Java application.

5.3 MusicLivestreamBot

The MusicLivestreamBot (MLB) class is designed to take a URL link and play the music playlist in a Java application. This class is used to stream and play music playlists with reliable performance. Overall, the MusicLivestreamBot class simplifies the process of streaming and playing music playlists.

5.4 PlaylistDownloader

The PlaylistDownloader (PD) class is designed to download music playlists. This class is used to extract music from online livestreams and download them. Overall, the PlaylistDownloader class simplifies the process of downloading live streamed music playlists and provides users with easy access to their favorite music.

5.5 SystemInfo

The SystemInfo (SI) class is used to retrieve system information that an API may need in a Java application. This module features methods that return the operating system name, version, and architecture. Overall, the SystemInfo class simplifies the process of gathering system information and provides useful data.

6. User Interface Design

Navigation menu: The navigation menu could include options such as "Home," "Add Songs," "Add Images," and "Settings." Clicking on each option would take the user to a different section of the UI where they can perform specific actions.

Aesthetics: The UI design could include a dark or light color scheme that matches the theme of the host machine. We could use an API to determine if the user is in dark mode or not.

6.1 Home

Stream preview display: The image display area could show a screenshot/video of the current live stream. This can be updated every 5-10 seconds.

Play button: The play button could be labeled "Start Stream" and would begin playing the selected songs and displaying the selected images.

Pause button: The pause button could be labeled "Pause Stream" and would pause the music livestream.

Feedback messages: Feedback messages could appear at the bottom of the UI to let the user know the status of their actions. For example, when the user adds a song to the playlist, a message could appear saying "Song added to playlist." If the user selects an invalid image file, a message could appear saying "Invalid file type."

6.2 Add Songs

Song selection input field: The song selection input field could allow users to search for songs by artist, album, or title. As the user types, the field could display autocomplete suggestions to help them find the song they're looking for.

Playlist display: The playlist display area could show the list of selected songs in the order they will play. Each song could include information such as the artist, album, and length of the song.

6.3 Add Images

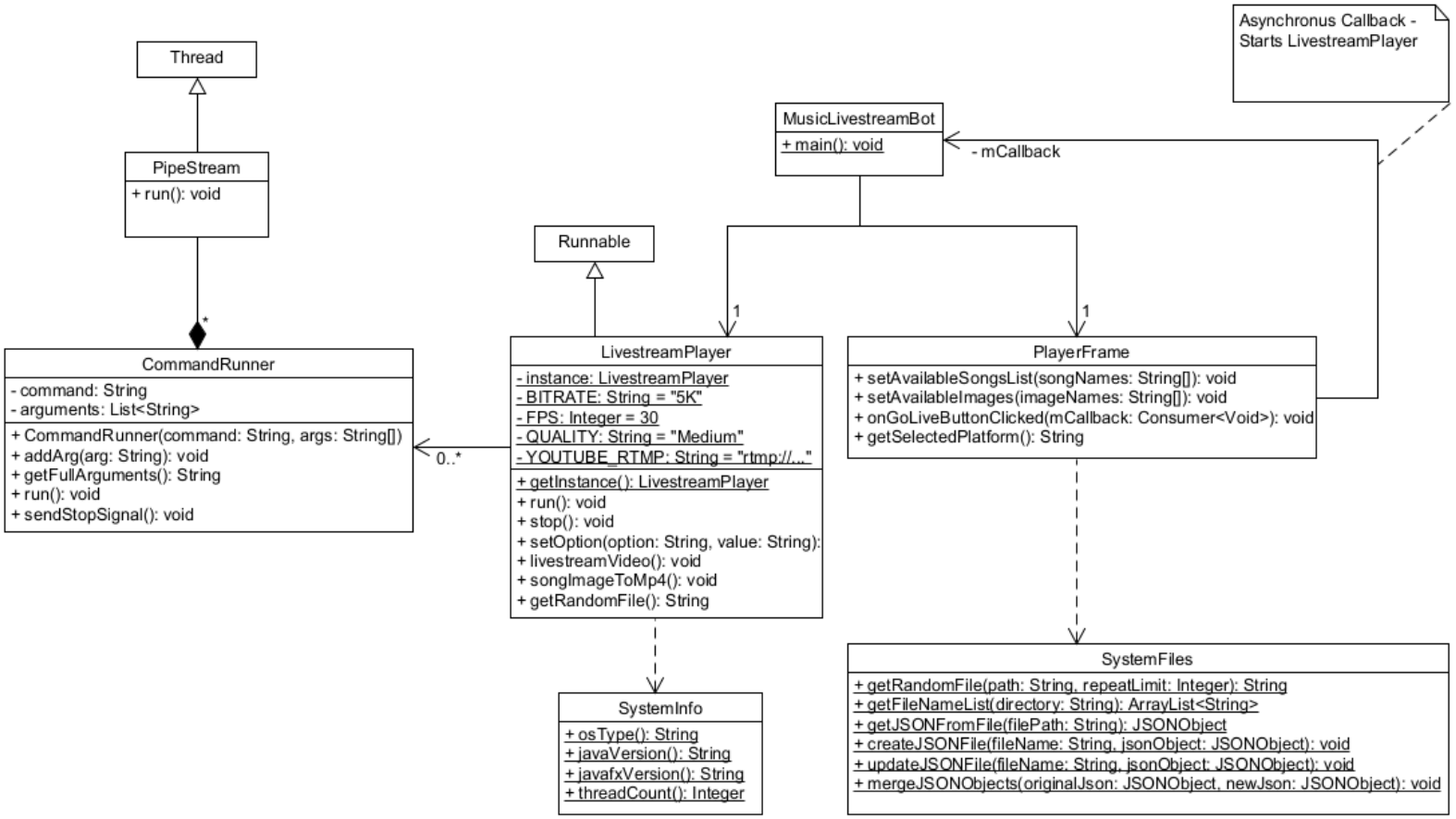
Image selection input field: The image selection input field could include options for uploading an image from the user's device or selecting from a pre-existing database of images. The UI could display a preview of the selected image before it is added to the playlist.

Random image generation info: The random image generation button could be labeled "Generate Random Image" and could use the GPT-3 API to generate an image based on a given theme or keyword.

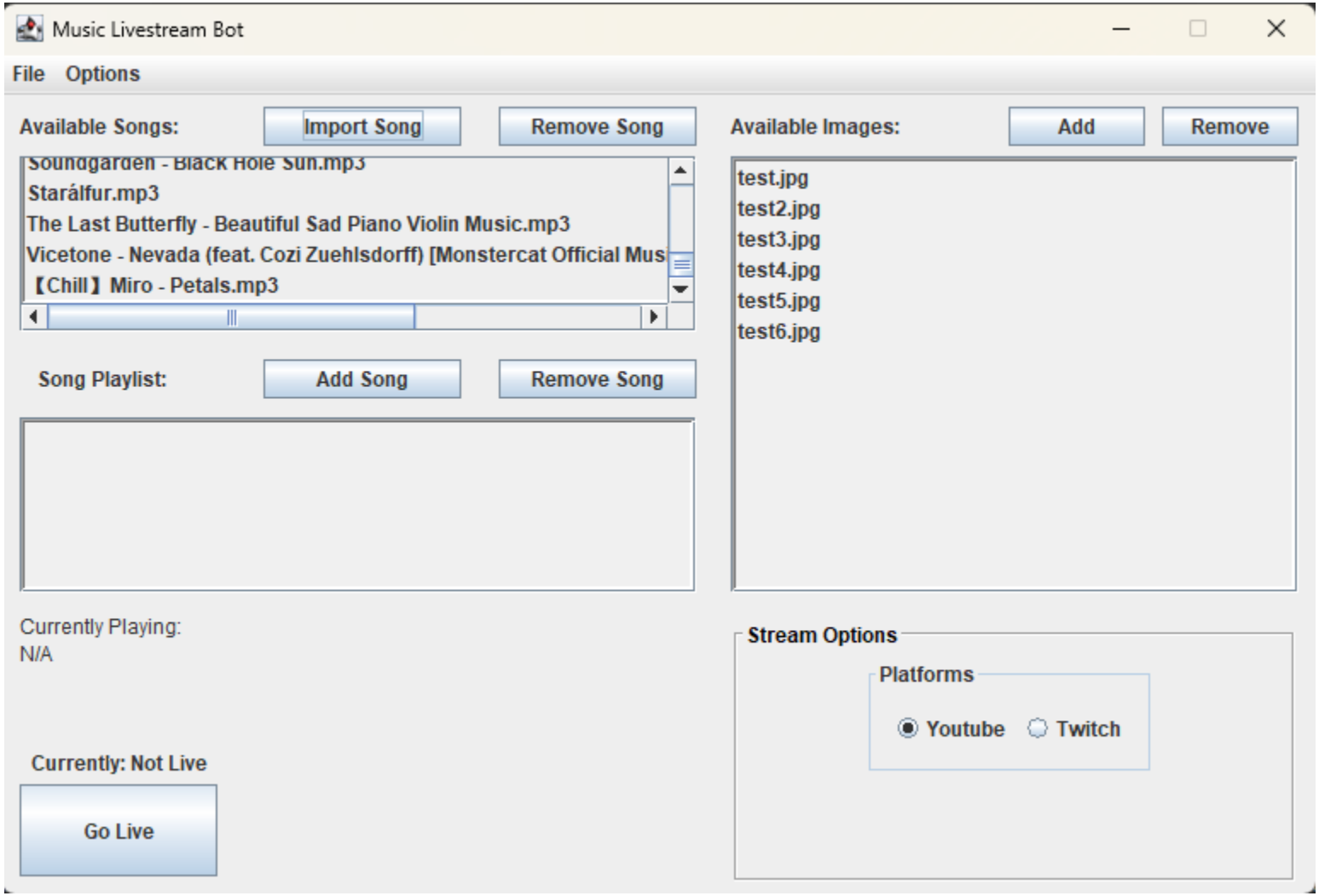
7. Appendix A – Definitions, Acronyms, and Abbreviations

Term	Abbreviation / Acronym	Definition
GPT-3		Generative Pre-trained Transformer.

8. Appendix B – Class Diagram of the Current Software System



9. Appendix C – User Interfaces



Team M

Music Livestream Bot
Software Test Document (STD)

Version 2.0

\

Document Preparation

Name	Role	Approval (Signature)	Approval Date
Ryan Grieb	Editor	Ryan Grieb	3/10/2023
Anthony Tran	Product Manager	Anthony Tran	3/10/2023
Ryan Seidel	Software Tester	Ryan Seidel	3/10/2023

Document Approvals

Name	Role	Approval (Signature)	Approval Date
Anthony Tran	Project Design Manager	Anthony Tran	3/10/2023
Ryan Seidel	Project Configuration Manager	Ryan Seidel	3/10/2023
Ryan Seidel	Project Test Manager	Ryan Seidel	3/10/2023
Ryan Seidel	Project Risk Manager	Ryan Seidel	3/10/2023
Ryan Grieb	Project Technical Manager	Ryan Grieb	3/10/2023
Anthony Tran	Product Manager	Anthony Tran	3/10/2023
Anthony Tran	Requirements Engineer	Anthony Tran	3/10/2023

Revision History

Date	Version	Description	Author
03/10/2023	1.1	Initial version	Team M
04/14/2023	2.0	Secondary version	Team M

Table of Contents

Software Test Document (STD)

1. Introduction

This document will cover the testing strategies, methodologies, and types of testing used to ensure that the software is fully functional, meets requirements, and is reliable. The document includes details on the test environment, testing tools, test cases, and test results.

1.1 Purpose of the Document

The purpose of this document is to provide a comprehensive guide to the testing process for the Music Livestream Bot application. Its goal is to outline the testing strategies and methodologies used to ensure that the software is fully functional, meets the requirements, and is reliable. By serving as a reference for the testing team and stakeholders, this document will facilitate the successful testing, deployment, and maintenance of the software.

1.2 Scope of the Document

The document provides details on the testing strategies, methodologies, and types of testing used to ensure that the Music Livestream Bot application is fully functional, meets the requirements, and is reliable. This includes information on the test environment, testing tools, test cases, and test results, making it a comprehensive guide to the testing process. The intended audience for this document includes the testing team, developers, and stakeholders involved in the project.

1.3 References

Team Project Proposal (ver 2.0)

Project Vision Document (ver 2.0)

Project Plan Document (ver 2.0)

Software Requirements Specifications (SRS) Document (ver 2.0)

Software Design (SDD) (ver 2.0)

1.4 Definitions, Acronyms, and Abbreviations

API: Application Programming Interface - A set of functions, protocols, and tools that allow developers to interact with an application, service, or platform.
SRS: Software Requirements Specification - A document that describes the intended purpose, requirements, and functionalities of a software system.
SDD: Software Design Document - A document that describes the architecture, components, and data flow of a software system.
STD: Software Test Document - A document that outlines the testing strategies, methodologies, and types of testing used to ensure that a software system is fully functional, meets requirements, and is reliable.
GUI: Graphical User Interface - The visual elements of an application, such as buttons, icons, and menus, that allow users to interact with the software.
UAT: User Acceptance Testing - A type of testing performed by end-users to verify that a software system meets their needs and requirements.
QA: Quality Assurance - The process of ensuring that a software system meets its intended purpose, requirements, and quality standards.
Test Case: A set of conditions or variables under which a tester will determine whether a system under test satisfies the requirements and works correctly.
Test Scenario: A high-level description of a test case that includes the steps and conditions necessary to execute the test.

2. Product Scope

The Music Livestream Bot API will provide a set of programming interfaces for developers to integrate live streaming, audience voting, and chat room functionalities into their own applications. The API will support scheduling of events

and payment processing for artists and their audiences. Out of scope for this product are features related to music production, such as recording or editing. The API is designed to be scalable and flexible to accommodate future expansion based on user feedback and needs, and will be documented and supported for ease of use by developers.

3. Test Planning

Clear and measurable test objectives, such as achieving a certain level of user satisfaction or reducing the number of reported defects, should be identified and documented in alignment with the project's goals.

To ensure effective testing, it is important to identify and prioritize the most critical test scenarios related to game data collection, based on feature criticality and potential impact of defects.

To properly define the testing environment, it is essential to identify the required hardware and software configurations, such as browsers, operating systems, and devices that will be used for testing.

In order to conduct testing that is efficient and reliable, the testing approach should be defined based on the project's needs and objectives, which includes selecting an appropriate test methodology and identifying the necessary tools and techniques to support the testing process, whether it's manual or automated testing.

To ensure timely and budget-friendly testing with desired quality levels, it's crucial to allocate sufficient resources including personnel, tools, and infrastructure for resource planning.

3.1 Responsibilities

Role	Name	Responsibilities
Test Engineer	Ryan Seidel	Prepares the test cases under the direction of the test manager.
Developer	Ryan Grieb	Supports the test engineers during the tests. Fixes the defects/bugs if identified during tests. Stay in the loop for the testing process.
Test Manager	Anthony Tran	Plans and coordinates all testing activities. Manages the testing team and test engineers. Take corrective action related to the testing activities if needed. Informs the project manager and technical manager regarding the test progress.
Customer/User Representative	Anthony Tran	Approves software test document, reviews test cases, and supports the testing process if possible.
Requirements Manager	Ryan Grieb	Approves software test document, reviews test cases, and supports the testing process if possible.
Technical Manager	Ryan Seidel	Ensures that all teams (project team members) work in tandem to achieve a successful testing process.

3.2 Test Schedule

Test Case	Scheduled Date	Actual Date / Duration
TC-1	4/1/2023	4/1/2023– 2 hours
TC-2	4/2/2023	4/2/2023 – 3 hours
TC-3	4/7/2023	4/7/2023 - 2 hours

4. Test Preparation

This section covers various aspects of testing, including identifying and documenting test objectives, selecting critical test scenarios, defining the testing environment, allocating sufficient testing resources, creating test data, developing test scripts, and selecting appropriate testing tools and techniques. By following these steps and utilizing the necessary resources, the testing process will be comprehensive and effective in ensuring the quality of the Music Livestream Bot application.

4.1 Test Environment

To set up and use the Music Livestream Bot, we will need a computer or laptop with internet connectivity and sufficient processing power. The application is run by calling the JAR file from the run.bat file, so we will need to ensure that the computer or laptop has run.bat installed and functioning correctly. We will also require access to a YouTube or Twitch account to obtain a stream key for the bot. The bot software may have specific system requirements, and we may need to acquire specialized equipment such as a mixer. We will also need to install and configure the bot software on the computer or laptop and may require additional software such as a digital audio workstation (DAW) for creating or editing the music. The testing can take place on the same computer or laptop, or on a separate computer or device to simulate the user experience.

5. Test Cases

5.1 Test Cases

Test Case ID	Test Case Name	Description
TC-1	Livestream Options	Livestream options (Youtube, Twitch) direct the stream to the proper source
TC-2	Livestream Key	The program should prompt the user for a livestream key after the livestream option is chosen.
TC-3	Livestream Video	Once a stream key has been entered and a song playlist has been made, the user should press “Start Stream”, a livestream should begin.
TC-4	Image Picker	There should be a new random image every song.
TC-5	Song Audio	Song audio from the mp3 file should be playing from the livestream video.
TC-6	Song Skip	Program should be able to skip ahead and go back with songs present in the playlist.

6. Appendix A – Test Case to Requirements Traceability

Test Case ID	Requirement ID
TC-1	Req-Func-Sw-2 Req-Func-Sw-9 Req-Func-Hw-4 Req-Func-Hw-6 Req-Func-Sw-1 Req-Func-Sw-4
TC-2	Req-Func-Sw-2 Req-Func-Sw-9 Req-Func-Hw-4 Req-Func-Hw-6 Req-Func-Sw-1 Req-Func-Sw-4
TC-3	Req-Func-Sw-6 Req-Func-Sw-7 Req-Func-Sw-9
TC-4	Req-Func-Sw-5 Req-Func-Sw-4 Req-Func-Sw-10
TC-5	Req-Func-Sw-6
TC-6	Req-Func-Sw-3 Req-Func-Sw-7 Req-Func-Sw-9

7. Appendix B – Test Case 1 – Livestream Options

Test Case ID	TC-1
Test Case Name	Livestream Options
Brief Description	Give the option to the User to be able to choose Youtube/Twitch.
Planned Test Duration	4 hours

Current Version	2.0
Date of Last Version	03/20/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M
The version of Software Under Test	N/A
Test Engineer to Conduct the Test	Ryan Seidel
List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 1 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Select the desired streaming platform (YouTube or Twitch).	The system directs the livestream to the selected platform.	Pass

8. Appendix C – Test Case 2 – Livestream Key

Test Case ID	TC-2
Test Case Name	Livestream Key
Brief Description	This test case verifies that the Music Livestream Bot prompts the user for a valid livestream key before starting the livestream on selected platforms such as Twitch.
Planned Test Duration	6 hours
Current Version	2.0
Date of Last Version	3/25/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M
The version of Software Under Test	N/A

Test Engineer to Conduct the Test	Ryan Seidel
List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 2 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Provide valid authentication credentials for the YouTube or Twitch account.	The system successfully connects to the account.	Pass
3	Select the desired streaming settings such as bitrate, resolution, and audio quality.	The system applies the selected settings and optimizes the streaming performance.	Pass
4	Enter the stream key and start the livestream.	The system begins streaming audio/video to the selected platform without interruptions.	Pass.

9. Appendix C – Test Case 3 – Livestream Video

Test Case ID	TC-3
Test Case Name	Livestream Video
Brief Description	Once the key enters, it plays a video with the song playing
Planned Test Duration	8 hours
Current Version	2.0
Date of Last Version	04/03/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M
The version of Software Under Test	N/A
Test Engineer to Conduct the Test	Ryan Seidel

List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 3 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Select the desired streaming platform (YouTube or Twitch) and provide valid authentication credentials for the selected platform.	The system successfully connects to the selected platform.	Pass
3	Select the desired streaming settings such as bitrate, resolution, and audio quality.	The system applies the selected settings and optimizes the streaming performance.	Pass
4	Enter the livestream key for the selected platform and create a song or song playlist.	The system accepts the stream key and allows the user to create a song or song playlist.	Pass.
5.	Press the "Go Live" button.	The system starts the livestream without any errors and the music is streamed with the video.	Pass

10. Appendix C – Test Case 4 – Image Picker

Test Case ID	TC-4
Test Case Name	Twitch
Brief Description	This test case allows the user to be able to select an image to display in the livestream video.
Planned Test Duration	4 hours
Current Version	2.0
Date of Last Version	4/05/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M

The version of Software Under Test	N/A
Test Engineer to Conduct the Test	Ryan Seidel
List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 4 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Select the desired streaming platform (YouTube or Twitch) and provide valid authentication credentials for the selected platform.	The system successfully connects to the selected platform.	Pass
3	Select the desired streaming settings such as bitrate, resolution, and audio quality.	The system applies the selected settings and optimizes the streaming performance.	Pass
4	Enter the livestream key for the selected platform and create a song or song playlist.	The system accepts the stream key and allows the user to create a song or song playlist.	Pass.
5.	Press the "Go Live" button.	The system starts the livestream without any errors and the music is streamed with the video.	Pass
6.	Verify that a new random image is displayed for each song played during the livestream.	The system picks a new random image for each song played during the livestream.	Pass

11. Appendix C – Test Case 5 – Song Audio

Test Case ID	TC-5
Test Case Name	Song Audio
Brief Description	This test allows the user of the GUI, to be able to control the audio of the stream.
Planned Test Duration	5 hours
Current Version	2.0

Date of Last Version	04/08/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M
The version of Software Under Test	N/A
Test Engineer to Conduct the Test	Ryan Seidel
List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 5 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Select the desired streaming platform (YouTube or Twitch) and provide valid authentication credentials for the selected platform.	The system successfully connects to the selected platform.	Pass
3	Select the desired streaming settings such as bitrate, resolution, and audio quality.	The system applies the selected settings and optimizes the streaming performance.	Pass
4	Enter the livestream key for the selected platform and create a song or song playlist.	The system accepts the stream key and allows the user to create a song or song playlist.	Pass.
5.	Press the "Go Live" button.	The system starts the livestream without any errors and the music is streamed with the video.	Pass
6.	Verify that the audio from the selected mp3 files is playing during the livestream.	The system successfully streams the audio from the selected mp3 files during the livestream.	Pass

12. Appendix C – Test Case 6 – Song Skip

Test Case ID	TC-6

Test Case Name	Song Skip
Brief Description	This test makes sure the User can skip the song.
Planned Test Duration	4 hours
Current Version	2.0
Date of Last Version	04/10/2023
Created By	Ryan Grieb
Last Update By	Ryan Grieb
Approved By	Team M
The version of Software Under Test	N/A
Test Engineer to Conduct the Test	Ryan Seidel
List of Test Support Software	Not Applicable (N/A)
List of Test Support Hardware	N/A

Test Case 6 Scenario

Test Step Number	Input	Expected	Result (Pass/Fail)
1	Launch the Music Livestream Bot by running the run.bat file.	The application starts running without errors.	Pass
2	Select the desired streaming platform (YouTube or Twitch) and provide valid authentication credentials for the selected platform.	The system successfully connects to the selected platform.	Pass
3	Select the desired streaming settings such as bitrate, resolution, and audio quality.	The system applies the selected settings and optimizes the streaming performance.	Pass
4	Enter the livestream key for the selected platform and create a song or song playlist.	The system accepts the stream key and allows the user to create a song or song playlist.	Pass.
5.	Press the "Go Live" button.	The system starts the livestream without any errors and the music is streamed with the video.	Pass
6.	Verify that the audio from the selected mp3 files is playing during the	The system successfully streams the audio from the selected mp3 files during the	Pass

	livestream.	livestream.	
7.	Verify that the user can control the order of the song playlist.	The user can control the order of the song playlist during the livestream.	Pass