Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4316: Senior Design I
Fall 2018

The Under Achievers
Synthify

Dominic Young
Endy Pluviose
Kolten Sturgill
Mary Huerta
Mitchel Smith
Minh-Quan Nguyen

Revision History

Revision	Date	Author(s)	Description
0.1	11.30.2018	DY, EP, MH, KS, MS, QN	document creation

CONTENTS

List of Figures

1 INTRODUCTION

Synthify will get playlists from a user from each platform to bring them all together and have the playlists unified under one service. The service will auto-update after a certain amount of time has passed to make sure it's consistent with any playlist changes that the user may have done on the respective platform. The "look and feel" of Synthify is similar to Spotify's dashboard. Users should expect to listen to music from each platform they have signed up for.

2 System Overview

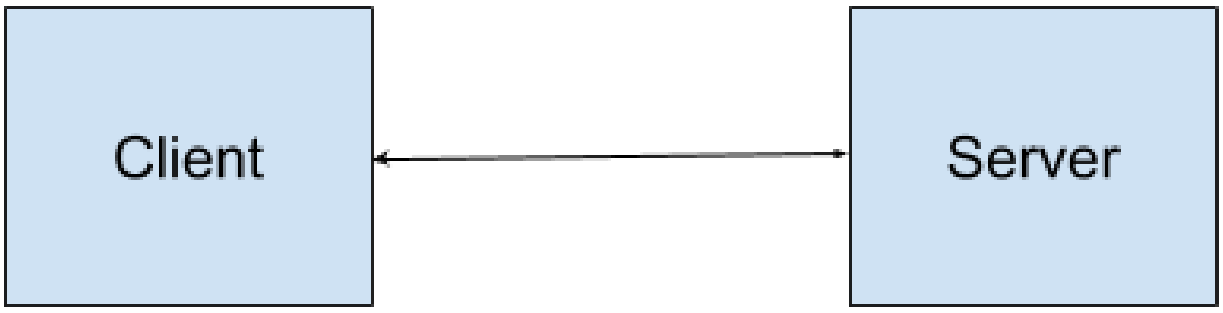

Figure 1: Breakdown of the Application Parts

2.1 Server Description

This layer will contains the resulting data and pulls from 3rd parties which will direct the data to the Client. The persistence frameworks will contain information such as the users log in credentials, previous queries to said music services, and play lists created by the user.

2.2 Client Description

The client will contain redux with will send data to the major viewable interfaces. These components will contain song and playlist information sent down from the server. The user can search for playlists.

3 Subsystem Definitions & Data Flow

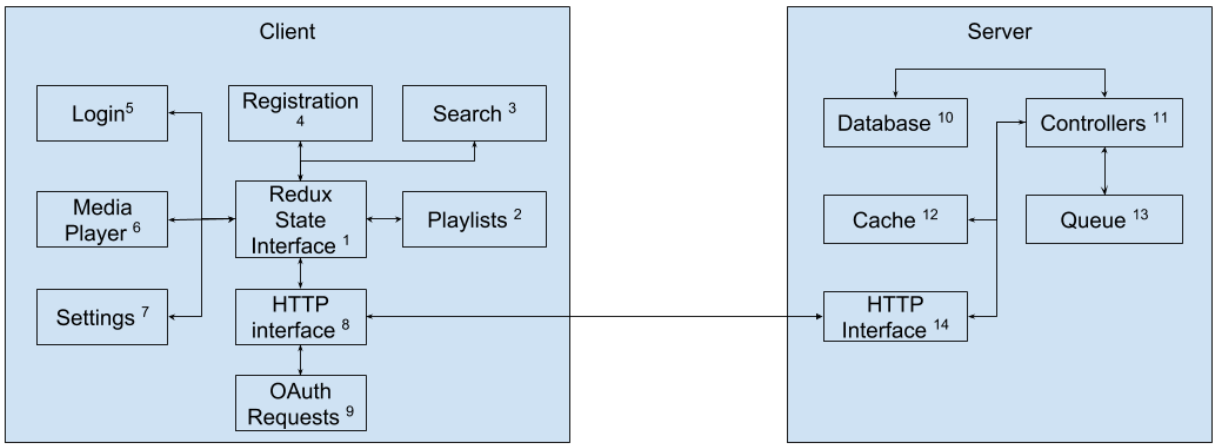

Figure 2: A simple data flow diagram in the form of the client and the server

4 CLIENT

The client is broken down to different components. The main component for the client will be the Redux state interface, which will keep track of states happening in the application. Other components include logging in as an existing user, registering as a new user, searching through your content once you have connected your accounts from different music platforms, the media player itself, user settings, playlists, oauth, and finally the interface, that will bring it together with the server.

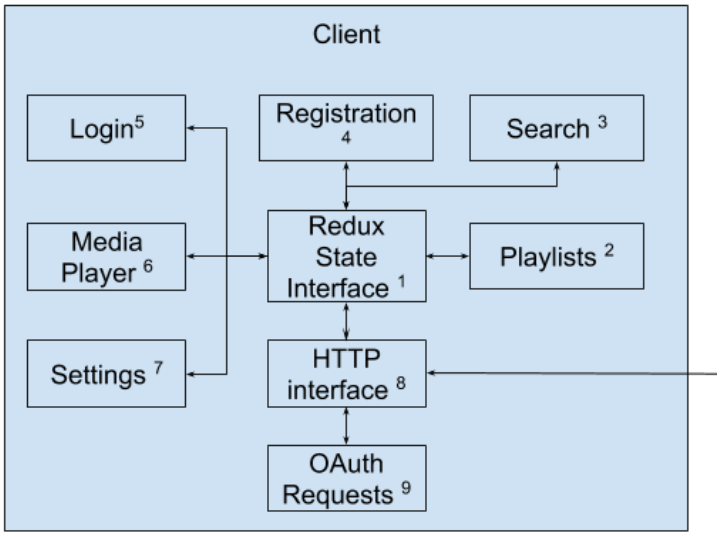

Figure 3: Client Subsystem

4.1 Redux

This subsystem allows the Client Layer to interact with the HTTP Interface. It will give information to all the different subsystems.

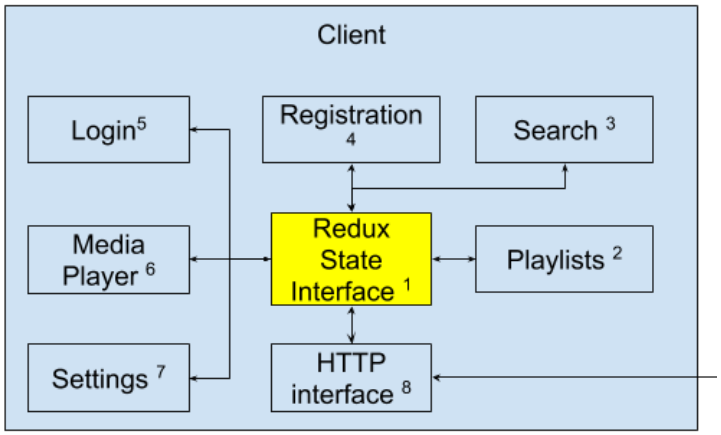

Figure 4: Redux subsystem

4.1.1 Assumptions

We are assuming the user is using Google Chrome, and the user does not require any accessibility needs for browsing/searching.

4.1.2 Responsibilities

The Redux Interface responsibilities include logging in and registering users, organizing the results from the search, and creating/editing/deleting the playlists from the various third party sources (e.g. Spotify, YouTube).

4.1.3 Subsystem Interfaces

Table 2: Redux interfaces

ID	Description	Inputs	Outputs
#1	Playlists	N/A	List of song information from specific Playlist
#2	Search	Search Term	Playlists
#3	Registration	E-mail Password Name	N/A
#4	Login	E-mail Password	N/A
#5	Media Player	Paused Volume	Song Functionality
#6	Settings	N/A	Output 1

4.2 Playlist

This subsystem will contain all the information on a users playlists and display it to them.

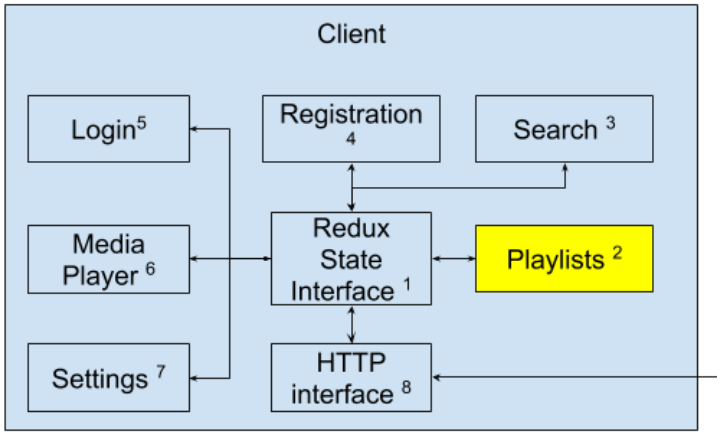

Figure 5: Playlist subsystem

4.2.1 Assumptions

One assumption we have made is that in order to display playlists the user will need to have at least one account connected to Synthify.

4.2.2 Responsibilities

The primary responsibility of this subsystem is to provide the user an easy way to view their playlists

4.2.3 Subsystem Interfaces

Table 3: Playlist interfaces

ID	Description	Inputs	Outputs
#01	Redux State Interface	User Id	Playlists for a user

4.3 SEARCH

This subsection will allow users to search for playlists on their music accounts through Synthify.

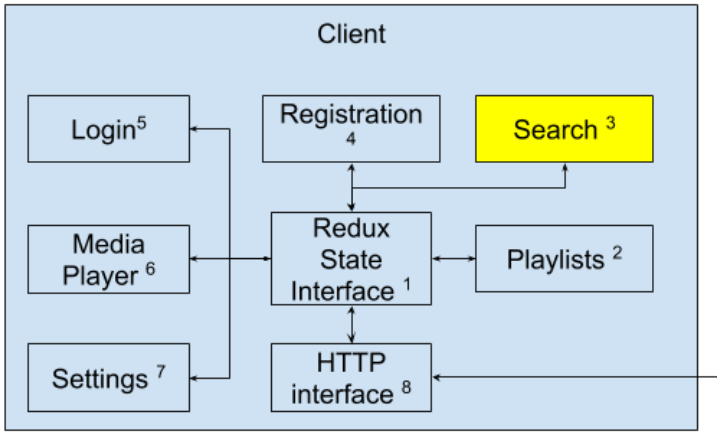

Figure 6: Search subsystem

4.3.1 ASSUMPTIONS

We are assuming that the user has at least one account connected to Synthify.

4.3.2 RESPONSIBILITIES

Responsibilities include displaying results (playlists based on user's search term) for the users and a search bar allowing a user to type in their search term.

4.3.3 SUBSYSTEM INTERFACES

Table 4: Search interfaces

ID	Description	Inputs	Outputs
#1	Redux State Interface	Search Term	List of playlists

4.4 Login

This subsystem is a graphical user interface that allows a user to log into Synthify

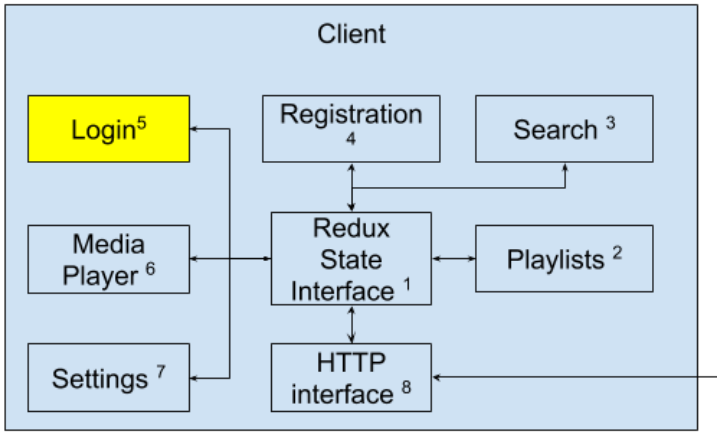

Figure 7: Login subsystem

4.4.1 Assumptions

One assumptions made is that the user will have an account in order to login.

4.4.2 Responsibilities

The login page will be responsible for recording the user's email and password, which will be sent to the Server subsystem via the HTTP interface.

4.4.3 Subsystem Interfaces

Table 5: Login interfaces

ID	Description	Inputs	Outputs
#01	User found	User's E-mail User's Password	HTTP 200 OK, JSON object containing information
#02	User not found or password incorrect	Invalid Login E-Mail Incorrect Password	HTTP 404, JSON Object containing error message

4.5 Media Player

This subsystem is apart of the graphical interface that allows the user to play/stop a song streaming from a third party music service.

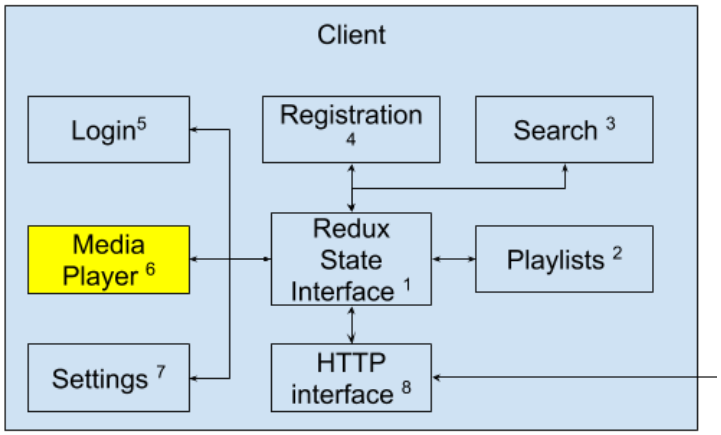

Figure 8: Media player subsystem

4.5.1 Assumptions

Assuming that the URL's that are sent from the third party music services are valid and are able to be streamed without any geographic restrictions.

4.5.2 Responsibilities

The Media Player subsystem is responsible for displaying the albums artwork (if available), and loading and playing/stopping a song from a remote URL from a third party music services. It also displays the remaining time left in the song.

4.5.3 Subsystem Interfaces

Table 6: Media player interfaces

ID	Description	Inputs	Outputs
#01	Play song, HTTP request	Clicking Play	Displays Artwork, music plays through user's speakers

4.6 SETTINGS

The settings subsystem allows users to changed their user account information. A user can add more music accounts (SoundCloud, Spotify, etc), logout, and change the theme.

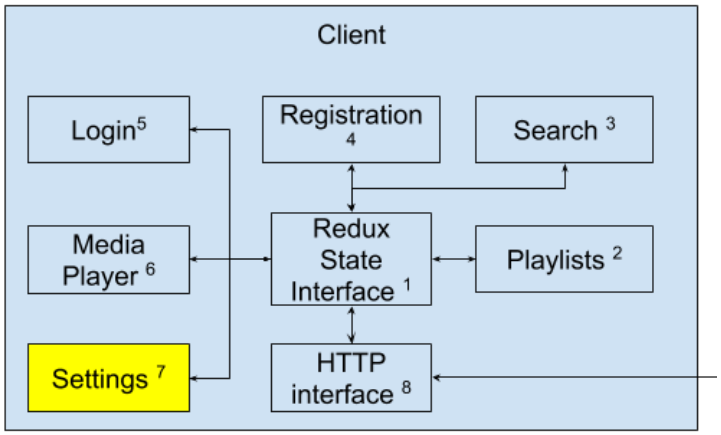

Figure 9: Settings subsystem

4.6.1 ASSUMPTIONS

One assumption made is that the user will be logged in before accessing the settings.

4.6.2 RESPONSIBILITIES

This subsystem's primary responsibility is to allow a user to change their user information. The settings subsystem will be able to interact the with the Redux State interface to change the users theme, log them out, and add a music account to their Synthify account.

4.6.3 SUBSYSTEM INTERFACES

Table 7: Settings interfaces

ID	Description	Inputs	Outputs
#01	Redux State Interface	Account setting to be changed user	Changed setting

4.7 HTTP Interface

The client side HTTP interface will use the built in Fetch JavaScript functionality to send and receive resources from and to the server.

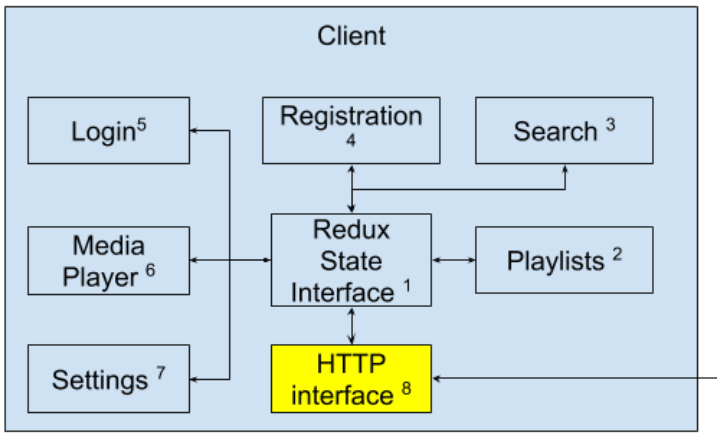

Figure 10: Client-side HTTP interface subsystem

4.7.1 Assumptions

We will assume we are using the latest version of NodeJS and React on the client side that contains the dependencies.

4.7.2 Responsibilities

This interface will be responsible for sending and receiving requests from the server, as well as interacting with Redux to keep the state of the application in sync. This interface will be involved in logging in, registering, streaming songs from the third party services, and sending updates/creation of playlists to the server.

4.7.3 Subsystem Interfaces

Table 8: Client-side HTTP Interface interfaces

ID	Description	Inputs	Outputs
#01	Description of the interface/bus	Input 1 Input 2	Output 1
#02	Description of the interface/bus	N/A	Output 1

5 SERVER

The server is broken down to being a database, controllers for the endpoints, a cache for preserving past results (so that we don't make unnecessary requests each time), a queue to keep track of the requests going out and what order they should be returned in, and finally the interface, that will bring it together with the client.

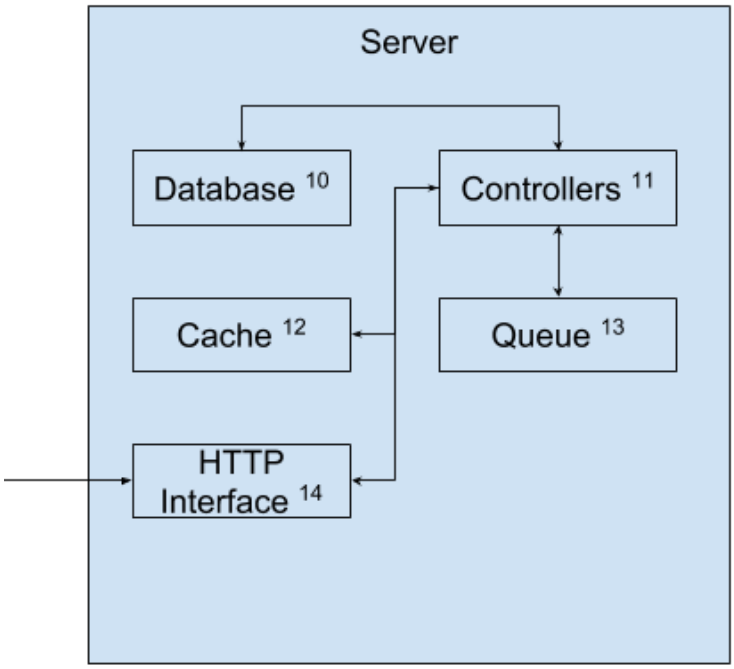

Figure 11: Server Subsystem

5.1 Database

The database will be used to keep track of user info which includes name, email, passwords, connections (to different services), user preferences, etc.

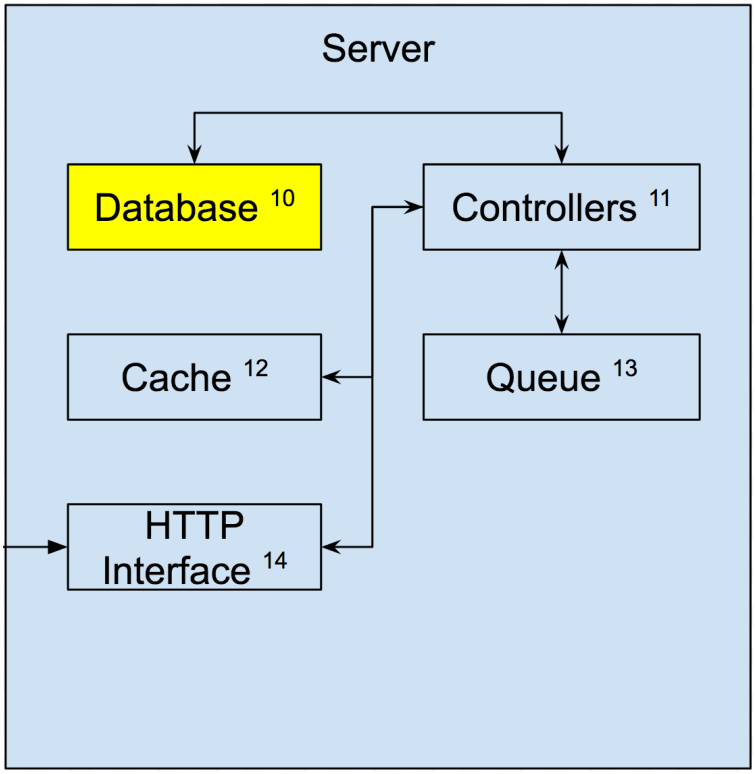

Figure 12: Database subsystem

5.1.1 Assumptions

N/A

5.1.2 Responsibilities

The responsibilities of the database will be to manage user data

5.1.3 Subsystem Interfaces

Table 9: Database interfaces

ID	Description	Inputs	Outputs
#01	Store new user	New user data	Confirmation query with new user data saved
#02	Returning user login credentials	Return user e-mail/password	Confirmation query with existing user data returned

5.2 Controller

This subsystem will be in charge of handling requests to the endpoints of the services we will be utilizing in Synthify (YouTube, Spotify, Soundcloud, etc.)

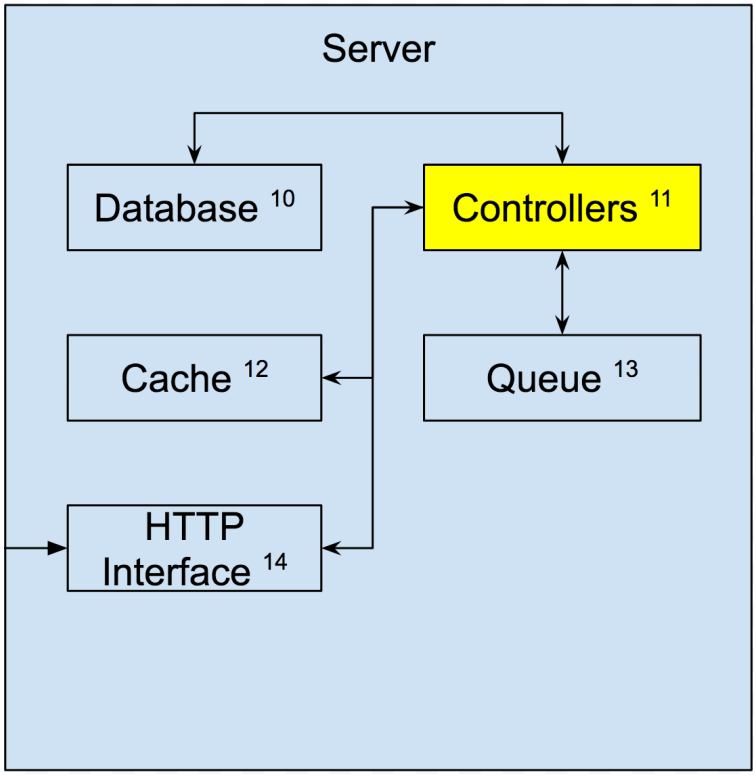

Figure 13: Controller subsystem

5.2.1 Assumptions

N/A

5.2.2 Responsibilities

Responsibilities include setting up the proper endpoints with the correct HTTP requests (GET, POST, etc.) to other subsystems. The controller will directly interact with the database, queue, cache, and HTTP Interface subsystems.

5.2.3 Subsystem Interfaces

Table 10: Cache interfaces

ID	Description	Inputs	Outputs
#01	Get user playlists	Connect spotify account	User's playlists are retrieved
#02	Get user info	N/A	N/A

5.3 Cache

The cache will be utilized so that we do not have to make unnecessary requests to the connections when we are grabbing a user content on the respective service

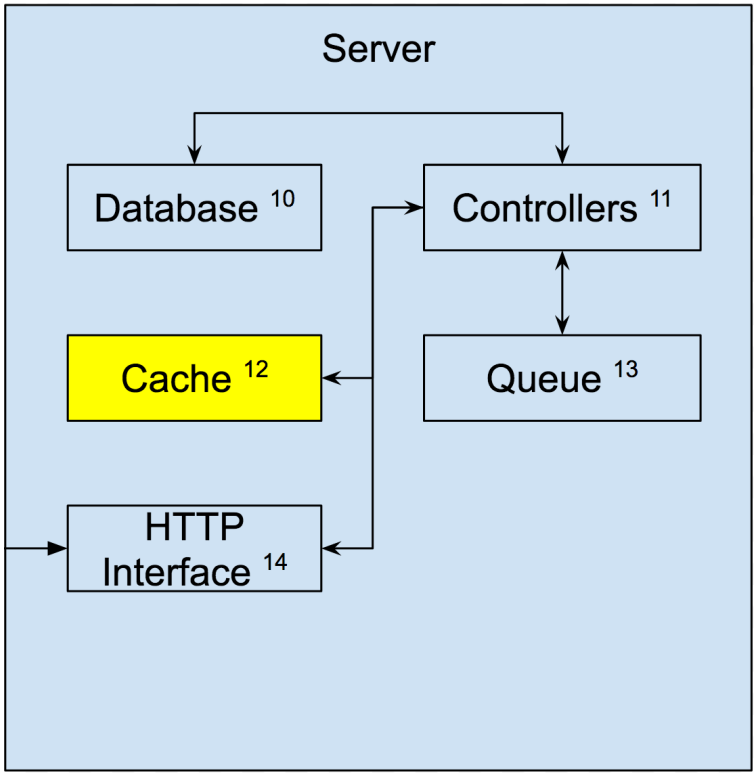

Figure 14: Cache subsystem

5.3.1 Assumptions

N/A

5.3.2 Responsibilities

To save previous results

5.3.3 Subsystem Interfaces

Table 11: Cache interfaces

ID	Description	Inputs	Outputs
#01	Make request	Retrieve song info	Store song info in cache

5.4 Queue

The queue will handle keeping track of requests made and their responses in the order that they are made

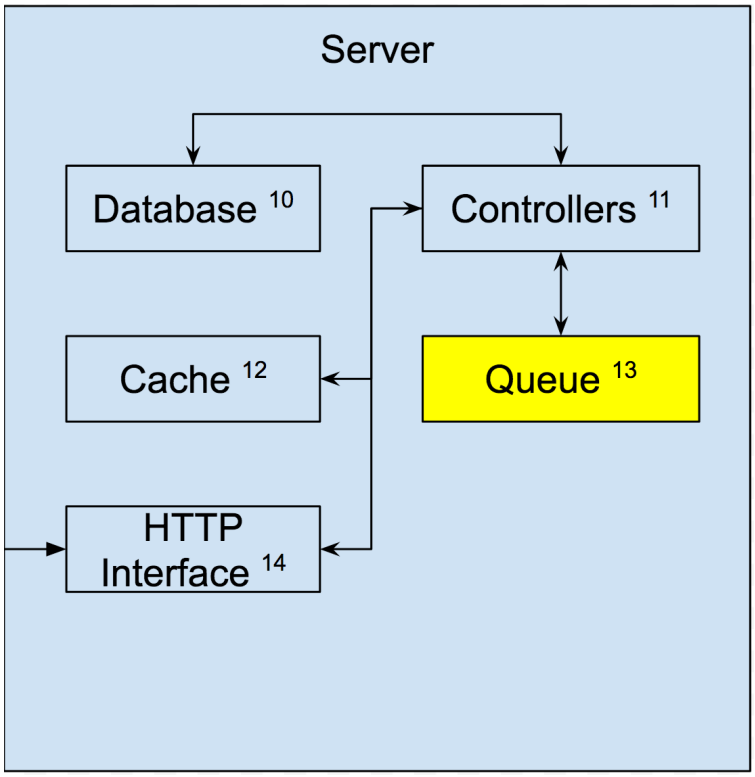

Figure 15: Queue subsystem

5.4.1 Assumptions

N/A

5.4.2 Responsibilities

Making sure the requests/responses are kept in the correct order

5.4.3 Subsystem Interfaces

Table 12: Queue interfaces

ID	Description	Inputs	Outputs
#01	Request is made from two different tabs	Songs	Songs info returned in order of request made

5.5 HTTP Interface

This interface will be the library used for the server that will direct the incoming request to the controllers via a HTTP/TCP socket.

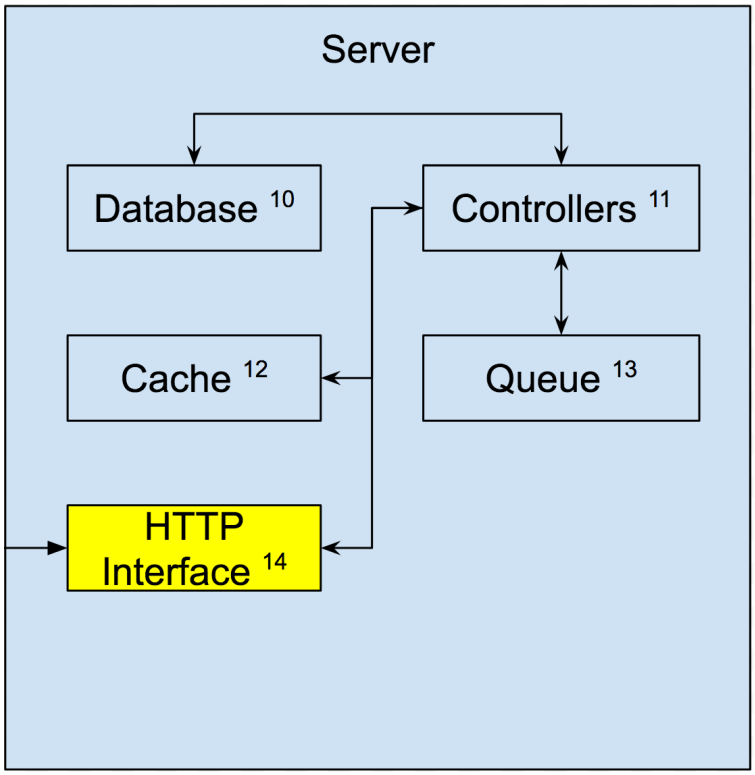

Figure 16: Server-side HTTP Interface subsystem

5.5.1 Assumptions

N/A

5.5.2 Responsibilities

The responsibility of the HTTP interface will include receiving HTTP requests via a chosen open source library, which will request a resource from the controller subsystem. This interface will also make use of Go's built in http package, which will make requests on to the other third party services.

5.5.3 Subsystem Interfaces

Table 13: Server-side HTTP Interface interfaces

ID	Description	Inputs	Outputs
#01	Request for login	JSON containing email and password	JSON response containing either users info or an error

References