This document provides more detail on how the movie ticketing system's architecture will be set up. It provides a logical sequence of how the user can interact with the system and how different pages/databases are linked together. This is a Client-Server Architecture where the theater company will host the server and provide the service of online ticket buying to clients.

Notes:

- The design and functionality here can change from version to version as per the needs of the movie ticketing system.
- Parts of the architecture that are directly referenced from the diagram are in bold.
- The words, "client", "user", and "customer" are used interchangeably.

Login Page:

This will be the first page a user sees. It will provide the option to either log in as an existing member or log in as a guest. If a user wishes to create a new account, they will have the option to sign up using the **Sign Up Page**. Login Page will also have the option for theater managers to log in using the **Admin Login Page.** The Login page *consists* of these two pages.
This page is responsible for starting a user "session" which will be active until the user either logs out or closes the tab. This page will internally and securely save some temporary data about the client that may be needed when the client pays for their tickets.
This page will be directly connected to, and have access to the **Customer Info Database**. Once identity is verified, the **Movie Selection Page** will open up.

Customer Info Database:

This will be one of the main hardware components of the movie theater ticketing system. This database will store name, email, payment info, purchase history, and rewards for each user account. The database will be secured to protect against any unauthorized access.
This database is internal and thus only connects to the **Login Page** for authentication and storing new user information. Any data that needs to be updated in this database will go through the **Login/Session Page.**

Movie Selection Page:

This page will act as the homepage for the website. It will list all necessary information like cinema locations, current movie listings, and movie schedules. This page is directly connected to the **Movie Info Database** to get information regarding the movies.
Once this page retrieves movie information from the database, users will have the option to select a movie. The page will then carry over the movie data to the **Seat Selection Page** where users can select their seats.

Movie Database:

This database stores information about the movies and showtimes for all cinema locations. While it will be internally implemented, most data will be publicly displayed on the **Movie Selection Page** so that users may choose what movies they want. Theater-specific data like seats and showtimes will be stored in this database and displayed on the **Seat Selection Page**.

This database also connects to the **IMDB Movie Database** for more information about movies.

IMDB Movie Database:
This database includes information that will help users decide what movie they want to select. It includes information such as movie ratings and reviews. It is a third party database that is not owned by the theater company. The **Movie Database** makes frequent calls to it and stores the information when new listings are posted. It is only connected to the **Movie Database** which then displays the information on the **Movie Selection Page**.

Seat Selection Page:
This page prompts the users to select a seat based on the theater location, movie, and timing preferences they previously selected. These preferences are carried over from the **Movie Selection Page**.
Data about available seats and other movie/theater information will be displayed on this page. The page will make frequent calls to the **Movie Database** so that it can stay updated about what seats, theaters, and timings are currently available.
Once a user has selected seats, they will have the option to confirm and pay. They will then be redirected to the **Payment Page**.

Payment Page:
This page is where the clients can pay for their tickets. This page will use different third-party API calls that can safely handle transactions. The API will take away some of the responsibility of handling transactions while offering clients with many payment options including Stripe, Paypal, and Apple Pay.
This page will take inputs from the **Seat Selection Page** to determine how much to charge the user. It will also be linked to the **Login Page** in order to get account and payment information that clients may have previously inputted.
The page will be connected to the **Confirmation Page**, where clients will be redirected to after they pay. This page will include showing and seat information carried over from the payment page.

Queue Page:
This page is where users will be directed if there is high traffic on the site for the same movie showing. Since many clients buying the same ticket can cause overbooking of the same type of seats, the Queue Page will allow people to pay one at a time. This page is implemented using Queue-It which is a service that provides companies the ability to integrate a virtual waiting room for their clients. It will be implemented in between the **Seat Selection Page** and **Payment Page**, and will only be in use during high traffic times.

Confirmation Page:
This page is the last one in the process of buying a ticket. It displays the information about movies and certain client information like name and email from the **Payment Page** to act as a confirmation for buying the ticket. It will give users the option to save a copy of this page, which will allow customers to dispute their tickets if a situation requires them to do so. It will also allow

users to log out so that their session ends, or redirect them to the Movie Selection Page if they choose to do so, in which case, their session will remain.
This page is linked to the **Login Page** so that it can retrieve contact information about clients. The page will then send a confirmation email to the client with their receipt.
It will also give data about this transaction to the **Login Page** so that the payment history field in the **Customer Info Database** can be updated.