# Theater Ticketing Software System

Software Design Specification

Version: 3.101.1

Oct 5, 2023

Group #1

Ryan, Nardos, Rhenz

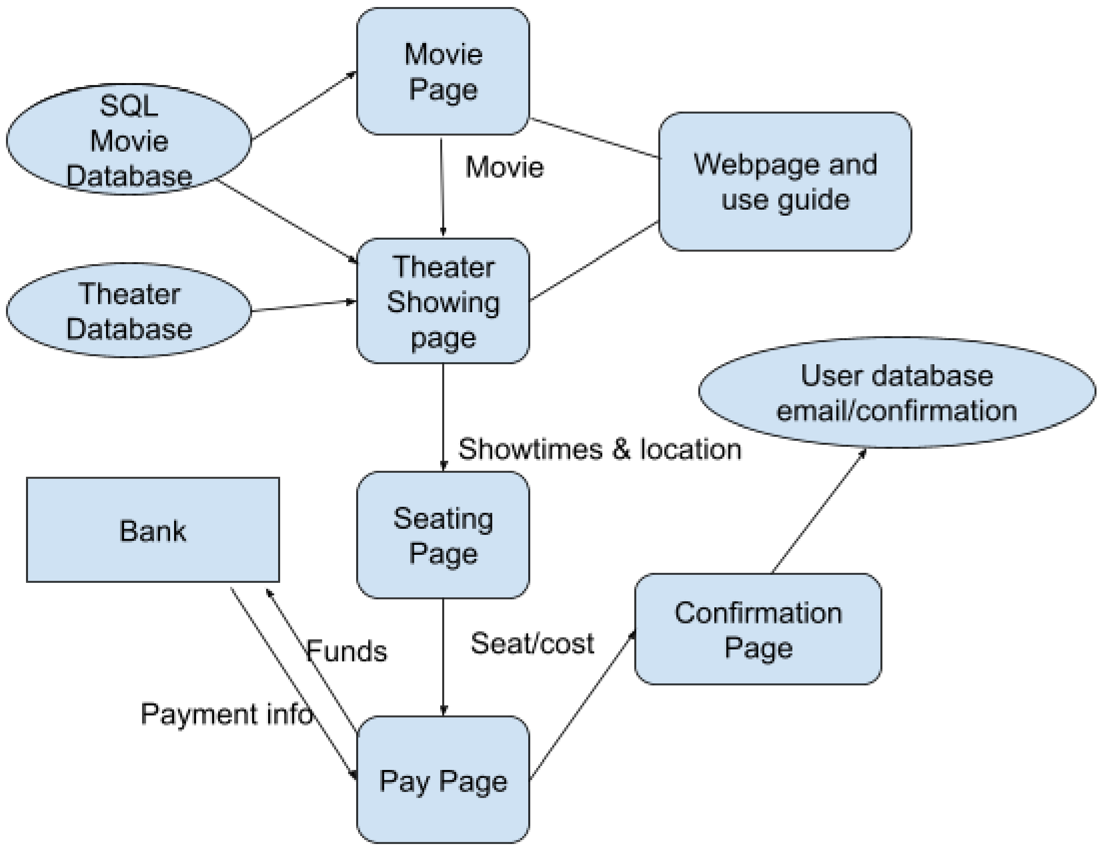
**Prepared for**

**CS 250 - Introduction to Software Systems**

**Instructor: Gus Hanna, Ph.D.**

**Fall 2023**

## Movie Theater Software Architecture Diagram



## Theater Ticketing Software System Architecture Diagram Description

We have several modules and databases that interact with each other depending on functionality.

### Modules

- Webpage and guide - welcomes and introduces customers to movies and ticket purchase availability.
- Movie page - displays movie by genre that draws from sql database
- Theater page - displays movie showtimes at specific theater that feeds from theater database

- Seat selection - displays seats at specific movie showtimes and theater
- Pay page - displays payment information required to book selected seats, confirms funds are available from the bank and proceeds to charge payment method.
- Confirmations page - displays confirmation of all the selections made, receipt of funds and sends it to email.

## Time

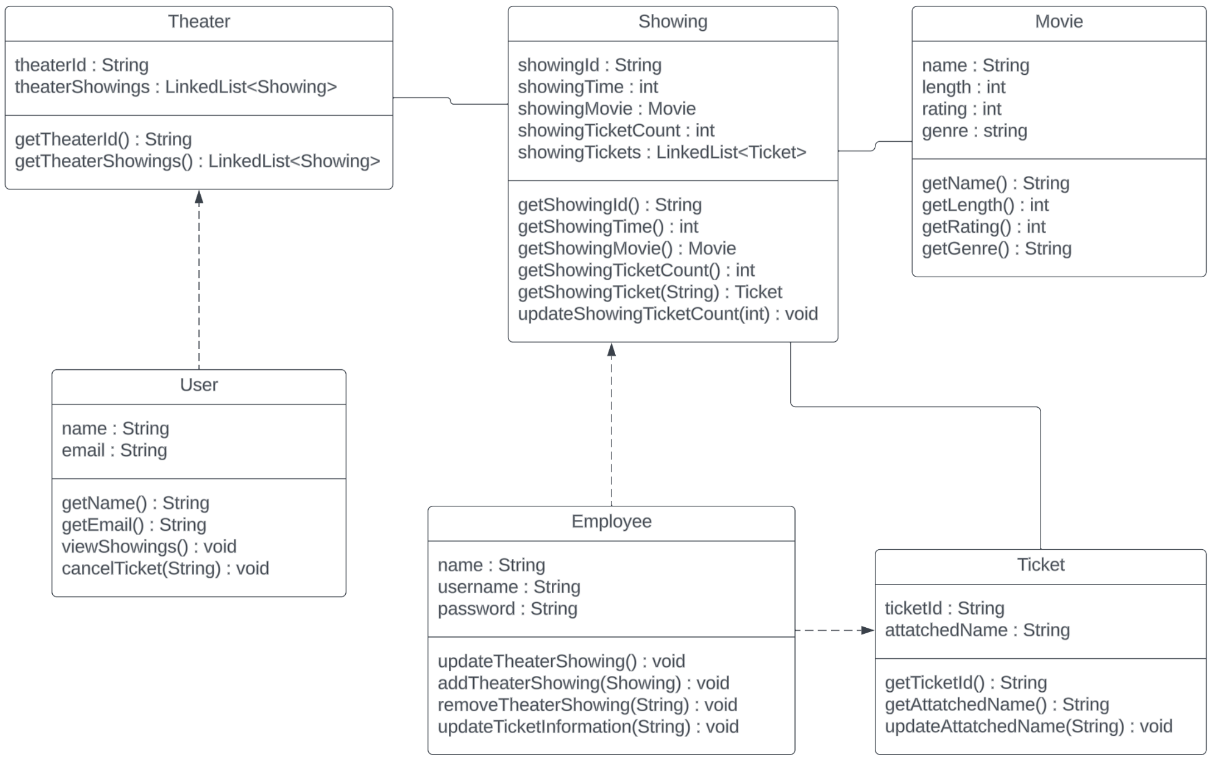
- The timeline allocated for this project is 3 months. There is flexibility to extend the timeline in accommodation of project needs.

## Teams

We will have 3 team member working on different aspects of the project

- Team 1  will have the responsibility of connecting the back end databases
- Team 2 - will work on the front end functionality.
- Team 3 - will work on the UI design to make sure it is user friendly.

## UML Class Diagram of System Operation



## Class Descriptions

### Employee

The employee class contains 3 attributes which represent the login information of the employee. The class interacts with the *showing* class and *ticket* class. They are given 3 operations relating to the *showing* class and 1 operation relating to the *ticket* class. The employee and update theater showings, add new theater showings, and update theater showings. The updateTheaterShowing() operation takes in a *showing* object as a parameter which is constructed within the operation. The removeTheaterShowing operation and updateTicketInformation operation take in a string parameter which represents the ID of the showing, from there the showing can be updated.

- Attributes
    1. Username: username of employee account
    2. Password: password of employee account
    3. Name: proper name attached to employee account
- Operations
    1. updateShowing: allows employee to update showing info such as showtime and showing moving
    2. addShowing: allows employee to add a new showing to a theater
    3. removeShowing: allows employee to remove existing showing from a theater
    4. updateTicketInformation: allows employee to adjust ticket information

### User

The *User* class seems to have two attributes which mainly represents the email of the user's account including their personal information like first and last name. They are given 2 operations which is getEmail which mainly retrieves the User's email address. And the getName operation retrieves the name of the User and a view object which allows the User to view all of the showings that are on in the ticketing system.

- Attributes
    1. Email : email of the user's account
    2. Name : first and last name of the user
- Operations
    1. getEmail: retrieve the Users email
    2. getName: retrieve the name of the Users
    3. viewShowings: allows Users to view all showings

## Theater

The Theater class contains two attributes which represent the theater id and theater showing. It interacts with *showing* class and *user* class. It has two operations that retrieve theater Id and showings list for the theater. The getThaterID takes in a string parameter.

- Attributes
    1. theaterId: unique identifier of the theater represented as a 4 character string
    2. theaterShowings: a linked list comprised of all showings at the specific theater
- Operations
    1. getTheaterId: retrieves theater ID
    2. getTheaterShowings: retrieves list of all showing for the theater

## Showing

The *showing* class contains 5 attributes that represent the unique information regarding the showing along with ticket information. It contains 5 get operations and 1 update operations. The getShowingId, getShowingTime, getShowingMovie, and getShowingTicketCount directly retrieves the related attribute information. The getShowingTicket takes a string as a parameter which represents the unique ID of the strings that is desired for retrieval, it then returns that ticket. The updateShowingTicketCount takes in an int as a parameter which represents the adjustment to the showingTicketCount attribute (i.e. -10 decrements the count by 10)

- Attributes
    1. showingID: allows to retrieve the unique identifier for time show movies
    2. showingTime: allows to retrieve the time movies of the shows
    3. showingMovie: contains the movie attached to the showing
    4. showingTicketCount: contains the number of remaining tickets for the showing
    5. showingTickets: contains a list of all tickets for the showing
- Operations
    1. getShowingId: retrieves showing ID
    2. getShowingTime: retrieves showing time
    3. getShowingMovie: retrieves movie played at the showing
    4. getShowingTicketCount: retrieves count of tickets remaining
    5. getShowingTicket: retrieves a specific ticket within it’s ticket list
    6. updateShowingTicketCount: updates the number of tickets

## Movie

The Movie class has four attributes that display information about the movies. It interacts with the *Showings* class. It contains four operation that retrieves the name, genre, length and rating of movie

- Attributes
    1. Name: shows movie Name
    2. Genre: genre of move
    3. Length: length of move in minutes
    4. Rating: rating of movie
- Operations
    1. getName: retrieves name of movie
    2. getGenre: retrieves genre of movie
    3. getRating: retrieves rating of movie
    4. getLength: retrieves length of movie

## Ticket

The *Ticket* class has two attributes which mainly represents a ticketID which identifies the person with the ticket and an attachedName that represents the user who purchased the ticket. And the class has 2 similar Operations which has the getTicketId() which is able to retrieve the ticket ID including the getAttachedName() which would be able to retrieve the name attached to the ticketing. And the updateAttachedName() Operation would show the ticketing of the attached name and update it.

1. ticketId: unique identifier of the ticket represented as a 16 character string
2. attachedName: name of user who purchased the ticket

- Operations
    1. getTicketId: retrieves ticket ID
    2. getAttachedName: retrieves name attached to ticket
    3. updateAttachedName: updates the attached name to ticket