Software Requirements Specification

for

Students' Auditorium Management Software (SAMS)

Version 1.0 approved

Prepared by

Rushil Venkateswar (20CS30045)
Rahul Mandal (20CS30039)
Sourabh Soumyakanta Das (20CS30051)

Indian Institute of Technology, Kharagpur

18th March 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

Management and scheduling of various events like booking tickets for shows through physical means has always been inefficient and hectic. So, an application of software interface can make this process more efficient.

1.1 Purpose

This document strives to provide a detailed description of the Students' Auditorium Management Software. It will explain the features and functionalities of the system, the constraints under which it will operate and how the system will react to external inputs. This document is designed for both the users and the developers of the system

1.2 Document Conventions

- Bulleted points have their brief idea underlined.
- Every requirement statement has its own priority.
- Every function in Section 4 is described in the form of 'x :: y'. This denotes that the y is a function of the class x.

1.3 Intended Audience and Reading Suggestions

Sections 2.4, 2.5, 2.6, 2.7, 3 are intended for users or specifically if you are an administrator and are interested in setting up the system for use at an auditorium. Future developers should refer to section 4 and the use-case and class diagrams at the very least although we'd recommend going through the entire document.

1.4 Product Scope

This software gives developers and students a portal for checking and booking of seats at the Auditorium. All information is stored internally. Software quickly finds out all the available shows, their dates, timings and available seats, students can choose from the available seats which they like. It reduces the time spent on booking tickets physically, and it also reduces crowding as students already know where they must be seated as a seat is allotted to them immediately after successful booking. As everything is systemized, different salespeople need not worry about how many tickets they have sold as every transaction by them will be stored and will be visible to the manager. Using this software can eliminate the complexity of booking a particular seat from two different salesperson manually as seats are getting updated as soon as they are booked.

This software is very easy to use in the back end, managers can easily schedule new shows with seats count instantly, and these will be visible to salesperson and students in no time. Which increases productivity.

It eliminates the need for manual accounting and record keeping, it gives nearly accurate results while eliminating physical accounting and written records.

1.5 References

Codemy Playlist for Python and Flask usage: link
https://www.python.org
https://pandas.pydata.org
https://www.python-excel.org/
https://stackoverflow.com/

2. Overall Description

2.1 Product Perspective

This software is designed to be able to add new events and modify existing events as per the requirements of the users (effectively an event manager). It also stores the transaction data which can be accessed later by the Show Manager. It provides an interface(website) for the end user to manage screening of movies and other events keeping the backend implementation details hidden from them. Different types of seats are also accommodated in the software and balance sheets are maintained as well.

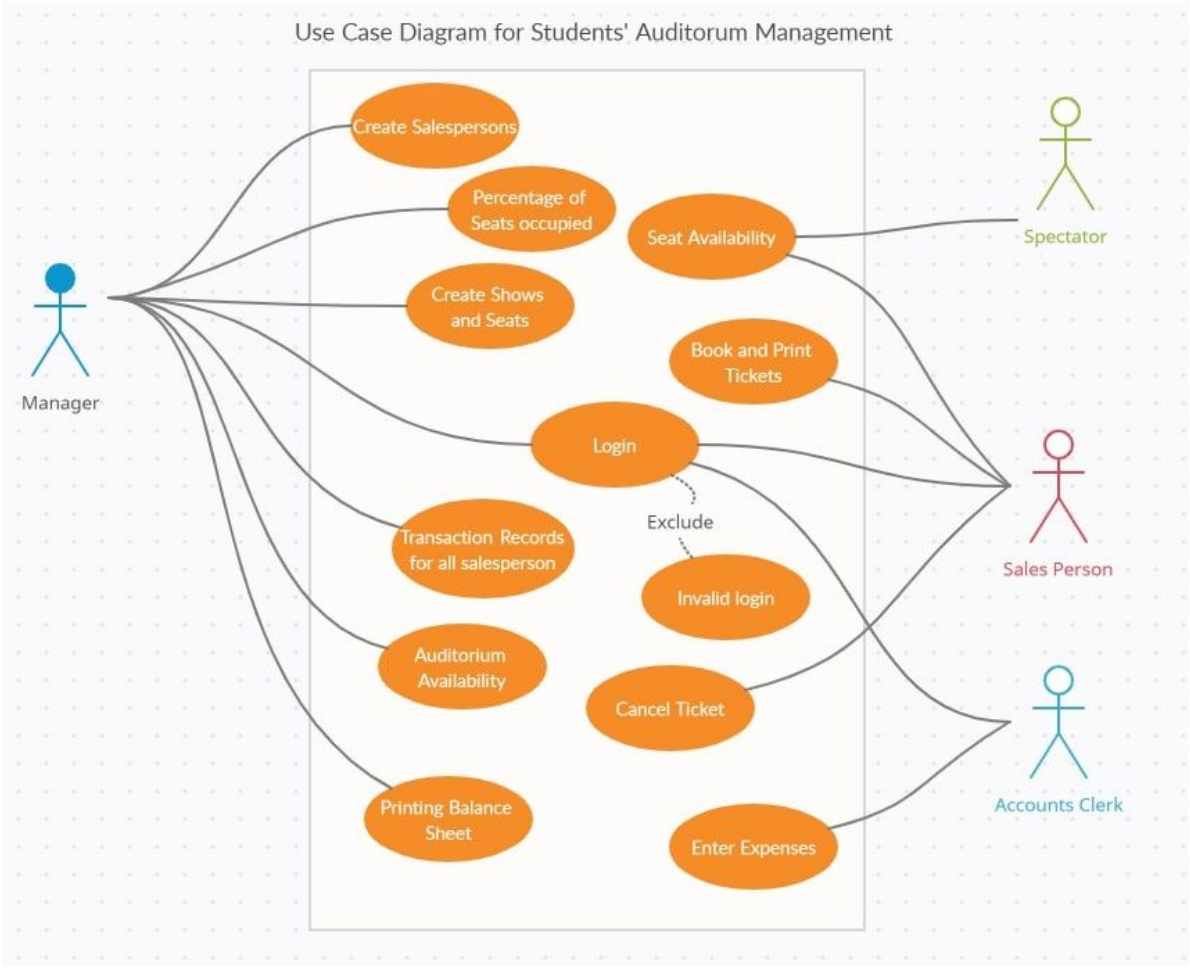
2.2 Product Functions

There are 4 types of users that can access the system. The users and the functionality that is appropriate for them are:

- The Show Manager can create new shows, set their dates and timings, set the price of tickets, control the number of seats of each type available that day, query the percentage of seats booked and the amount collected and can also view the various transactions made by the Salesperson (and also the commission granted) and expenditures incurred as updated by the Accounts Clerk.
- The Spectator can query the availability of several types of seats.
- The Salesperson can login to the system, enter the details of sales transactions made which can be accessed by the Show Manager, select the type and number of seats booked and cancel them as per the requirement of the Spectator.
- The Accounts Clerk can enter the details of various expenditures incurred for the show.

2.3 User Classes and Characteristics

All the user classes and what functions they can do are summarized in section 2.2, and also shown in the use case diagram below.



2.4 Operating Environment

It is implemented as a web app but on a local server for initial stages of its deployment.

- The system would run on *nix systems and Windows devices as well having Python3 installed.
- Dependencies include Pandas (for data management and processing) and Flask libraries (for backend).
- HTML and CSS are used for frontend. SASS may be used.
- Database is based on SQLAlchemy – a Python SQL toolkit.

2.5 Design and Implementation Constraints

In case of dealing with a large amount of data, there may be system or hardware constraints since it is based on a local server (and database) with finite resources. However, a web server may be used by future developers to alleviate this constraint. Practically no design constraints are present.

2.6 User Documentation

A README documentation would also be provided consisting of instructions for deploying the app and the usage mechanism for different classes of users

2.7 Assumptions and Dependencies

- The user should know the proper handling and usage mechanism of the software.
- The system should have Python3 installed along with Pandas and Flask libraries.
- The database files created while running the software should not be edited or deleted before the next run as all data is stored on this file.
- Any operating system capable of running MacOS, Windows or Ubuntu would be capable of running our software.
- Assumption: The commission rate would be decided by the show manager and the commission would vary linearly with the quantity of sales made.

3. External Interface Requirements

3.1 User Interfaces

The entire system is web-based, and a web browser would be running a static website which would store all information locally on the user's system. 2FA is also implemented for login for Show Managers, Salespeople and Account Clerks.

3.2 Hardware Interfaces

Software will run locally on personal computer, it just requires the device it is being run on, to have standard input devices like keyboard, mouse (or trackpad). A printer may be required to print the tickets.

3.3 Software Interfaces

The following are the libraries/software needed to run our SAMS system.

- The system runs on a browser and any modern browser (Chromium-based, Web Kit-based, and so on) would work
- Python3 must be installed on the system (Version 3.9)
- Pandas library of Python
- Flask library of Python
- SQLAlchemy library of Python

3.4 Communications Interfaces

The system would be run on a local machine, however the login portal would support SMS based user verification. Hence an OTP would be sent at the time of login, and also during booking of seat.

4. System Features

4.1 Salesperson :: Ticket Booking and Printing/Emailing Digital Ticket

4.1.1 Description and Priority

The most important function of the system is the Ticket booking. This is done by a Salesperson for spectators after getting all the required information like details of show, no. of seats, positions of seats, type of seats (normal or balcony).

4.1.2 Stimulus/Response Sequences

- Salesperson navigates to the section containing Booking and Printing info.
- System prompts the user to enter show name.
- The Salesperson enters the show name.
- System displays all the matching show names and gives the user an option to select one from this list.
- The Salesperson selects a particular screening of a show, asks the spectator their name and email ID and enters the seat numbers to be booked.
- System generates a transaction ID (or ticket ID) and prints the ticket if needed, while automatically sending an email to the spectator's email ID.
- System asks for booking confirmation.
- Salesperson confirms if payment has been received and rejects otherwise.

4.1.3 Functional Requirements

- Validity of show name: The show name must be valid, or else the system would return an empty list to choose from.
- Availability of seat: The seat number entered must be available, or else the system prints an error message that the seat is already booked.
- Validity of email ID of spectator: The email ID must be valid as the system would not be able to verify if the email ID sent to the spectator has actually been sent or not.

4.2 Show Manager :: Create Salesperson

4.2.1 Description and Priority

Show Manager can create an account for a salesperson with a unique login ID and password. It is a low priority feature because the event of creating a unique salesperson ID is not very frequent.

4.2.2 Stimulus/Response Sequences

- The show manager logs in and selects the Create Salesperson function.
- The system asks the show manager to input User ID.
- The show manager enters the User ID.

- If the user ID entered by manager does not clashes with the existing user ID, then the system prompts the manager for password.
- The manager enters the password.
- The manager submits credentials and it gets saved in the database.

4.2.3 Functional Requirements

- Unique User ID: The User ID entered by manager for creating salesperson account must be unique. If the manager enters the preexisting user ID, then the system returns to the manager menu and manager has to again select the feature to create new User ID.

4.3 Show Manager :: Create Shows and Seats

4.3.1 Description and Priority

The show manager can create a show which is to be screened on a future date (or multiple future dates as well), along with the type and number of seats. The system automatically checks if this scheduling is possible and doesn't clash with other screenings. This is a high priority feature as new shows would be added frequently and is essential to the process of ticket booking.

4.3.2 Stimulus/Response Sequences

- Show manager logs in and navigates to the menu option for this feature.
- The system asks the show manager to create a new show and select dates on which it is to be screened. If any date clashes, then the system throws an error for that day.
- Once the days are selected, the system asks the show manager to create the types and number of seats along with the prices of all the seats.
- The show manager enters these details sequentially.
- The system stores these in the database and returns to the show manager menu.

4.3.3 Functional Requirements

- Date/Time validity: The date and time of the event should be in the format specified.
- Auditorium availability: The auditorium must be free for the scheduling of the given event.

4.4 Show Manager :: Display Seat Occupancy details

4.4.1 Description and Priority

The show manager can find out the seat occupancy details for any type of seat for a given show. This is a low priority feature as it is not essential to the daily functioning of the auditorium.

4.4.2 Stimulus/Response Sequences

- The show manager logs in navigates to the menu option for this feature.

- The system prompts the show manager for the show name.
- The system displays any matching names and provides the option to select a particular screening of the show.
- After entering the show name, the show manager is shown the percentage of seats booked for each type of seat and the amount collected through sales of those seats.

4.4.3 Functional Requirements

- Show name validity: The show name entered must be valid other.

4.5 Show Manager :: Display Transaction Records for all Salespersons

4.5.1 Description and Priority

This function allows the show manager to display the transaction records for all salespeople. It displays the data salesperson-wise. This is a medium priority feature as transaction verification would not be carried out very frequently.

4.5.2 Stimulus/Response Sequences

- Show manager logs in and navigates to the menu item for this feature.
- The system displays the details for a particular salesperson (differentiated by ID) and provides functionality to navigate to other salesperson IDs.
- A search box is provided if needed to navigate to a salesperson with a given ID.
- For each salesperson, the system provides the data with a predefined number of rows. Forward and backward navigation buttons are provided to traverse the sales.
- A search box is provided to check sales for a particular day.
- It also prints the commission that the salesperson is due to receive.

4.5.3 Functional Requirements

- Valid ID for search: The ID entered to the system for viewing sales by a particular salesperson must be valid otherwise the system throws an error.

4.6 Show Manager :: Print the Balance Sheet

4.6.1 Description and Priority

This feature can only be used by the manager. This function prints the auditorium's balance sheet. This includes all the tickets booked and cancelled and expenditure entered by audit clerk. This feature has medium priority as it doesn't directly interfere in auditorium's functioning.

4.6.2 Stimulus/Response Sequences

- The manager logs in and selects this feature from the manager menu.
- The system prompts for the start date and end date.
- The manager enters the start and end date.
- The system prints all the transaction details for the auditorium in the designated time period.

4.6.3 Functional Requirements

- Valid Date Period: The date period musted be entered in the specified period. The start date entered must be before the ending date.

4.7 Seat Availability

4.7.1 Description and Priority

Everyone can view the seats available for the desired show, no login is required for this feature. It is a high priority feature as every spectator will be using this feature for viewing the available seats and choosing the appropriate ones for themselves.

4.7.2 Stimulus/Response Sequences

- The person clicks on the Seat Availability function.
- Then a person enters the name of the desired show.
- Then the person selects the show from search result for the desired time.
- The seating arrangement is shown by the system and the person can select any seat from the available ones.

4.7.3 Functional Requirements

- Valid Show Name: The show name entered by the person must be valid, else the system returns empty search result.

4.8 Salesperson :: Cancel Ticket

4.8.1 Description and Priority

This feature enables the salesperson to cancel tickets desired by the spectator. This feature is essential for spectators. This feature is of high priority as there can be cases of ticket cancellation very frequently.

4.8.2 Stimulus/Response Sequences

- The sales person logs in and selects this function from the menu.
- The sales person enters the name of the show for which the ticket is to be cancelled.
- System shows the results and sales person select the desired show.
- The salesperson selects the seats to be cancelled.
- The system finds the details of the spectator and prints the confirmation cancellation message and shows the amount to be refunded.
- The salesperson then issues the refund to the spectator.

4.8.3 Functional Requirements

- Validity of show name: The show name must be valid, or else the system would return an empty list to choose from.
- Booked seats: The seat number entered must be booked, or else the system would give an error message of seat not being booked.

4.9 Accounts Clerk :: Add Expenses to Ledger

4.9.1 Description and Priority

Enables accounts clerk to add expenses to auditorium ledger. Medium priority as failure of adding expenses would not cause the system of booking to fail.

4.9.2 Stimulus/Response Sequences

- The accounts clerk logs in and navigates to the menu option for this feature.
- The system prompts the accounts clerk to enter the name of the show. All matching shows are displayed by the system.
- The accounts clerk then adds the name and value of the expense incurred for that show.
- The system creates a transaction ID for this expense, and adds it to the ledger.
- A success message is printed.

4.9.3 Functional Requirements

- Valid expense value: The accounts clerk should enter the correct value for expense.
- Valid Show Name: The show name entered by the person must be valid, else the system returns empty search result.

4.10 Login

4.10.1 Description and Priority

Enables the different types of entities (except spectators) to login and perform desired functions in the system. This is a high priority function as the system is unusable without logging in.

4.10.2 Stimulus/Response Sequences

- System asks for user ID and password.
- The user enters the ID and password in the order.
- System checks if the details are valid and then shows the corresponding entity menu.
- If invalid, the system shows the error of Invalid Credentials and prompts the user to enter the credentials again.

4.10.3 Functional Requirements

- Valid Login Details: The details entered by user must be valid.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The performance of the system may vary based on the feature used. For example, generation of balance sheet may take more time than, let us say, entering data into the system or checking seat availability.
- Also, if it is used as an actual web application with remote server rather than local server, then internet connection might also affect the user experience.

5.2 Safety Requirements

Since the software is based on an auditorium seat management system which works as a record maintenance system, there are no major safety issues or threats with it working or failing**.**

5.3 Security Requirements

- Unique User Login Profile: Each user should have only one account. No fake accounts should be present.

- No other types of illegal login (generally incorrect or malicious) should be allowed.
- Feature of 'Forgot Password' is not implemented. So, the user must take care regarding this.

5.4 Software Quality Attributes

- RELIABILITY: The system should work reliably and not fail so often. The creation, viewing and modification of data should work properly over multiple runs of the system.
- SECURITY: Every user must have unique and single user login ID. Incorrect logins should be disallowed.
- PORTABILITY: It should be accessible through different platforms.

5.5 Business Rules

The software is primarily owned by the show manager who controls most of the aspects of show management and has almost complete autonomy over its working.

6. Other Requirements

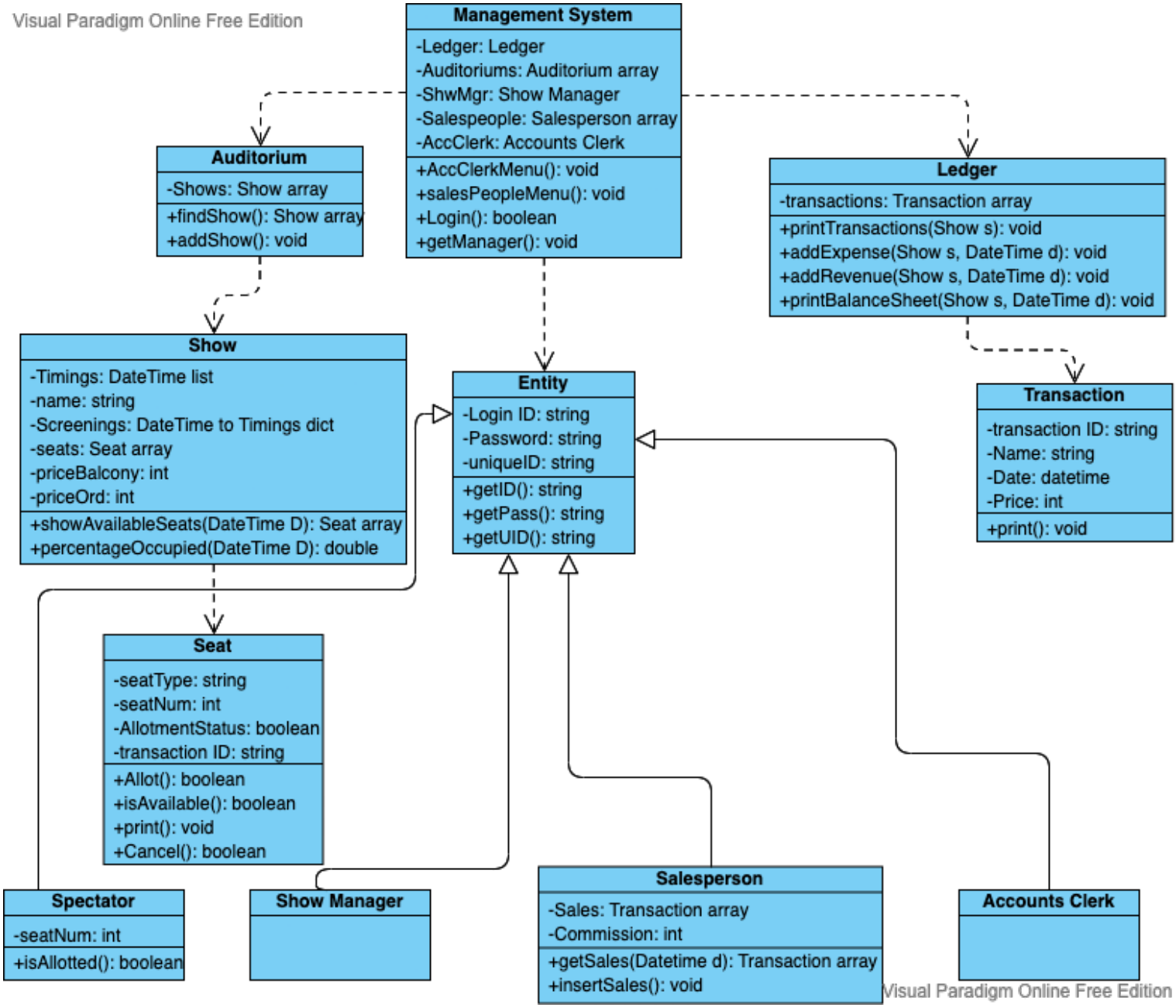
Appendix A: Glossary

Almost all terms used in this document are predefined. The same applies to the various features implemented in this software, whose functionalities have been almost properly explained. Some abbreviations are used like:

- DB or Database
- SQL or Structured Query Language
- HTML or Hyper Text Markup Language
- CSS or Cascading Style Sheets
- SASS or Syntactically Awesome Style Sheets
- SAMS or Students' Auditorium Management System
- 2FA or 2 Factor Authentication

Appendix B: Analysis Models

Class Diagram


Visual Paradigm Online Free Edition
Management System
-Ledger: Ledger
-Auditoriums: Auditorium array
-ShwMgr: Show Manager
-Salespeople: Salesperson array
-AccClerk: Accounts Clerk
+AccClerkMenu(): void
+salesPeopleMenu(): void
+Login(): boolean
+getManager(): void
Auditorium
-Shows: Show array
+findShow(): Show array
+addShow(): void
Ledger
-transactions: Transaction array
+printTransactions(Show s): void
+addExpense(Show s, DateTime d): void
+addRevenue(Show s, DateTime d): void
+printBalanceSheet(Show s, DateTime d): void
Show
-Timings: DateTime list
-name: string
-Screenings: DateTime to Timings dict
-seats: Seat array
-priceBalcony: int
-priceOrd: int
+showAvailableSeats(DateTime D): Seat array
+percentageOccupied(DateTime D): double
Entity
-Login ID: string
-Password: string
-uniqueID: string
+getID(): string
+getPass(): string
+getUID(): string
Transaction
-transaction ID: string
-Name: string
-Date: datetime
-Price: int
+print(): void
Seat
-seatType: string
-seatNum: int
-AllotmentStatus: boolean
-transaction ID: string
+Allot(): boolean
+isAvailable(): boolean
+print(): void
+Cancel(): boolean
Spectator
-seatNum: int
+isAllotted(): boolean
Show Manager
Salesperson
-Sales: Transaction array
-Commission: int
+getSales(Datetime d): Transaction array
+insertSales(): void
Accounts Clerk
Visual Paradigm Online Free Edition