Project Documentation

Online Movie Ticket Booking System

Version 1.0
12 April 2021

EDAC September 2020 Batch

Submitted by

Shalini G (200950181092)
Harish Tiwari (200950181035)
Vishakha Pahune (200950181113)
Rishabh Rao (200950181082)

1.0 Title of the project

Online Movie Ticket Booking System.

1.1 Purpose

The purpose of this project is as follow:

1. To maintain data of the movies in different theatres along with the showtime.
2. To assist an interested customer on the availability, time of Show, price of the ticket and to book their favorite movie.

2.0 Working of the Project

Design and development of web based movie ticket booking system for both desktop and mobile platforms.

- Login Page:
  - This will have three types of login facility namely Admin, Vendor and Users.
  - The login part consists of user id and password.
- Website Pages:
  - Cities, all cities are shown.
  - Theater, theaters in a particular city are shown.
  - Movies, movies available in particular theater are shown.
  - Shows, shows available for particular movie are shown.
  - Seats, seats available for particular show are available.
  - Profile, customer profile (with ticket's related to the customer) and vendor profile (with theater's related to the vendor)
- Forms:
  - Registration Form: Customer registration and Vendor registration.
  - Vendor Add: Admin can add vendor.
  - City Add/Update: Admin can add/update city.
  - Theater Add/Update: Vendor can add/update theater.
  - Movie Add/Update: Vendor can add/update movie.
  - Show Add/Update: Vendor can add/update show.
- Customer can only book the show and cancel the show.

➢ ADMIN PERSPECTIVE

- Admin is already been given the user id and password, so registration is not allowed.
- Admin user can login and he can see all the cities and followed by the vendor list, user list.
- The Admin can add the vendor and also block/delete the vendor.
- The Admin can block a user.

➢ VENDOR PERSPECTIVE

- Vendor is the movie owner who has the privilege to add the theaters and also the shows running in the theater.
- He can updated the address details, delete the movie etc.
- He can also update his own profile.

➢ USER PERSPECTIVE

- As a user, he can register into the website using the details such as name, username password.
- After registration user can login to the system when he can view the movies, book tickets, cancel tickets and update his profile.
- User first can select a city, once selected he/she will view the list of theater available in that particular city.
- Once user clicks on the preferred theater, the user will be able to see the different movies showing in that theater, once the movie is selected use can see the showtimes of the movie.
- Then when the user clicks on the particular showtime he will be able to view the seating where he can choose the ‘n’ seats which he wants and click on confirm which will book the ticket and the ticket is confirmed.
- Similarly user can also cancel the tickets which are booked for n upcoming show if he does not want to watch the movie anymore.
- User also has an option to update his profile, under the update profile option where he can update his name, email id etc but not the username.
- User can also view the list of the tickets which he has booked.

3.0 Workflow

Customer:

Presentation Layer: (React JS)

a) User can login using username and password.
b) User search city and clicks on the city name
c) Where it redirects to list of theatres present in city
d) User clicks on theatre name and redirects to list of movie page
e) User can click on the movie from where it redirects to list of show timings.
f) User can select on show time, it redirects to seat booking place and user can book seats and proceed to payments.

Service Layer: (Spring Boot)

a) Login is authenticated by spring security
b) It calls username from function getCustomerByUserName(String name).
c) It lists city using List<City>getAllCities().
d) It searches city by using getCityById(Integer cityId).
e) It gets all theatres by List<Theater>getAllTheaters().
f) It searches theatre by getTheaterById(Integer theaterId)
g) It lists all movies by List<Movie>getAllMovies().
h) It searches movies by getMovieById(Integer movieId)
i) It get show timings from getShowById(Integer showId)
j) It books seat using bookSeats(Integer showId, Integer customerId, List<String> seats).

DAO Layer: (Spring Hibernate-JPA)

a) It gets username from database usingfindById(username).get().
b) It gets user details from findCustomerByApplicationUser(applicationUser);
c) It searches city by using findById(cityId).get().
d) It searches theatres by using findById(theaterId).get()
e) It searches movies by using findById(movieId).get()
f) It searches shows by using findById(showId).get()
g) It gets seats by findById(seatingId).get().

4.0 Problems Faced and how we Overcame it.

1) Authorization:
   We overcome it by using spring security. We used JWT as an authorization mechanism.

2) Committed the wrong data on github
   a. We needed to go back to the working commit and update the same on github also. So, We resolved it by command git reset –hard "commit name" which permanently resets to the previous commit(we should only use this command when it is very important), and then we use git push –force origin main, which updates the commits on github(we should only use this command when it is very important).

3) Error while using get customer by Id:
   a. We had this issue because of the relation between the Entity.
   b. Customer had one to many relation with ticket, ticket has many to one relation with show
   c. To resolve this error we added " spring.jackson.serialization.FAIL_ON_EMPTY_BEANS=false " to application.properties file.

5.0 Learning during the Project.

While doing the project each of us learnt the key functionalities of different technologies.

- We have learnt how to build a full stack application using Sprint boot + React, and how the components of the Presentation, service and the DAO layers communicate with each other to provide the required outcome.
- In sprint boot we learnt about how to divide the project in 3 sub divisions (Model View Controller –MVC) mainly Presentation, service and the DAO layers and make the model more interactive and efficient.
- We also learnt the usage of various annotations which is used in class-level, method –level as well as the variable level.
- We also learnt about how to use the featured of JPA as a repository to manage the data between the Java objects / classes and the RDBMS (MySQL).

- We also learnt about the MySQL which acts as the database storage management in our web application. We learnt how to perform the basic CRUD operations using the Spring Boot(JPA) and the RDBMS.
- As our frontend is React we first made basic POC on how to connect the backend components with the frontend components.
- Further which we learnt advanced feature on how to reuse certain components for all the operations which helps in making the code more efficient.
- The sources that were used for learning was the official documentations of the various technologies used, YouTube tutorials and also stackOverflow.