COEN 6311 – Software Engineering
Term: Winter 2022

CENTRELAIZED MOVIE TICKET BOOKING SYSTEM V3.0 SPRINT 3 REPORT

Group Members	Student ID
Abishek Arumugam Thiruselvi	40218896
Srinidhi Honakere Srinivas	40221128
Manish Pejathaya	40194909
Mohammad Abdul Wasay Ejaz	40185576
Mohammad Alikhanizanjani	40221406

Instructor: Prof. Dr. Paula Lago

Date: 24th April 2022

Table of Contents

Contents

1. INTRODUCTION

1.1. PRODUCT PERSPECTIVE:

A customer wants to create a movie ticket booking system for their company. In the first phase of development, they want their customers to view the list of movies available and allow customers to add them to their wish list. This document specifies the list of requirements needed for the initial phase of the application launch.

1.2. PROJECT LINKS:

Please find the Minutes of Meetings of the final sprint meeting:

1. GitHub URL: https://github.com/abishekat/SE-DEPLOY

2. JIRA: https://aarumugam.atlassian.net/jira/software/projects/CA/boards/3

3. Project Video
   - One Drive CU URL: PROJECT VIDEO.mov
   - Google Drive URL: https://drive.google.com/file/d/184EEXMOauvr1QTpgv3rNPhmeJ_Z09us0/view?usp=sharing

4. Production URL: http://54.167.74.161:5000/

   - There is an issue in the deployment URL, if I log off PC (the AWS EC2 instance), the production URL does not work. It requires my PC to be ON always.
   - Kindly prompt before viewing the project as I could **connect the EC2 instance again**.

1.3. PRODUCT USERS AND STAKEHOLDERS:

The Product users and stakeholders of the software application are:

- **Customer:** The one who specifies our project requirements and helps us gather information to proceed with our project.
- **Admin:** The one who runs and manages the application constantly, managing movie information and details.
- **Development Team:** Is responsible for developing and testing the software according to the customer's requirements.
- **Product owner:** Is responsible for bringing up new ideas and designs that can be implemented in the agile process.

2. SPRINT REPORT

Provide a summary of the previous sprint results to show the project status before the current spring.

2.1. PRODUCT BACKLOG

- As a user, I should be able to register in the application.
- As a user, I should be able to log in to the application.
- As a user, I should be able to view the application's homepage.
- As a user, I should be able to select the location.
- As a user, I should be able to view movie timings and the screen it is being played on.
- As a user, I want to view the details about the movie.
- As a user, I should be able to view a short trailer of the movie.
- As a user, I should be able to book tickets for the movie of my choice.
- As a user, I should be able to select the seat and lock the seats of my choice.
- As a user, I must view the ticket after the seat selection
- As a user, I should be able to contact the application team for any customer support.
- As a user, I should be able to save my personal information in the database securely.
- As an admin, I want to log in to my admin account.
- As an admin, I want to edit the movies details like posters, timings, screens, and descriptions in the database from the application.
- As an admin, I should be able to manage the screen and seats available on every screen with the user booking.

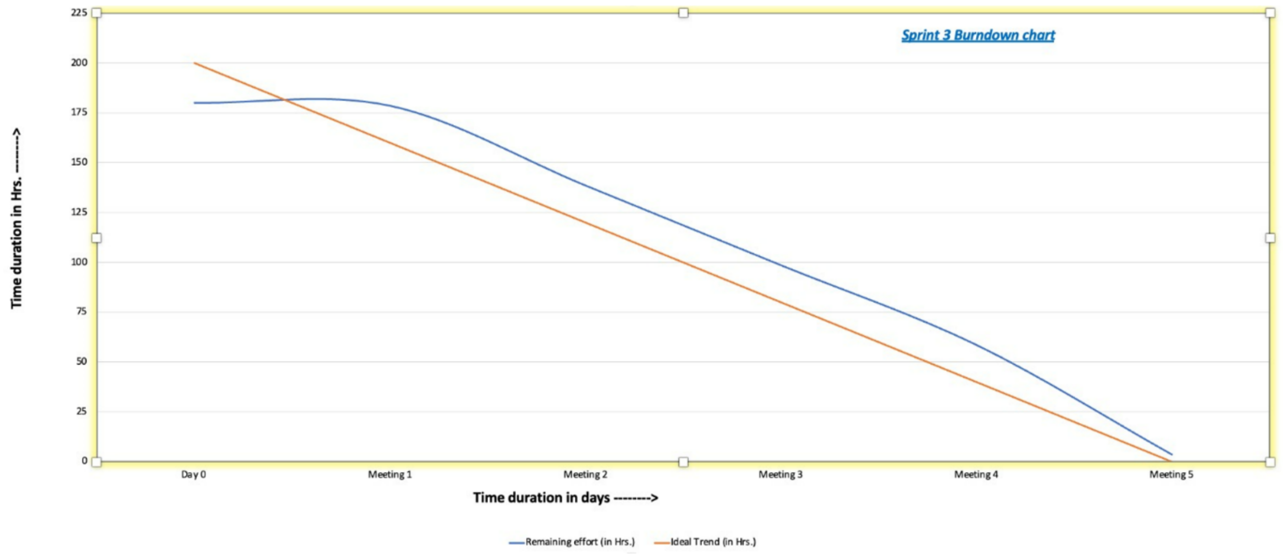
2.2. SPRINT GOAL

- Users should be able to view movies at a location.
- Users should view a short snip about the movie and its description
- Users should be able to select and lock the seats directly from the database.
- Users should be able to view the ticket booked for the movie.
- Users should be able to save their credentials using hashing at the login post.

2.3. SPRINT BACKLOG

- As a user, I should be able to view movies at a location.
- As a user, I should be able to select the movie to be booked.
- As a user, I should be able to view the short description, cast, and trailer of the movie.
- As a user, I should be able to book the ticket from the selected screen, timings, and seats.
- As a user, I should be able to view movie tickets after seat selection.

2.4. BURNDOWN CHART



2.5. SPRINT REVIEW

Goals:

- Users should be able to view movies at a location.
- Users should view a short snip about the movie and its description
- Users should be able to select and lock the seats directly from the database.
- Users should be able to view the ticket booked for the movie.
- Users should be able to save their credentials using hashing at the login post.

Accomplished:

- Users should be able to view movies at a location.
- Users should view a short snip about the movie and its description
- Users should be able to select and lock the seats directly from the database.
- Users should be able to view the ticket booked for the movie.

2.6 SPRINT RETROSPECTIVE

- We have successfully reached the tickets view page
- The web application includes registration, login, the home page, movie selection, ticket seat selection, and ticket view page.
- All the data is not from JS. They are from AWS, which can be modified by the admin.
- We learned and did time management, teamwork and KT.
- We followed the code stacking arrangements to make the code readable.
- Pair programming (like Scrum) helped. Mapping one developer to 1 tester resolved many issues.

3. REQUIREMENTS

3.1. USE CASE DIAGRAM

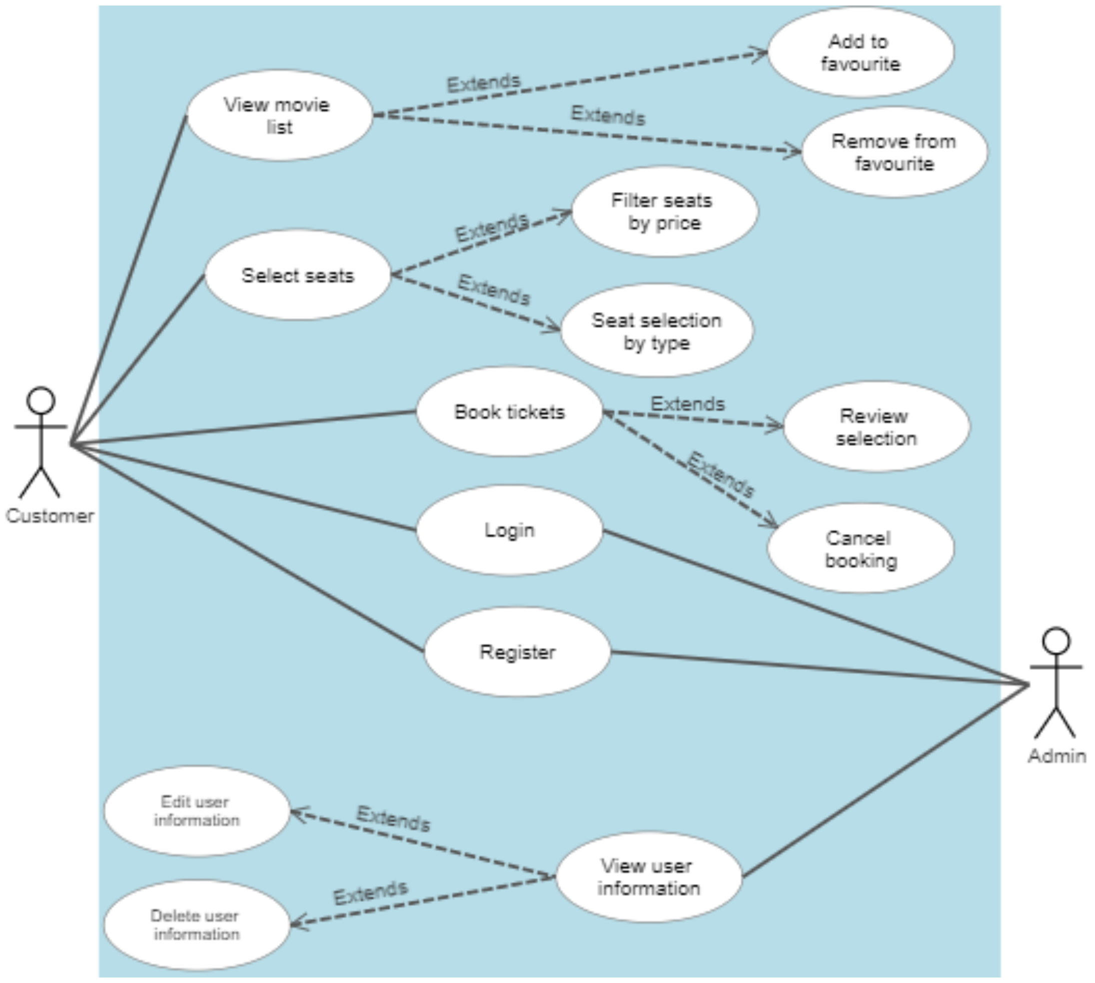


Figure 1, Use case Diagram

1. **View movie list**: The list of movies being played near the user's location will be shown on the home page, and the user can click on any movie to get the details of the movie.
2. **Select seats:** The user will be taken to a seat selection page once he clicks the book ticket button. The seat selection page shows the user all the seats available at the theatre, and the user can block any available seats they want to sit in.
3. **Book tickets:** Once the user confirms the seat, they can block their seats by clicking the book ticket button. Once clicked, the user is redirected to a ticket view page where they can view their tickets.
4. **View user information:** The admin can view the user's information and make changes or delete them accordingly.

3.2 NON-FUNCTIONAL REQUIREMENTS

S.NO	Requirements	Priority
1.	Hashing to be done to the user login information to protect user data	LOW
2.	The software should be available 24 hours a day and 7 days a week.	HIGH
3.	Load balancer must be incorporated in order to manage the incoming traffic.	LOW
4.	The system will have to store the booking details of each user.	MEDIUM

4. SOFTWARE ARCHITECTURE

4.1. SYSTEM CONTEXT

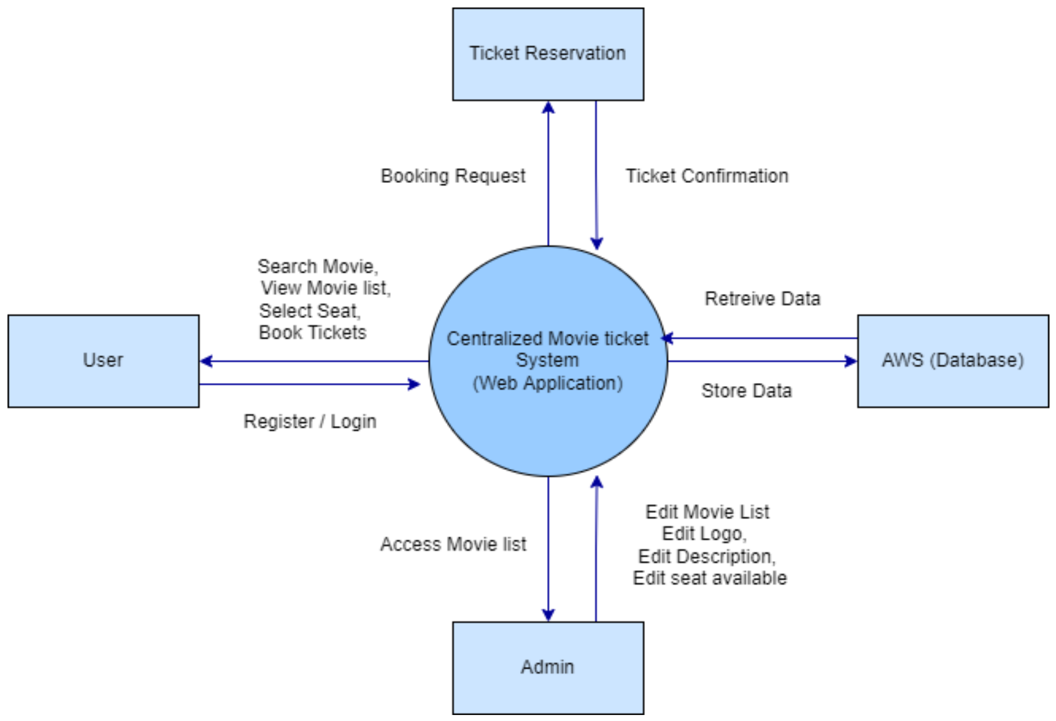


Figure 2, Context Diagram

4.2. ARCHITECTURAL STYLE:

The layered architecture is used in this project. The layered architecture is suitable for most applications when it's unclear which architecture style will be employed. Developers can start with this architecture style and change it easily to other architectural styles such as microservices. This will make the transition to a different architectural style easier. [1].

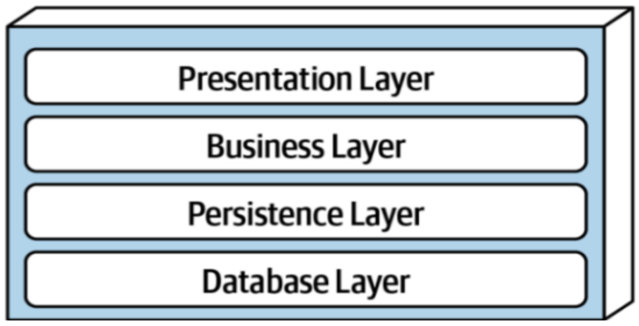


Figure 3, Architecture Layer

Why Use This Architecture Style?

The layered architecture style is excellent for small, simple applications or websites. It's also an intelligent architecture choice for circumstances with restricted budgets and timelines, especially as a starting point. Layered architecture is likely one of the lowest-cost architecture types, promoting ease of development for smaller applications due to its simplicity and familiarity among developers and architects [1]. The concept of layers of isolation means changes that happen in one layer will not affect other layers. For example, we want to change the presentation layer of the website from simple JavaScript to powerful AngularJS, so our website behaves responsively. Furthermore, it is easy for developers to understand layer architecture, so it is a significant point why we used this architecture.

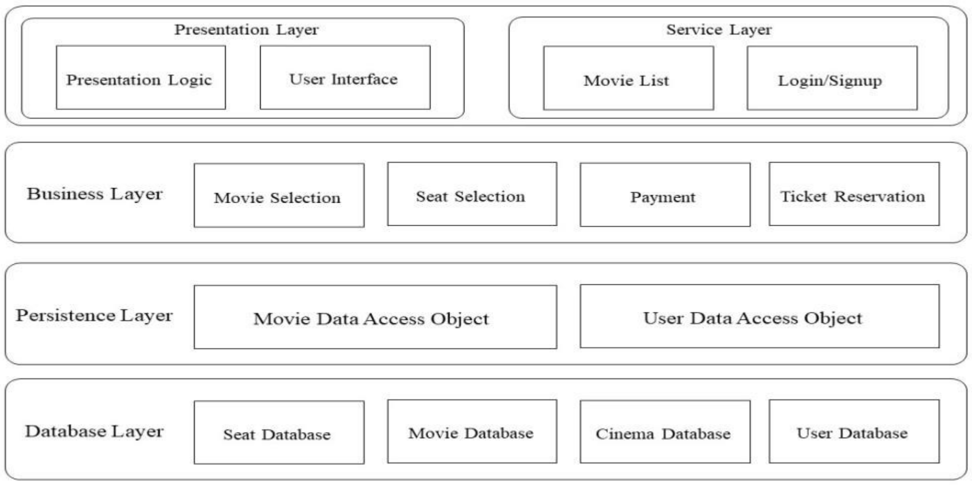


Figure 4: Architecture style

4.3 COMPONENT DIAGRAM

The system's main functional components are identified, and their responsibilities are described below.

TABLE. COMPONENT

Component	**Responsibilities**
Movie	This module represents movie list.
Authentication	This module is responsible for authenticate the users after login
Search Engine	This module receives the typed name and represents the related movies.
Seat	This module represents available seats.
Ticket Reservation	This module is responsible for reserving the ticket after registration.
Bank Transaction	This module communicates with payment processing service after reservation which is external component.
Database	This module represents movie/seat/customer database.

The component diagram is shown in the figure below.

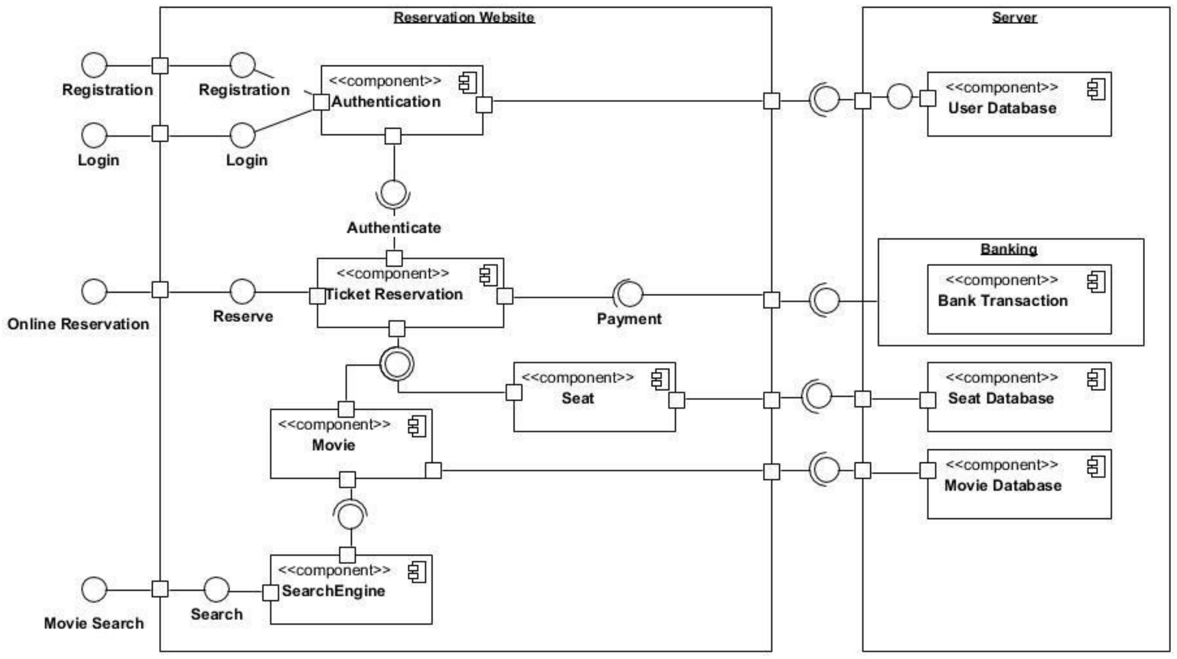

Reservation Website
Server
Registration
Registration
<<component>> Authentication
Login
Login
Authenticate
<<component>> User Database
Banking
<<component>> Bank Transaction
Online Reservation
Reserve
<<component>> Ticket Reservation
Payment
<<component>> Seat
<<component>> Seat Database
<<component>> Movie
<<component>> Movie Database
<<component>> SearchEngine
Movie Search
Search

Figure 5: Component diagram.

4.4 ACTIVITY DIAGRAM

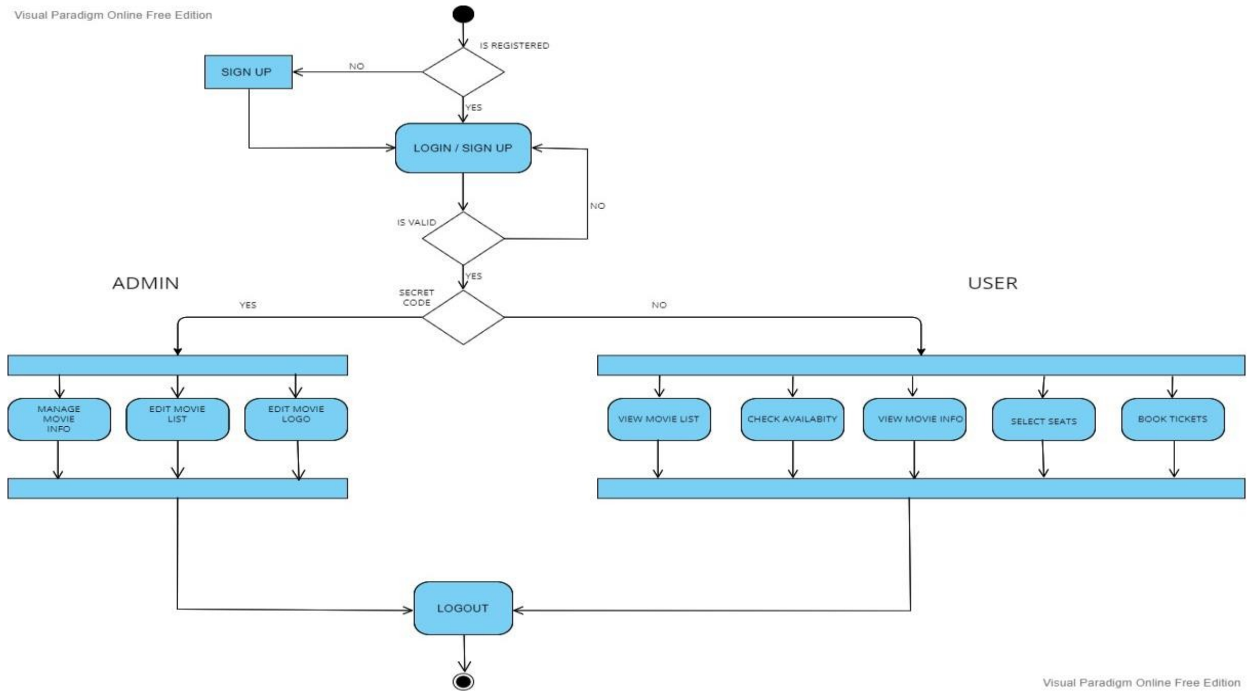


Figure 6: Activity diagram.

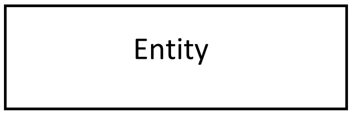
5 SOFTWARE DESIGN

It is the process of implementing software solutions to one or more problems. It usually comprises problem-solving and planning software solutions.

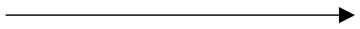
4.3 DATA DESIGN

Once all the requirements have been completed, the next phase represents how various entities are related to each other.

- Rectangles represent entities. An entity is a concept or an object in which you want to store the information.



- The data flow symbol represents the data flow occurring between two external entities.



- Cardinality represents how many instances of an entity relate to one instance of another entity.

1. One to one – 1: 1
2. One to many – 1, n or 1..*

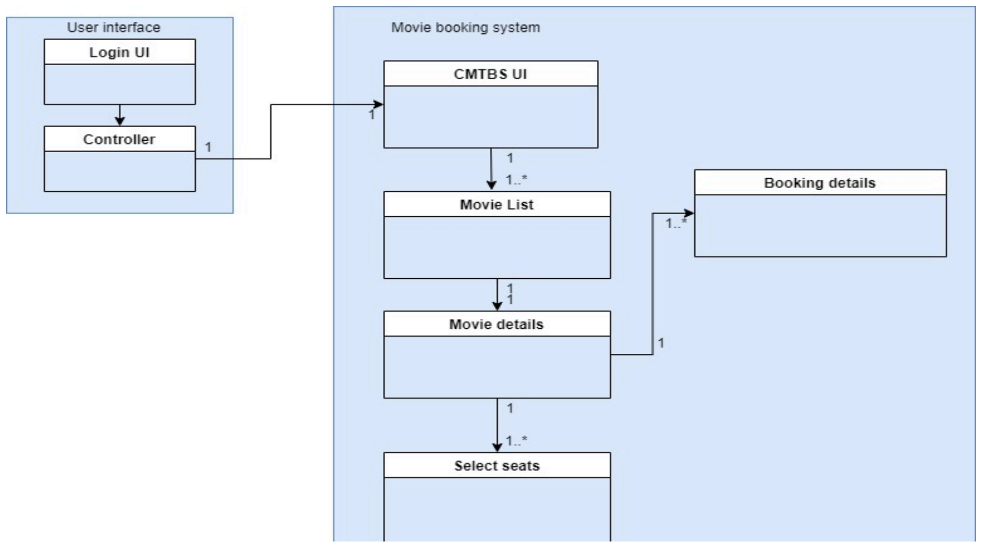


Figure 7, Data Design Diagram

5.2 COMPONENT DESIGN

Component Design comprises a Class diagram in which object-oriented systems are visualized. It is a static structure diagram that includes the system's classes, attributes, operations, and relationships between objects. For modeling object-oriented systems, class diagrams are the only UML-based diagrams that can be linked directly. The top box represents the name of the class, followed by the attributes and then the operations. Classes may be related to each other in unique ways, such as inheritance, composition, or direct association. Here we have assigned the attributes., Username, Password, Address, Contact no,etc. We use set methods to set the value of attributes and get methods which help access the attributes. The data is inserted according to the user's requirements.

Some factors, such as coupling and cohesion, are essential factors for a good design. Coupling describes the connection or link between design components. It is reflected by the number of links an object has and the number of interactions the object has with others. Cohesion describes as an element that contributes to a single purpose. The idea of coupling and cohesion are not mutually exclusive but support each other.

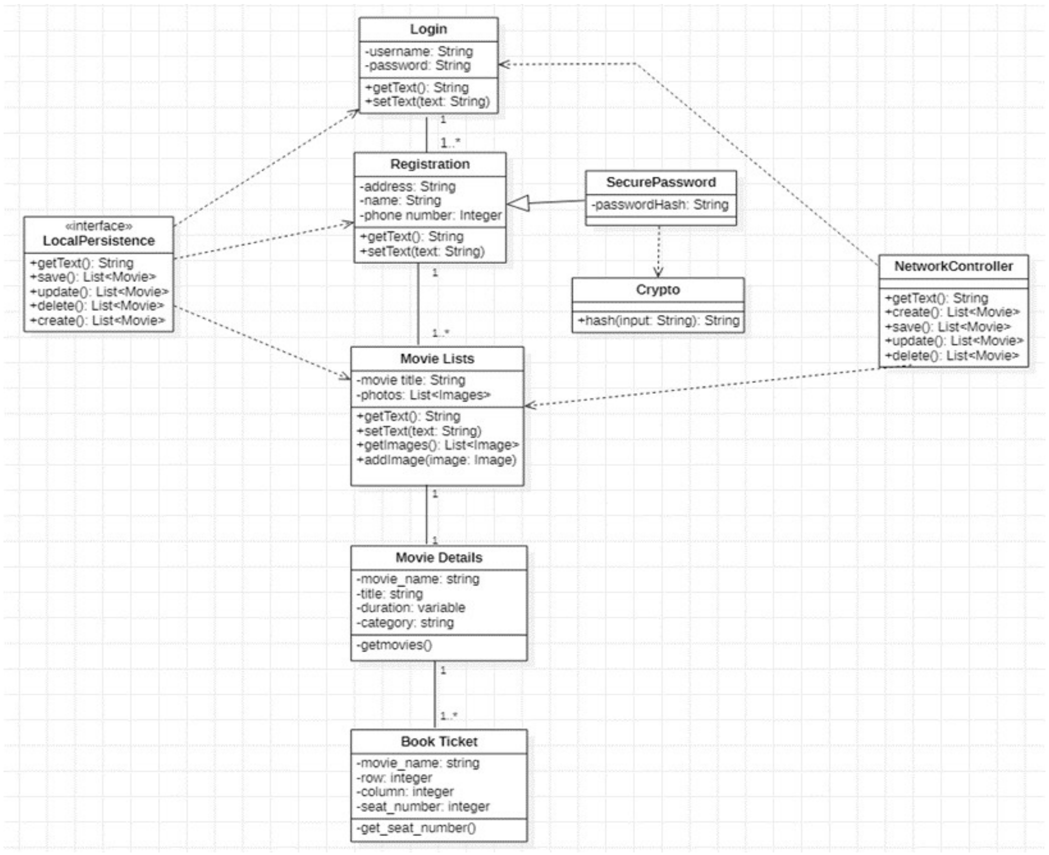


Figure 8, Component Design Diagram

5.3 SEQUENCE DIAGRAM

A sequence diagram for reserving the ticket function is shown below.

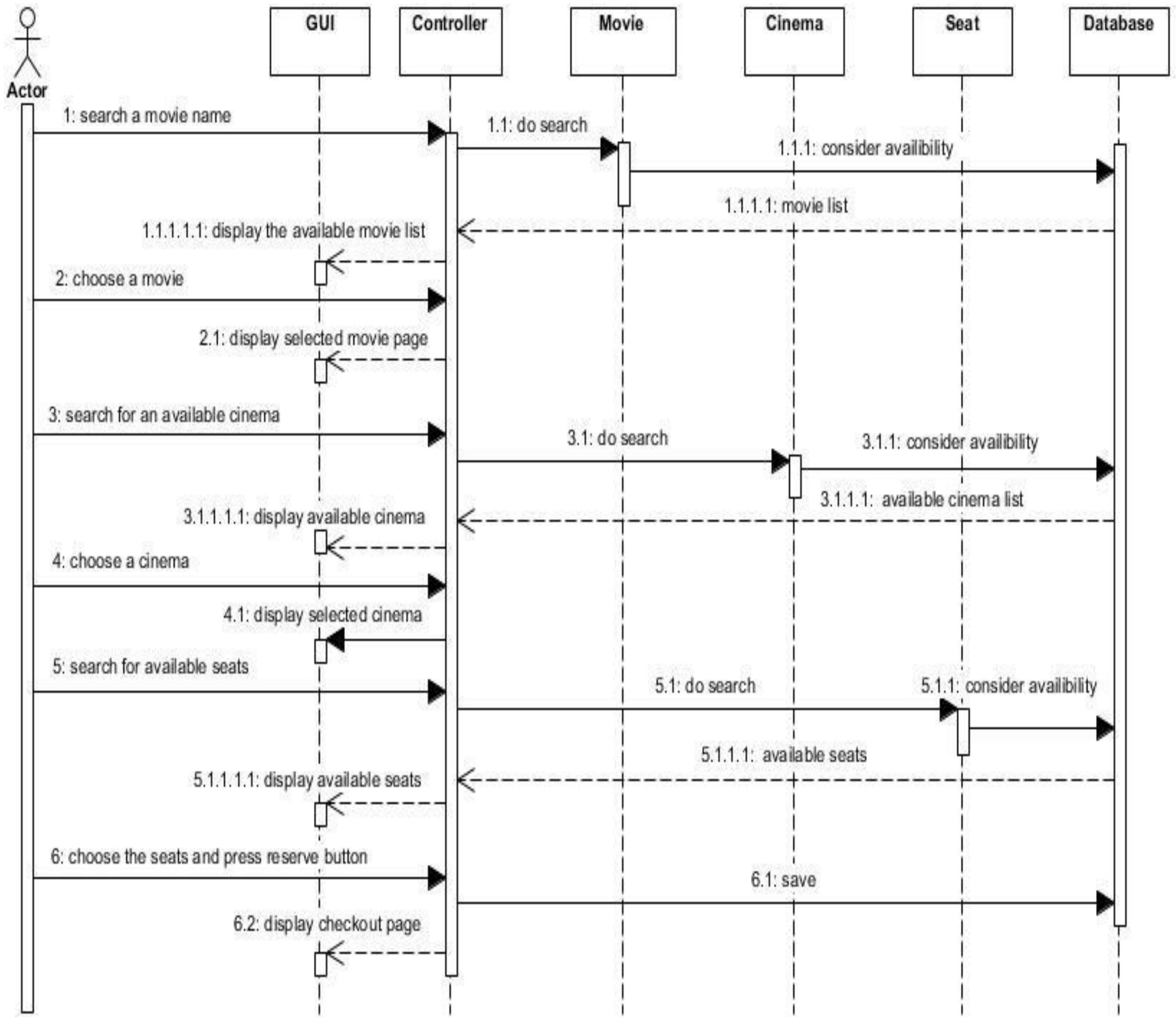

Actor
GUI
Controller
Movie
Cinema
Seat
Database
1: search a movie name
1.1: do search
1.1.1: consider availibility
1.1.1.1: movie list
1.1.1.1.1: display the available movie list
2: choose a movie
2.1: display selected movie page
3: search for an available cinema
3.1: do search
3.1.1: consider availibility
3.1.1.1: available cinema list
3.1.1.1.1: display available cinema
4: choose a cinema
4.1: display selected cinema
5: search for available seats
5.1: do search
5.1.1: consider availibility
5.1.1.1: available seats
5.1.1.1.1: display available seats
6: choose the seats and press reserve button
6.1: save
6.2: display checkout page

Figure 9, Sequence Diagram

5.4 User Interface Diagram

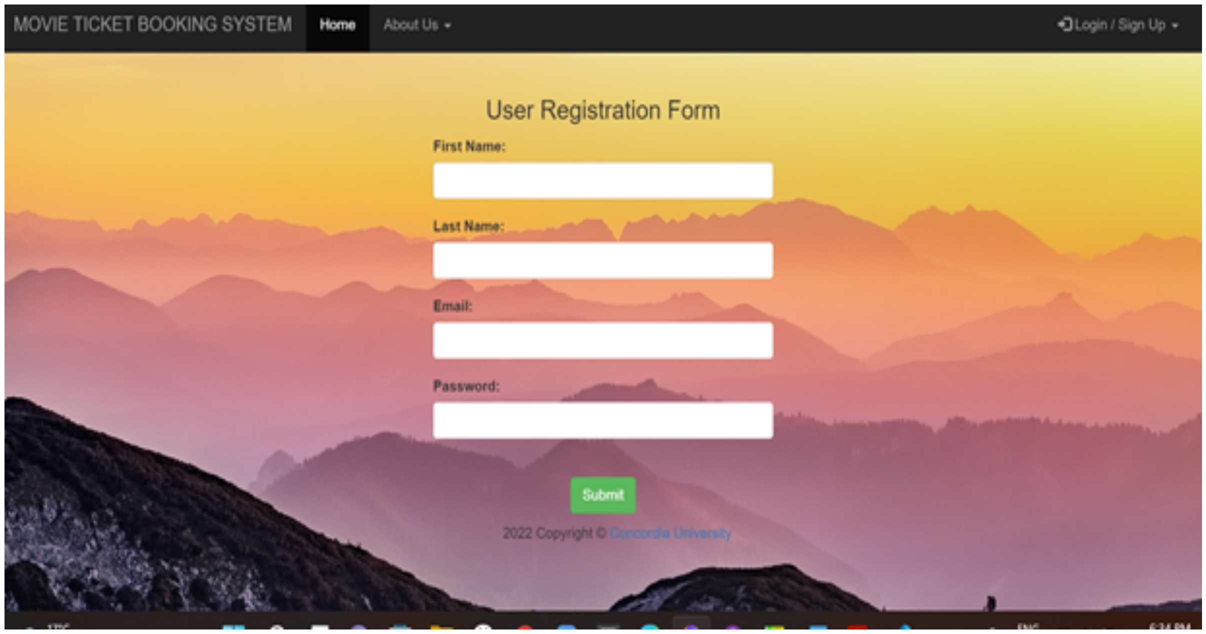


Figure 10, User Registration UI

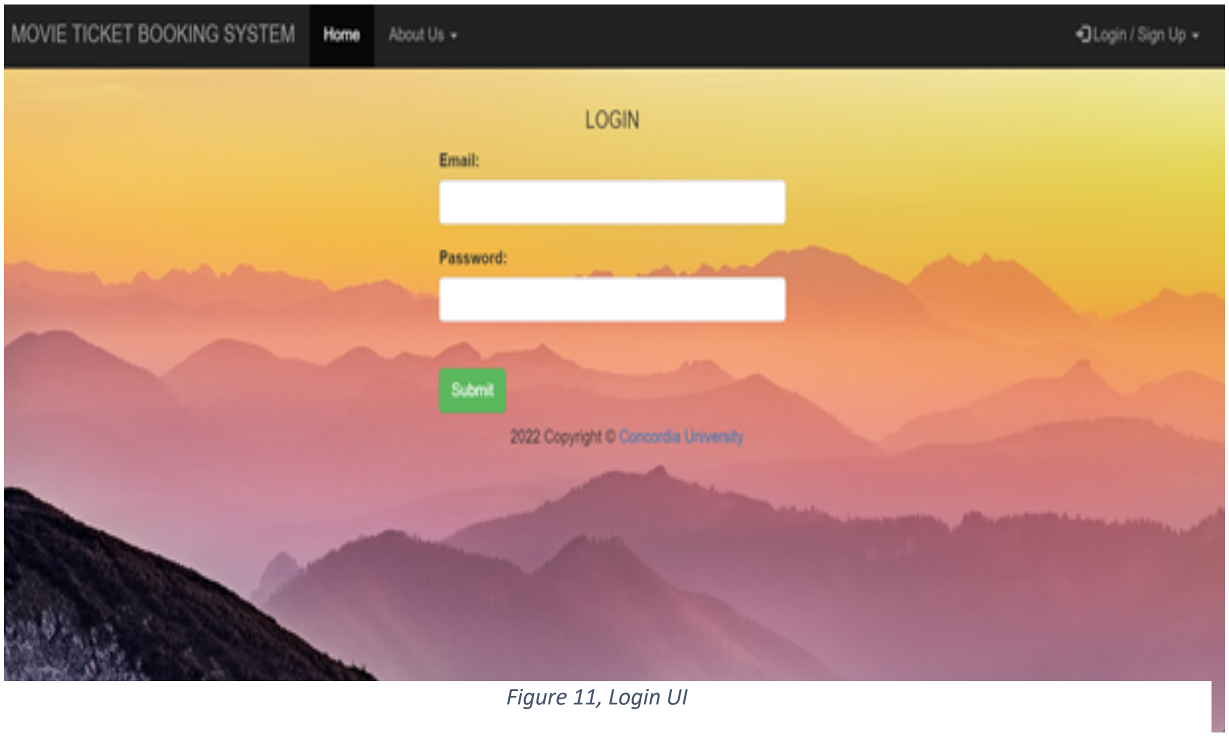
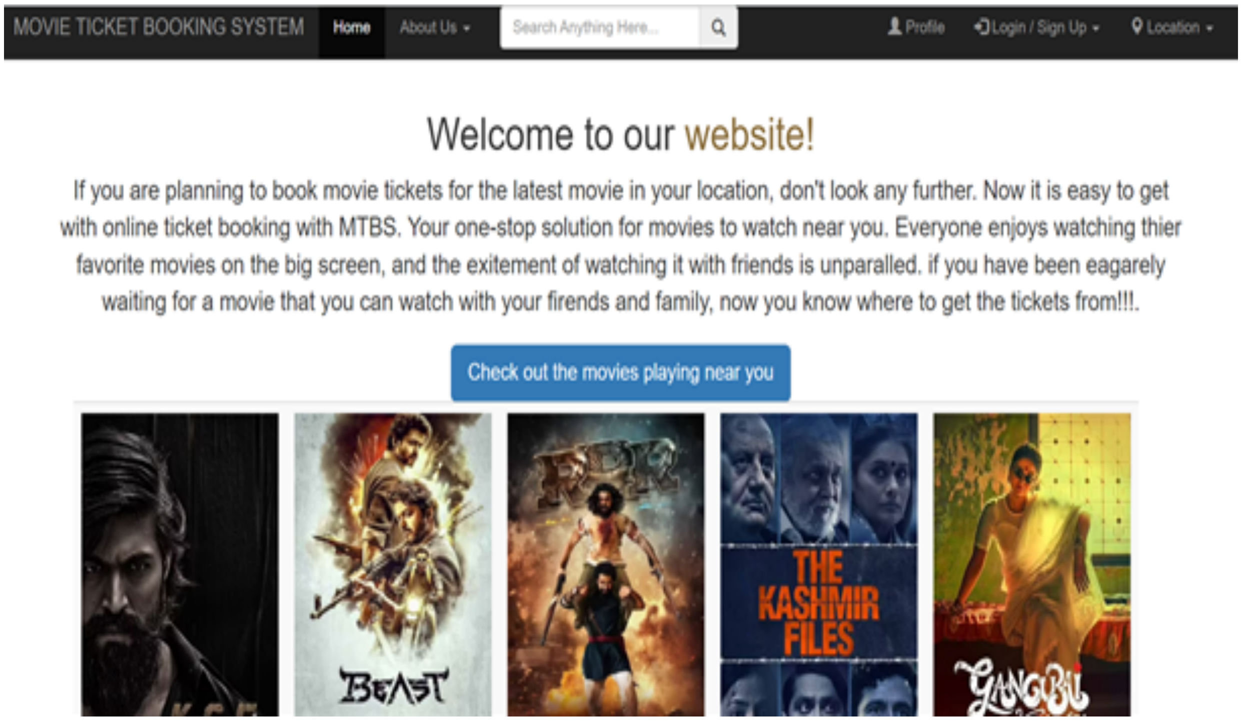

MOVIE TICKET BOOKING SYSTEM
Home
About Us
Login / Sign Up
LOGIN
Email:
Password:
Submit
2022 Copyright © Concordia University

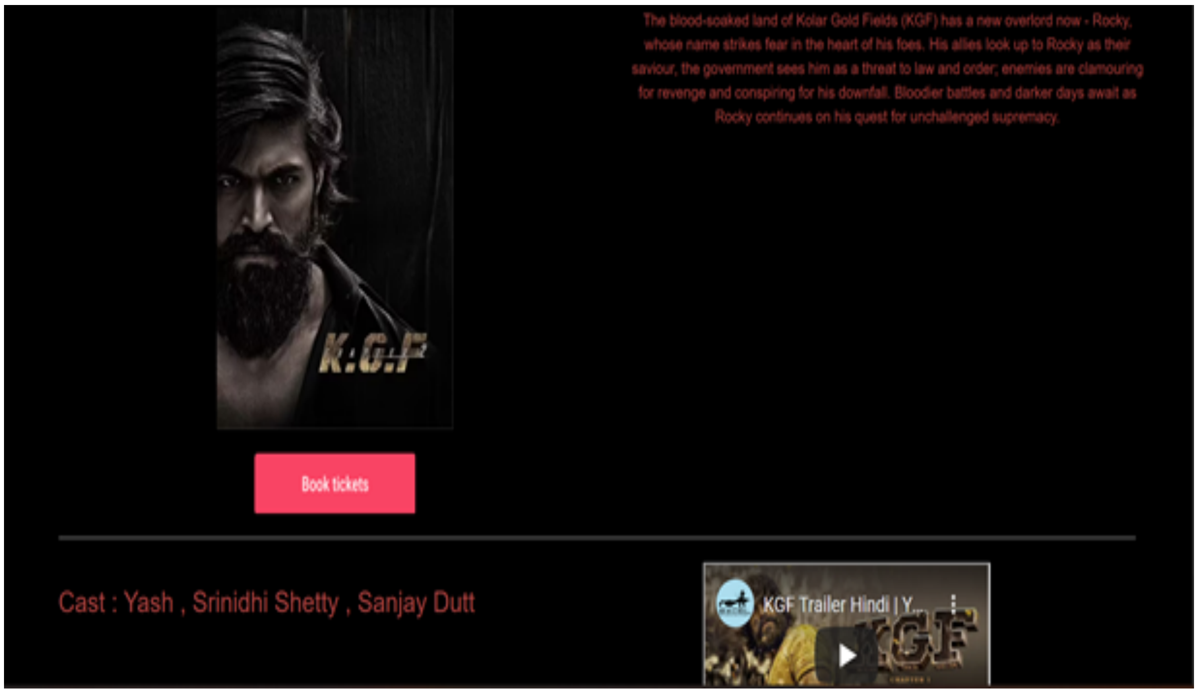
Figure 11, Login UI



Welcome to our website!

If you are planning to book movie tickets for the latest movie in your location, don't look any further. Now it is easy to get with online ticket booking with MTBS. Your one-stop solution for movies to watch near you. Everyone enjoys watching thier favorite movies on the big screen, and the exitement of watching it with friends is unparalled. if you have been eagarely waiting for a movie that you can watch with your firends and family, now you know where to get the tickets from!!!.

Figure 12, Homepage Ui



The blood-soaked land of Kolar Gold Fields (KGF) has a new overlord now - Rocky, whose name strikes fear in the heart of his foes. His allies look up to Rocky as their saviour, the government sees him as a threat to law and order; enemies are clamouring for revenge and conspiring for his downfall. Bloodier battles and darker days await as Rocky continues on his quest for unchallenged supremacy.

Cast : Yash , Srinidhi Shetty , Sanjay Dutt

Figure 13, Movie Page UI

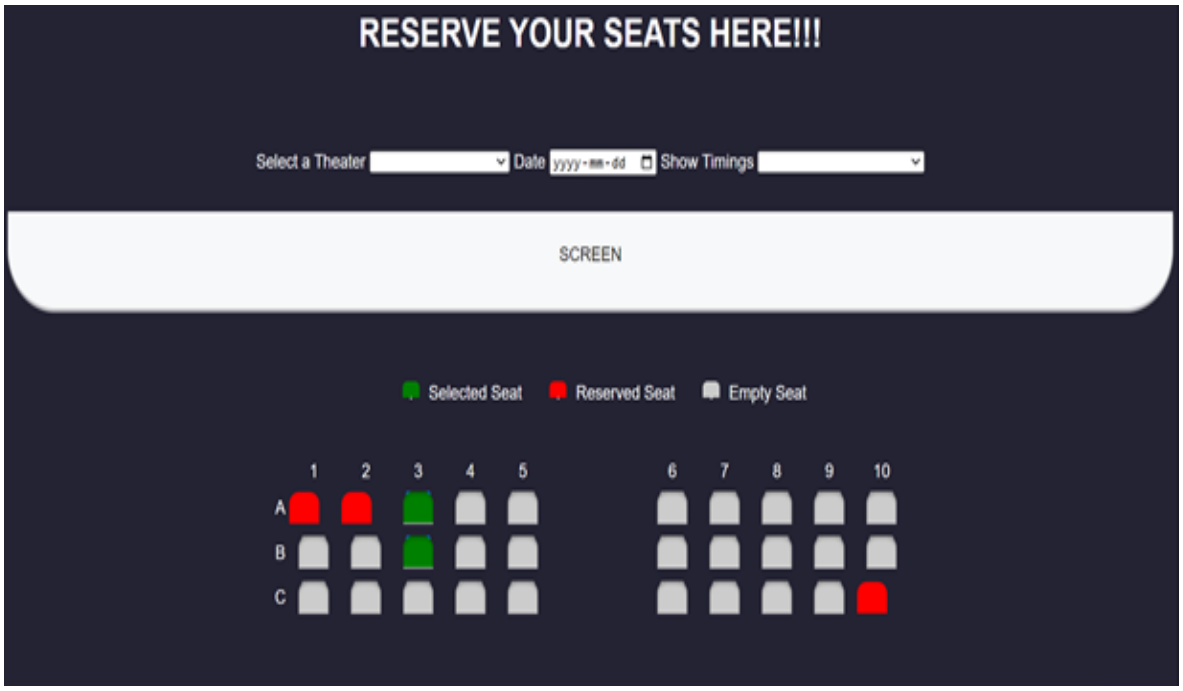


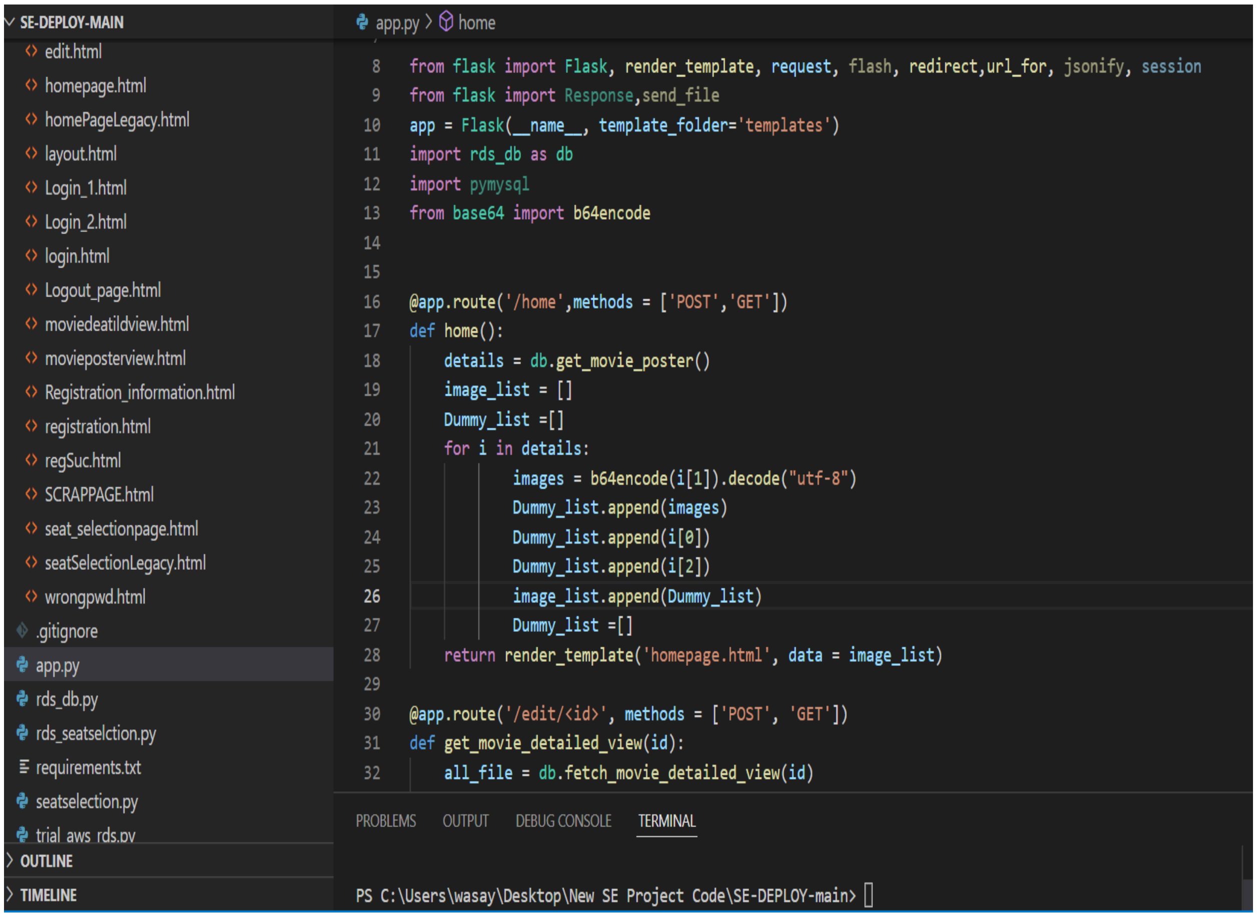
Figure 14, Seat Selection UI

The video demo for the project can be found by clicking on the following link-

https://drive.google.com/file/d/184EEXMOauvr1QTpgv3rNPhmeJ_Z09us0/view?usp=sharing

5.5 CODE STRUCTURE

The code structure of the project is shown below.



```
SE-DEPLOY-MAIN
  edit.html
  homepage.html
  homePageLegacy.html
  layout.html
  Login_1.html
  Login_2.html
  login.html
  Logout_page.html
  moviedeatildview.html
  movieposterview.html
  Registration_information.html
  registration.html
  regSuc.html
  SCRAPPAGE.html
  seat_selectionpage.html
  seatSelectionLegacy.html
  wrongpwd.html
 .gitignore
 app.py
 rds_db.py
 rds_seatselction.py
 requirements.txt
 seatselection.py
 trial_aws_rds.py
OUTLINE
TIMELINE
```

```python
from flask import Flask, render_template, request, flash, redirect,url_for, jsonify, session
from flask import Response,send_file
app = Flask(__name__, template_folder='templates')
import rds_db as db
import pymysql
from base64 import b64encode


@app.route('/home',methods = ['POST','GET'])
def home():
    details = db.get_movie_poster()
    image_list = []
    Dummy_list =[]
    for i in details:
            images = b64encode(i[1]).decode("utf-8")
            Dummy_list.append(images)
            Dummy_list.append(i[0])
            Dummy_list.append(i[2])
            image_list.append(Dummy_list)
            Dummy_list =[]
    return render_template('homepage.html', data = image_list)

@app.route('/edit/<id>', methods = ['POST', 'GET'])
def get_movie_detailed_view(id):
    all_file = db.fetch_movie_detailed_view(id)
```

```
PS C:\Users\wasay\Desktop\New SE Project Code\SE-DEPLOY-main>
```

Figure 15, Code Structure

Resources:

[1] M.Richards & N.Ford, Fundamentals of Software Architecture, An Engineering Approach, first ed., O'Reilly Media, 2020.

6. SOFTWARE TESTS

6.1. TEST PLAN

Once the development phase is completed, we must test the system to ensure that it is working as expected and that there are no apparent bugs. These are basic mistakes in the software code. Testing is carried out to ensure that the system meets the client's needs (functional requirements). The System Test is done to find defects and correct them before implementation. However, following a System Test approach will assist in reducing risks and ensuring a successful project.

Following are the test case scenario for the Centralized Movie Booking System:

- Seat selection
- View Movie ticket
- Logout

Test Case				
Test Scenario name		Seat Selection		
Description		This scenario covers the functionality of movie seat selection		
Module		Seat Management		
Status		Completed		
S/No	**Test case**	**Purpose**	**Actual Result**	**Expected Output (Pass / Fail)**
1.	Display white (empty / available) coloured seats.	The user can view the available seat	Show available seat	Pass
2.	Display red (booked) coloured seats.	The next user will know that the seat is already booked by someone else.	Unable to select the booked seat	Pass
3.	Display green colour when user select the seat.	User can view the selected seat.	Select the available seat.	Pass

Test Case				
Test Scenario name		View Movie ticket		
Description		This scenario covers the functionality of movie ticket		
Module		Book Tickets		
Status		Completed		
S/No	**Test case**	**Purpose**	**Actual Result**	**Expected Output (Pass / Fail)**
1.	Display movie name in the ticket.	The user can view the name of the movie in the ticket.	Movie name (Ticket)	Pass
2.	Display Customer name in the ticket.	The user can view its name in the ticket.	Customer name (Ticket)	Pass
3.	Display Seat number in the ticket.	The user can view the seat number in the ticket.	Seat number (Ticket)	Pass

Test Case				
Test Scenario name		Logout		
Description		This scenario covers the functionality of exiting movie booking system		
Module		Exit Portal		
Status		Completed		
S/No	**Test case**	**Purpose**	**Actual Result**	**Expected Output (Pass / Fail)**
1.	Exit Portal upon clicking “Logout” button from homepage.	The user can exit portal anytime he/she desires.	Upon clicking logout, it directs to the login page.	Pass

6.2 Test Result

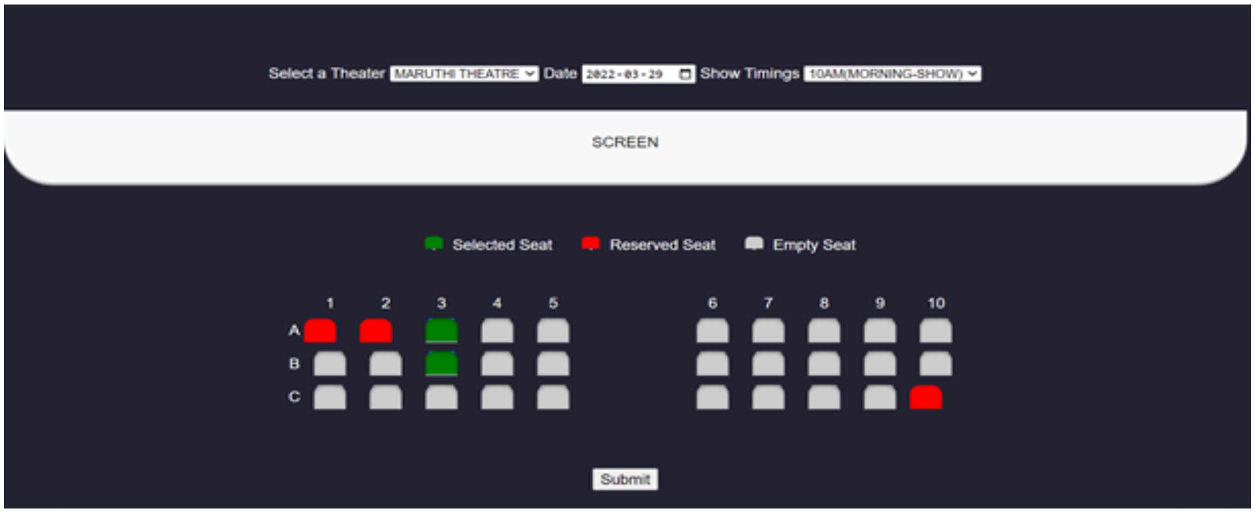
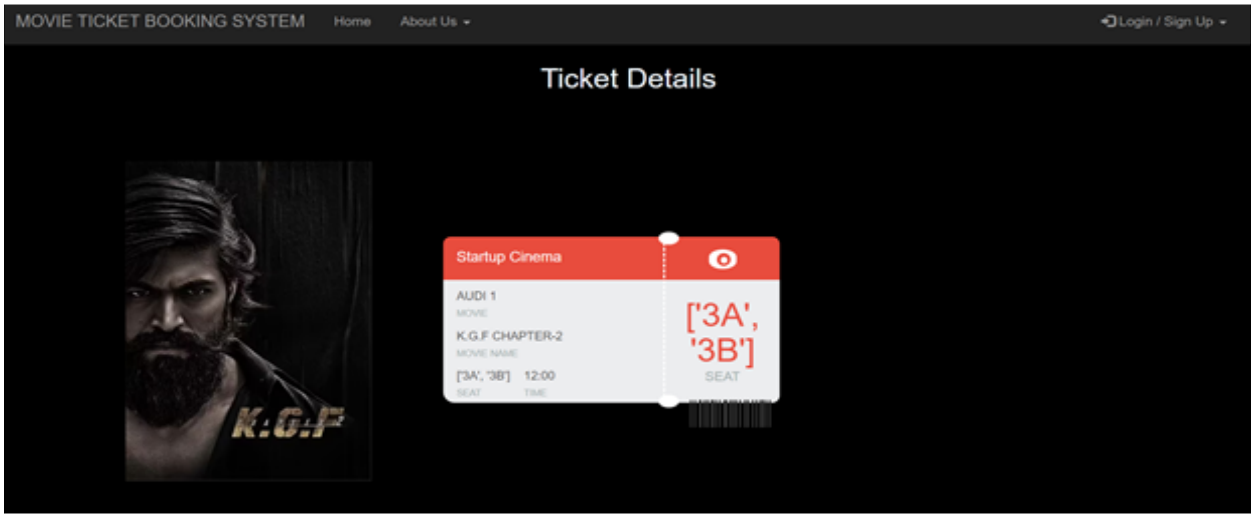


Figure 16, Seat Selection page showing Reserved, Selected and Empty seats



Startup Cinema

AUDI 1
MOVIE
K.G.F CHAPTER-2
MOVIE NAME
['3A', '3B'] 12:00
SEAT TIME

['3A', '3B']
SEAT

Figure 17, Movie Ticket Showing Movie, Seat and Customer name

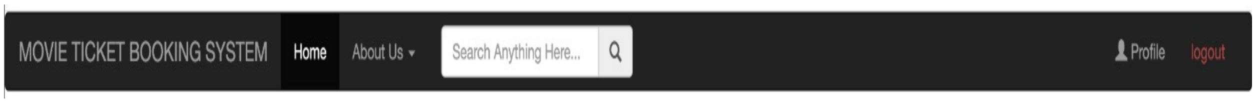
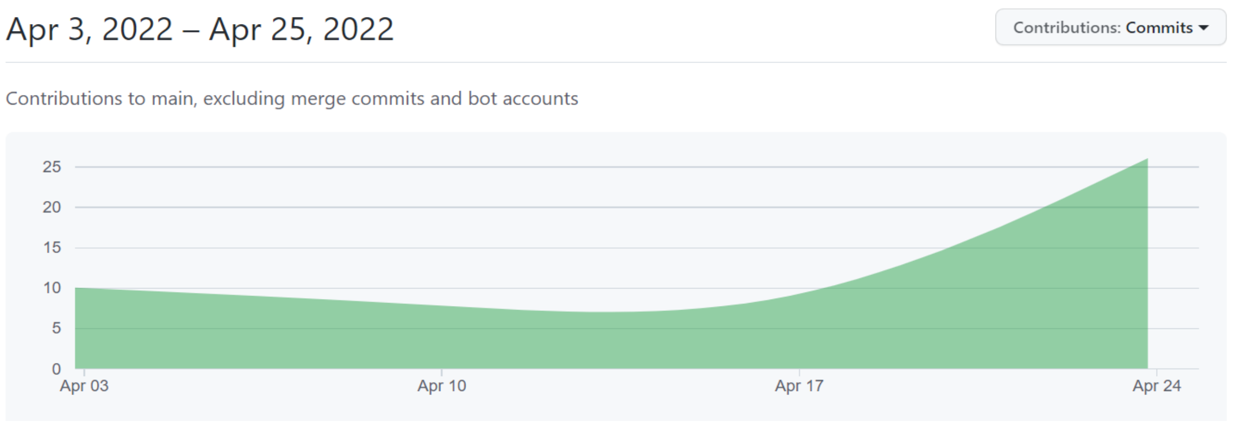


Figure 18, Logout button allows to exit user portal

7 INDIVIDUALCONTRIBUTION

Member	Contribution
Abishek Arumugam Thiruselvi (40218896)	1. Styling of seat selection, home page and ticket view page. 2. Worked on the styling of the admin page. 3. Deployed the software online using AWS.
Srinidhi Honakere Srinivas (40221128)	1. Worked on restoring the contents of database that was previously lost. 2. Worked on the backend part of the seat selection page. 3. Worked on the Admin part of the project. 4. Worked on the backend part of the movie detail's view page.
Manish Pejathaya (40194909)	1. Styled the main home page. 2. Created a ticket view page. 3. Styled the movie details view page. 4. Worked on the backend part of the seat selection page.
Mohammad Alikhanizanjani (40221406)	1. Designed the User interface of the seat selection page. 2. Organized sprint meetings. 3. Updated the blog and noted down the minutes of meeting.
Mohammad Abdul Wasay Ejaz (40185576)	1. Designing Main UI page (adding images, styling, navbar). 2. Modifying About us Page 3. Linking multiple pages. 4. Creating user profile and Registration page. 5. Creating user login and logout option.


Apr 3, 2022 – Apr 25, 2022

Contributions to main, excluding merge commits and bot accounts

Figure 19, Overall Contribution for 3rd Sprint