Software Requirements Specification

For

Online Movie Ticket Booking System

By

SRIMATHI.R (2018506125)

ROSHIN FARHEEN(2018506098)

Table of Contents

1. Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms and Abbreviations

1.4 References

1.5 Technologies to be used

1.6 Overview

2. Overall Description

2.1 Use-Case Model Survey

2.2 WEB Architecture diagram

2.3 ER Diagram

2.4 Architecture diagram

2.5 Data Dictionary

2.6 Assumptions and Dependencies

2.7 Requirements

3. Specific Requirements

- Use-Case model
- Use case Report
- Use case diagram

3.2 **Class based modelling**

- Class diagram
- Identify analysis classes
- CRC modelling
- Analysis packages

3.3 **Scenario based Modelling**

- Activity diagram
- Swimlane diagram

3.4 **Behavioral Modelling**

- Creating a behavioral model
- State representation and Diagram
- Analysis pattern
- Functional model

3.5 **Flow based modelling**

- Identify data elements
- Data flow diagram

3.6 **Design Modelling**

- Data design elements
- Design concepts
- Architecture design elements
- Interface design elements
- Component level design
- Deployment level design
- Cost estimation

1. Introduction

1.1 Purpose

This document is meant to delineate the features of OMBS,so as to serve as a guide to the developers .To develop a fully functional and user interactive online tool Movie ticket booking processing that allows customers to know about new movies,theatre locations (filtering based on city)and seat availability.Graphical layout of seat status is visible to the customer .Admin will be able to manage Movies and customer details.

1.2 Scope

- ➢ Admin can add,delete and update maintain movie details
- ➢ Admin can view the booked and cancelled tickets of customers.
- ➢ Admin can send confirmation messages for successful payment.
- ➢ Users can view movie details.
- ➢ Registered users can cancel the booked ticket and get a refund.
- ➢ Registered users can book tickets by selecting language and no of seats .
- ➢ Users can receive confirmation message from admin
- ➢ Users can update their profiles

1.3 Definitions, Acronyms and Abbreviation

Admin – Administrator.

HTML – HyperText Markup Language.

XHTML – Extensible Hypertext Markup Language.

HTTP – Hypertext Transfer Protocol.

JSP – Java Server Pages.

EJB - Enterprise Java Beans.

WAS – Websphere Application Server.

J2EE – Java 2 Enterprise Edition.

DB2 - DB2 Database Management System.

Doc – document.

1.4 References

IEEE SRS Format.
TGMC-2008 Sample Synopsis Format.
Problem Definition Provided By TGMC-2008

1.5 Technologies to be used

Application Architecture - JAVA
User Interface:swings,JSP
Database Application - MYSql
Web Deployment Server - WAS.

1.6 Overview

This project facilitates the customer to search for the movie details,availability of seats,prices of various seat types and also can cancel the booked tickets the graphical layout helps users to visualize seats.This project also helps admin to easily manage movie details,add movies,delete movies,add theatres in particular location and manage user details

2.OVERALL DESCRIPTION:

2.1 Use case template:

1.Administrator:

Responsible for All the activities in the system

A. add /Update /delete movies details:
B. add /Update /delete theatre details:
C. Update price of ticket
D. Managing user details-view booked history
E. Account the income
F. Send confirmation messages(ticket) for successful transaction

2.Bank-

Authenticate payment

A. Authenticate secure payments through third-party apps.
B. Provide Refund for cancelled tickets

3.Registered user-

Users who are already registered and have account

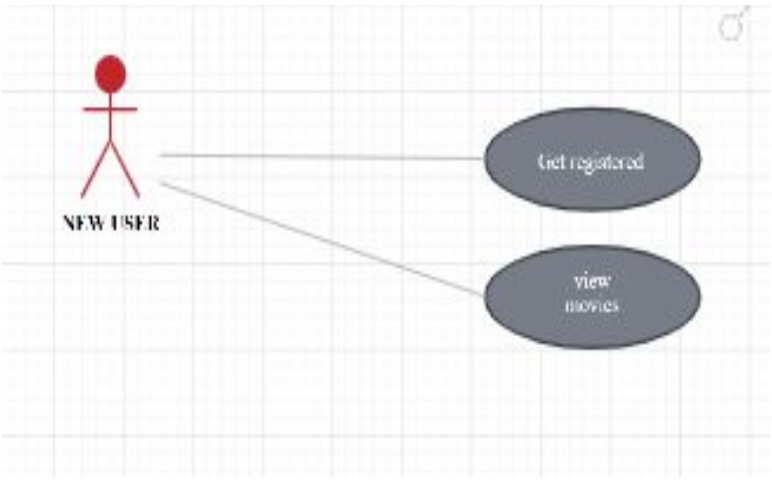
A. View movie details.
B. View graphical layout of seats
C. BOOK ticket by paying through card/net -banking
D. Cancel the ticket
E. View booked history
F. Update profile

4.New user- unregistered

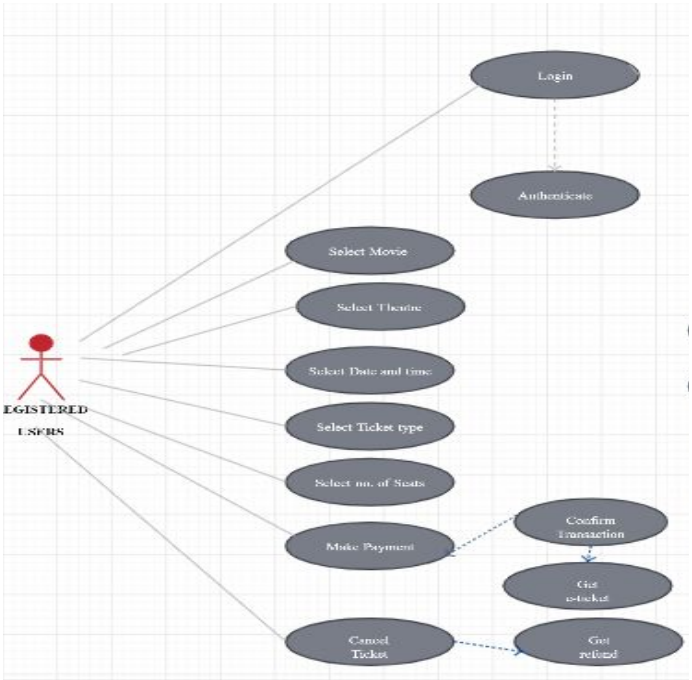
A. View movie details and price

USE CASE TEMPLATE:

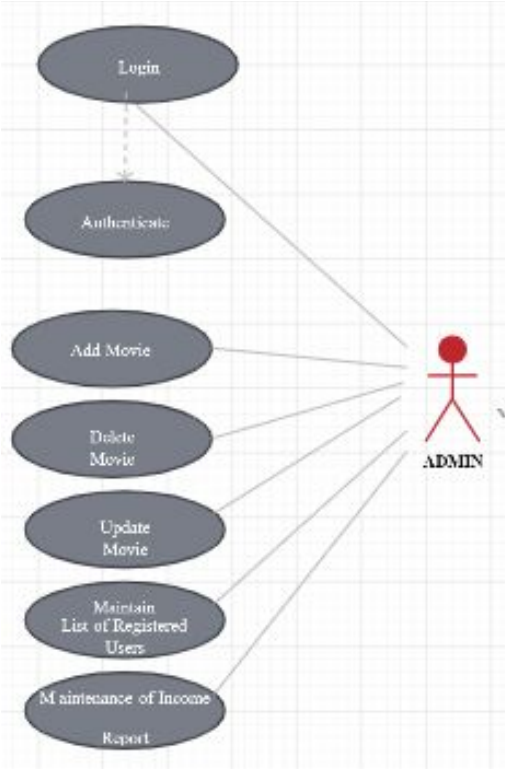
1.New users:



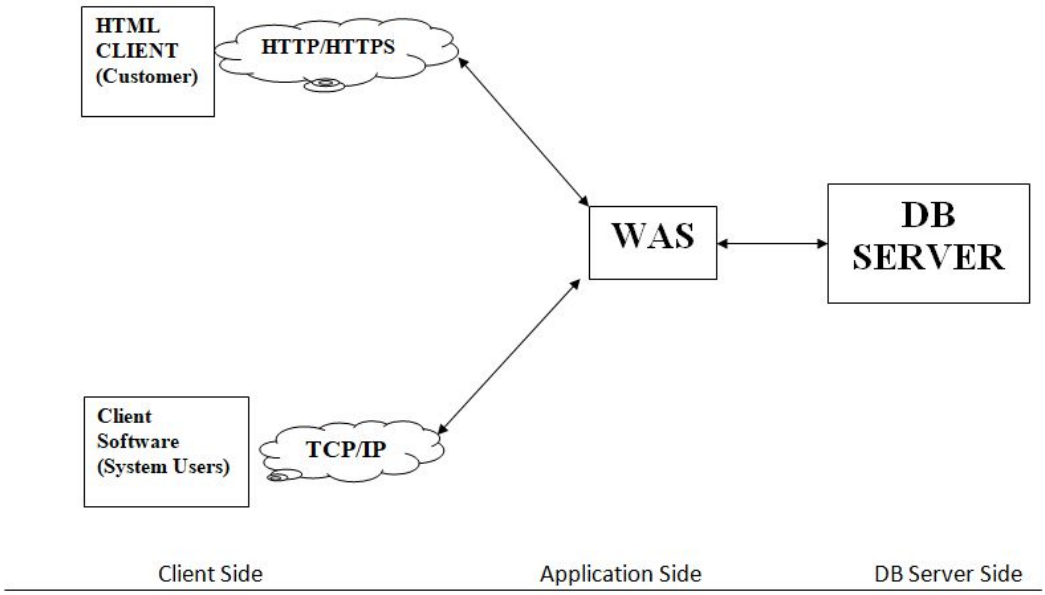
2.Exiting users:

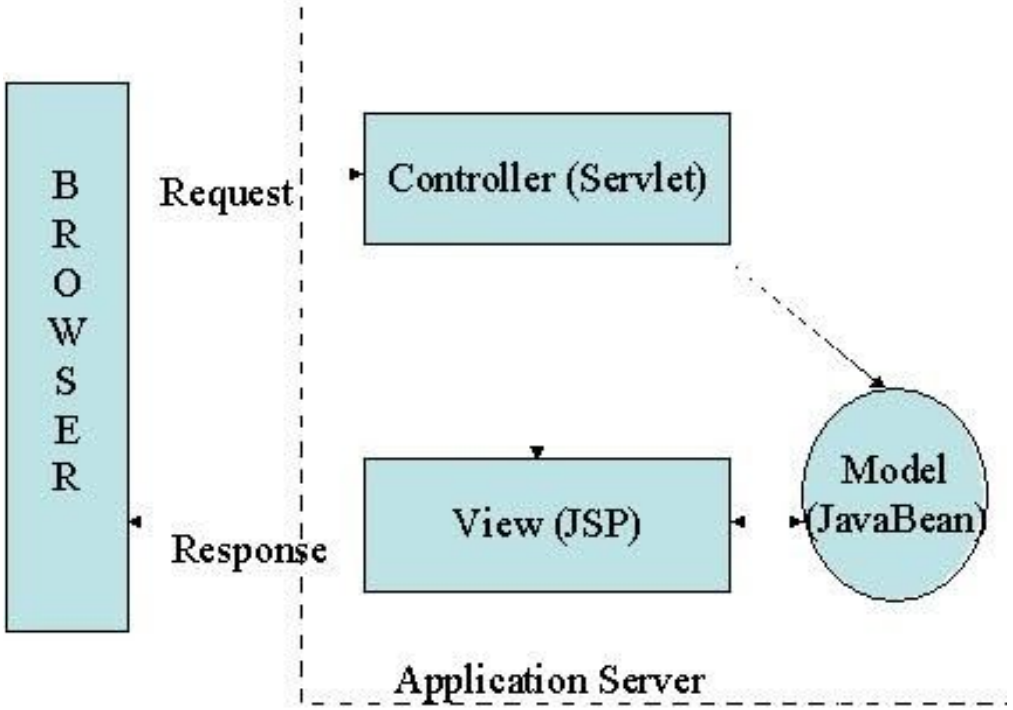


3.Admin:

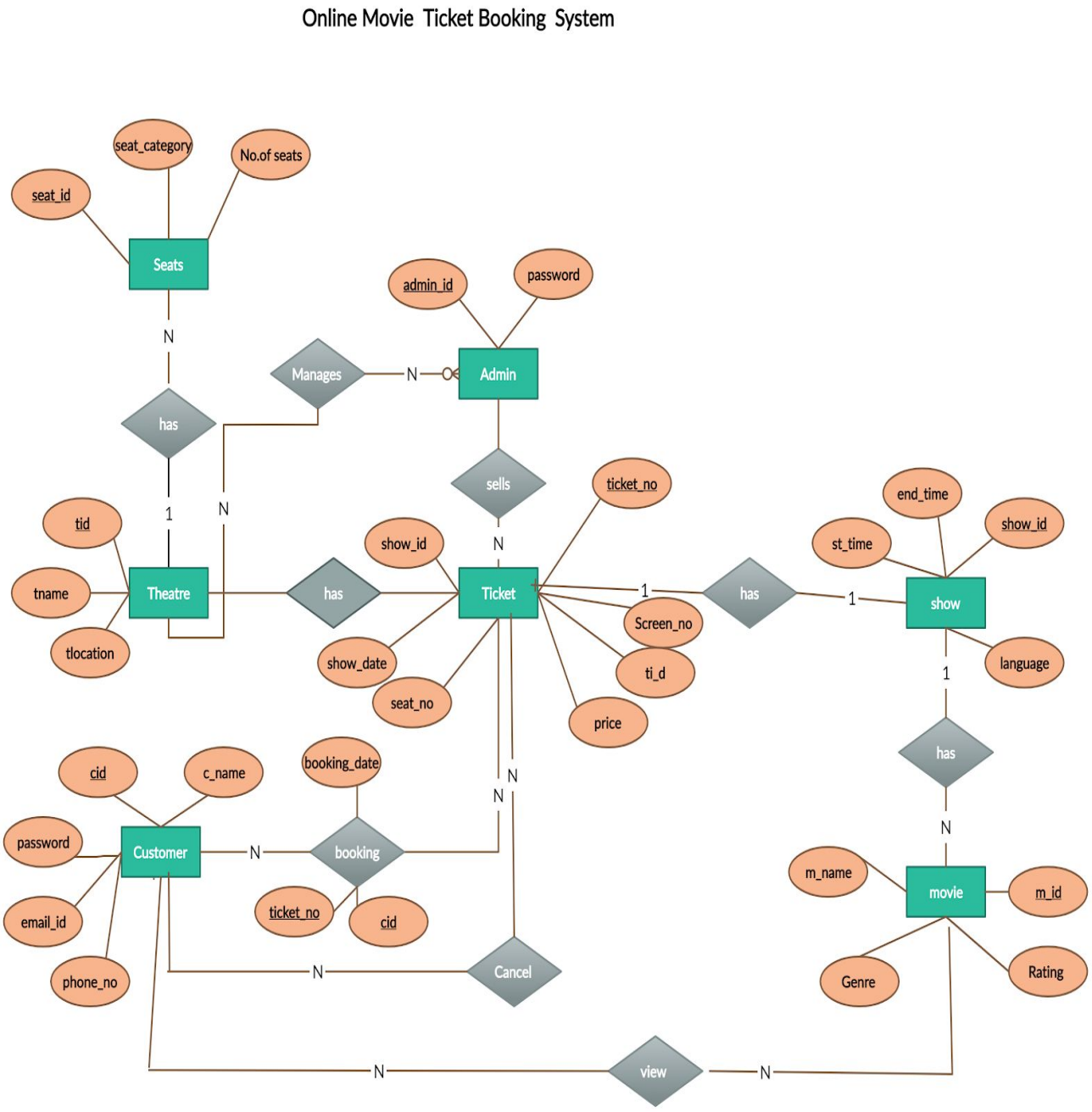

Login
Authenticate
Add Movie
Delete Movie
Update Movie
Maintain List of Registered Users
Maintenance of Income Report
ADMIN

2.:2 Web architecture diagram

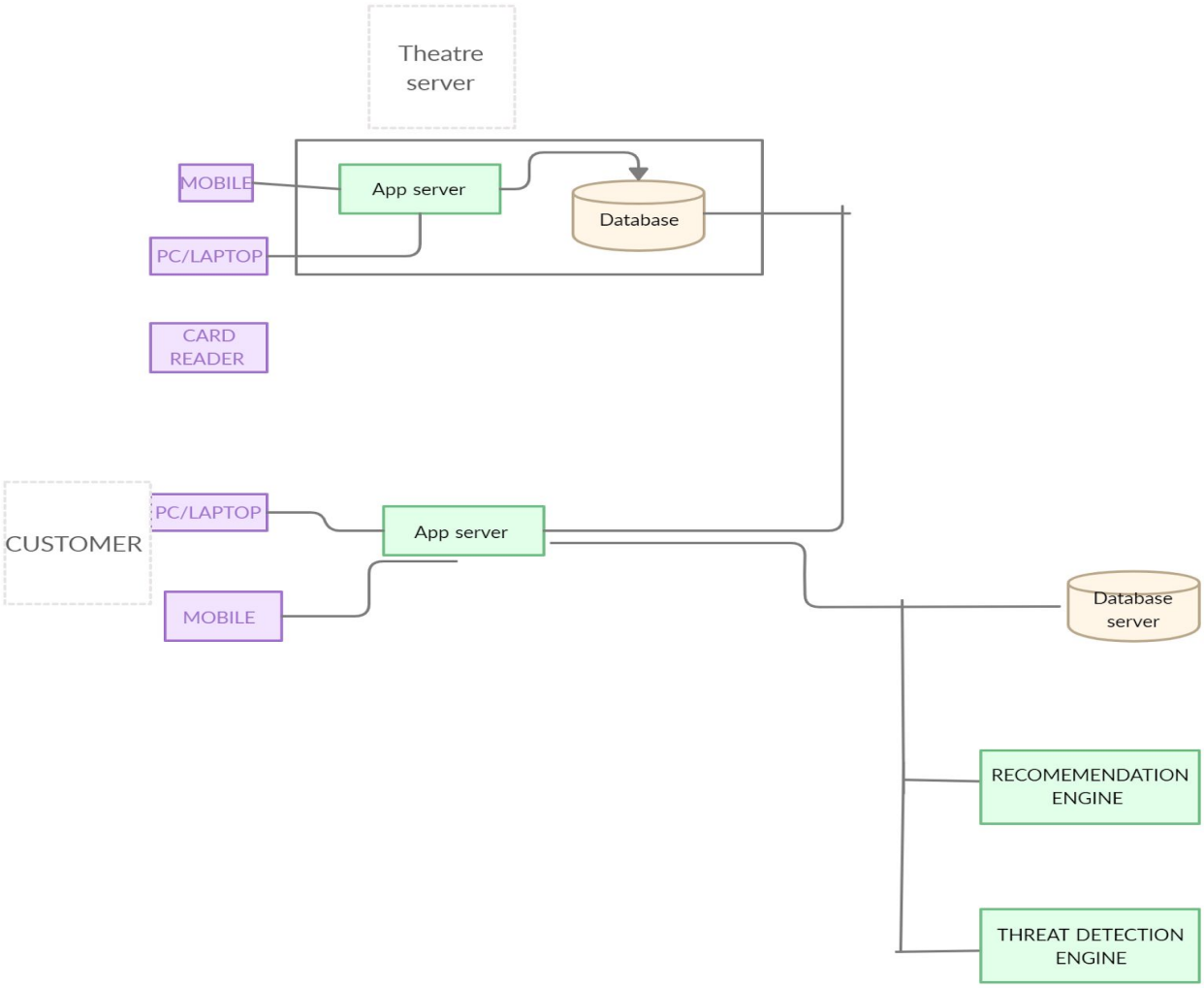




2.3 ER Diagram:



2.4 Architecture Diagram:



2.5 DATA DICTIONARY

ADMIN TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
a_id	integer	Admin id
password	varchar	Admin password

SEAT TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
s_id	varchar	Stores the seat no
s_category	varchar	Premiere,deluxe,vip
no_seats	Integer	No of seats available

Registered users table:

FIELDS	***DATA TYPE***	***DESCRIPTION***
cid	varchar	Customer id
cname	varchar	Customer name
email_id	varchar	Customer email-id
cpass	varchar	Customer login password
phone _no	Integer	Customer contact no

MOVIE TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
m_name	Varchar	Movie name
m_id	Integer	Movie id
Genre	Varchar	Genre of movie
Rating	Integer	Average rating given by various users

THEATRE TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
tid	Integer	Theatre id
tname	varchar	Theatre name
tloc	varchar	Theatre location(city)

TICKET TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
ticketno	Integer	Ticket number
show_id	Integer	Show id
show_date	Date	Show date
seat_no	Integer	Seat _no
price	Integer	Price of ticket
screen_no	Integer	Screen number in theatre

SHOW TABLE:

FIELDS	***DATA TYPE***	***DESCRIPTION***
st_time	time	Show starting time
end_time	time	Show ending time
s_id	integer	Show id
s_lang	varchar	Show language

2.6 Assumptions and Dependencies

- Client or user has active internet connection

- Client runs an operating system which supports web browsing or online application.
- When using the application it will assume that there is enough hard disk space for the file to be saved.

- Administrator is created in the system already.
- Admin has already been assigned with a id and password
- Roles and tasks are predefined.

2.7 .1 Requirements:

Functional Requirements

1. Registration -

If a customer want to book the ticket then he/she must be registered, an unregistered user can’t book the ticket. ts

2. Login -

Customer logins to the system by entering valid user id and password

3. Search Movie -

Customer or visitor can search movies based on movie name, date, time and venue

4. Seat Viewing -

The customer shall be shown a 2D image graphical layout of seats in the theatre.

5.Ticket **canceling -**

The customer shall be given an option to cancel the ticket 45 minutes before the movie with some fine.

6. Payment -

Users can pay by debit or credit card.

7. Logout -

After the payment or browse the movie, the customer will log out.

8. Generate ticket -

System sends a confirmation message to the registered mobile number.

9. Add ,remove and update movies -

The system shall have a feature for admin to add movies,Update and delete and their details.

Non-Functional Requirement

1. Security -

The system must automatically log out all customers after a period of inactivity.

The system should not leave any cookies on the customer's computer containing the user's password.

The system's back-end servers shall only be accessible to authenticated administrators.

2. Availability -

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the downtime of the server on which the system runs.

3. Maintainability -

A database is used for maintaining the data

4. Portability -

The end-user part is fully portable and any system using any web browser,any OS.

The system shall run on PC, Laptops, smartphones and PDA etc.

5. Accessibility -

The system will be a web-based application. It is accessible on the web browser.

6. Back up -

Need backup in our system database. In order to enable the administrator and the user to access the data from our system!

7. Performance -

The product shall be based on the web and has to be run from a web server.

The product shall take initial load time depending on internet connection strength

The performance shall depend upon hardware components of the client/customer

8. Accessibility -

The system shall provide multi-language support.

9. Supportability -

The source code developed for this system should be supported by all systems

2.7.2 Phases :

1. Communication

In the communication phase, the major task performed is ***requirement gathering*** which helps in finding out the exact need of the customer.

Stakeholder Requirement

1.The user should be able to view the list of movies which are running near to his location(based on GPS).

2.The user should be able to select the seat as per his choice in the hall.

3.The user should have different options for payment.

4 The system shall enable the user to search.

5.The system shall authenticate user credentials to login

6. The system shall send an order confirmation to the user through email.

2. Planning

- keeping tracks on the processes
- estimations related to the project are done.
- Planning is even used to find the types of risks involved throughout the projects.

3. Modeling

- .An analysis is carried out and depending on the analysis a software model is designed.
- Different models for developing software are created depending on the requirements gathered in the first phase and the planning done in the second phase.

4. Construction

- .The actual coding of the software is done in this phase
- . This coding is done on the basis of the model designed in the modeling phase
- . In this phase software is actually developed and tested.

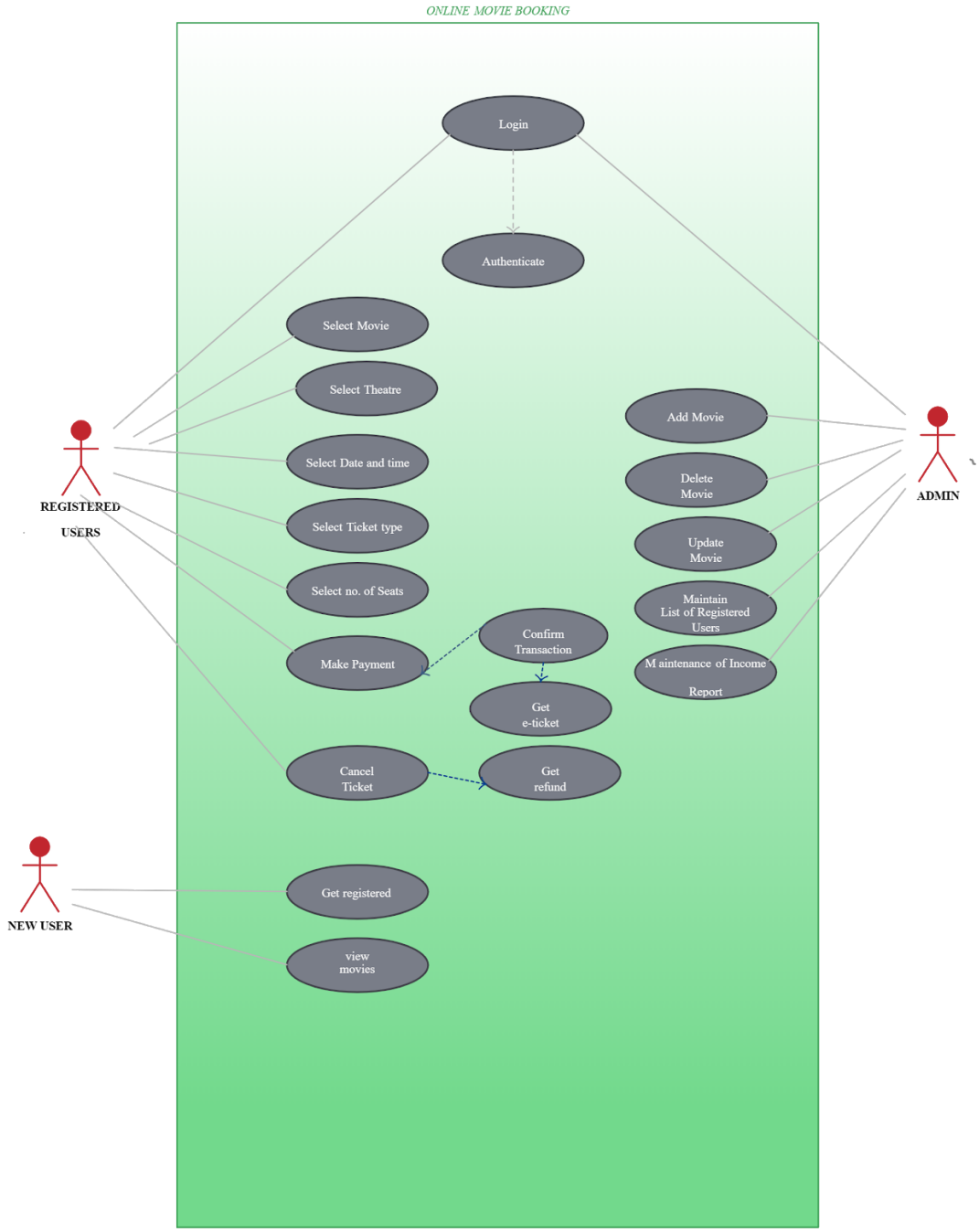
5. Deployment

- In this last phase, the product(OMBS) is actually rolled out or delivered
- installed by customer
- Feedback is taken from the customer about the movie based on which rating is displayed on the website to ensure the quality of the product.

3. Specific Requirements :

3.1 use case model

3.1.1 Use case Diagram:



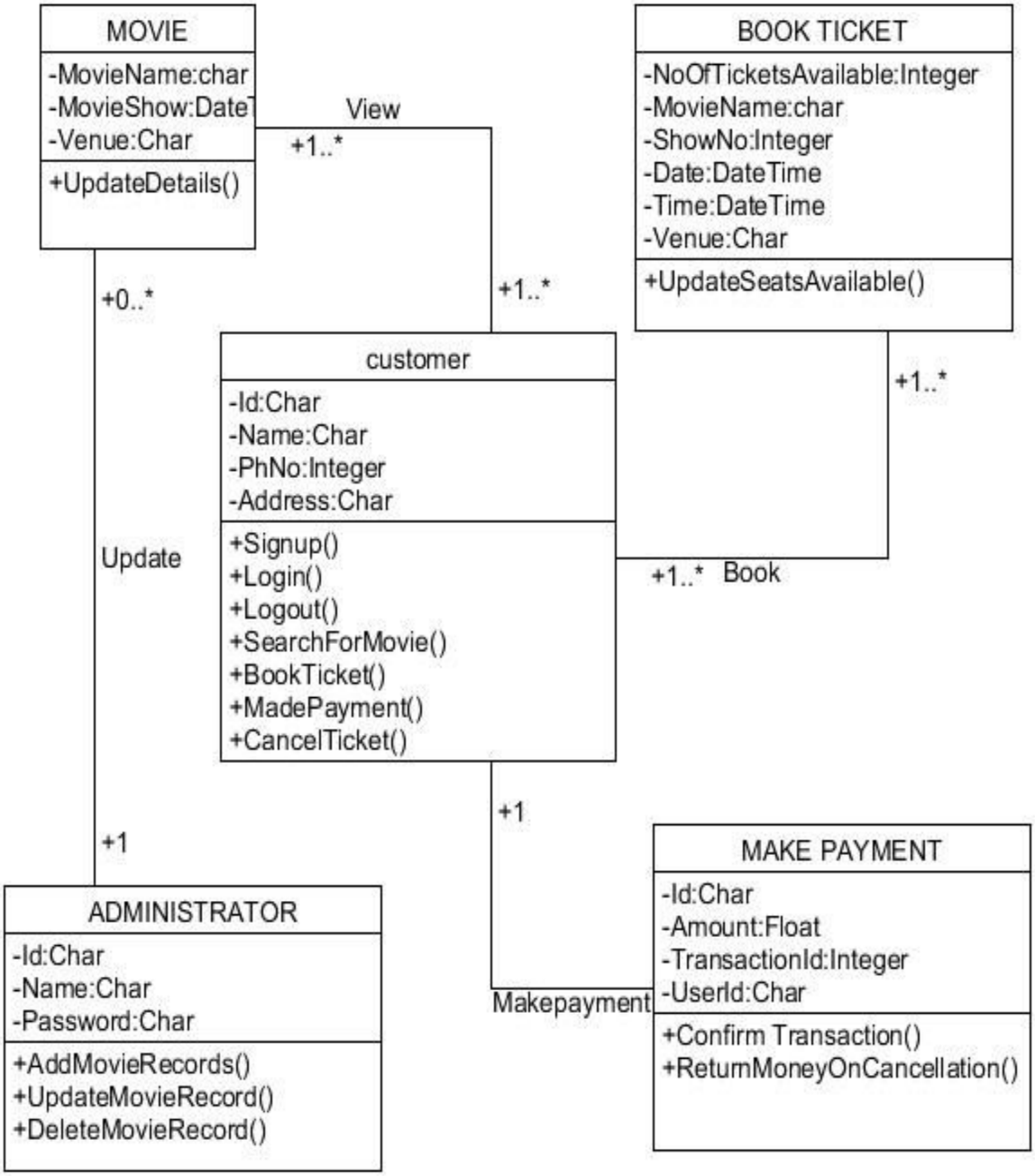
3.1.2 Use case report:

Use case	bookingticket
Primary actor	Customer
Secondary actors	Admin Theatre rep
Primary actors Goal	To book the selected movie for customer.
preconditions	Admin has been defined by system and given password credentials.
Trigger	The customer decides to "*book*"ticket
Main functions	To login with credentials given To search for movie To book ticket (make payment) To cancel ticket To view booked history
Scenario	User:search for movie User:make payment User:cancel ticket
Exceptions	1.password is incorrect:Re-enter credential details 2.payment unsuccessful: If amount debited must be refunded and logged out 3.Seat unavailable: Re-select theatre and given notifications
variations	1.change of password(profile updation) 2.cancel booked ticket
Frequency of use	For 1 customer (1 to 2 times a day)

priority:	essential
Secondary Actors	Admin Theatre Rep.
Admin goals	Manage user information Maintain income reports Add/update/delete movie details Add/update/delete theatre details
Channel	internet
Constraints	Any updation should be correctly changed. For unsuccessful transaction payment must be refunded(if credited)

3.2 Class based modelling :

3.2.1Class diagram:



3.2.2Crc modelling :

Class Name :Admin
RESPONSIBILITY: manage user booking and cancellation details manage income report add movie details delete movie details update details
COLLOBORATOR: customer movie theatre

Class Name :show
RESPONSIBILITY: displays show start time displays show end time. displays show date and time displays screens in theatre display price of show
COLLOBORATOR: admin movie theatre

Class Name :customer
RESPONSIBILITY: log in with credentials search movies make payment viewed booking history cancel ticket
COLLOBORATOR: admin

Class Name :ticket
RESPONSIBILITY: displays booking date and time displays price of the ticket displays the seat type
COLLOBORATOR: admin

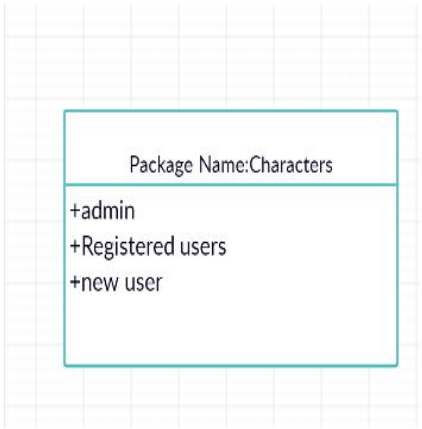
Class name:payment
RESPONSIBILITY: refund for cancellation request confirmation messsage for successful transaction
COLLOBORATOR: admin

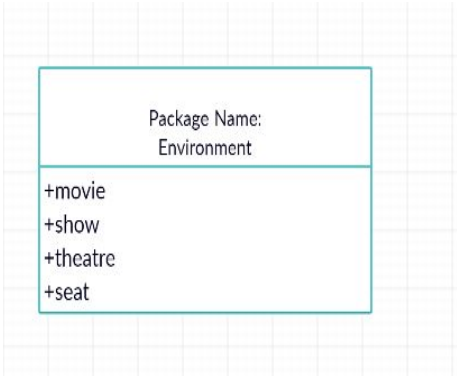
Class name:Movie
RESPONSIBILITY: gives movie language displays release details display movie rating displays movie genre
COLLOBORATOR: admin show theatre

2.2.3 IDENTIFYING ANALYSIS CLASSES:

Potential class	General classification
admin	External entity
customer	External entities
booking	occurrence
ticket	Thing
show	External entity
language	Attributes of show
theatre	places
password	Thing
payment	Organizational unit

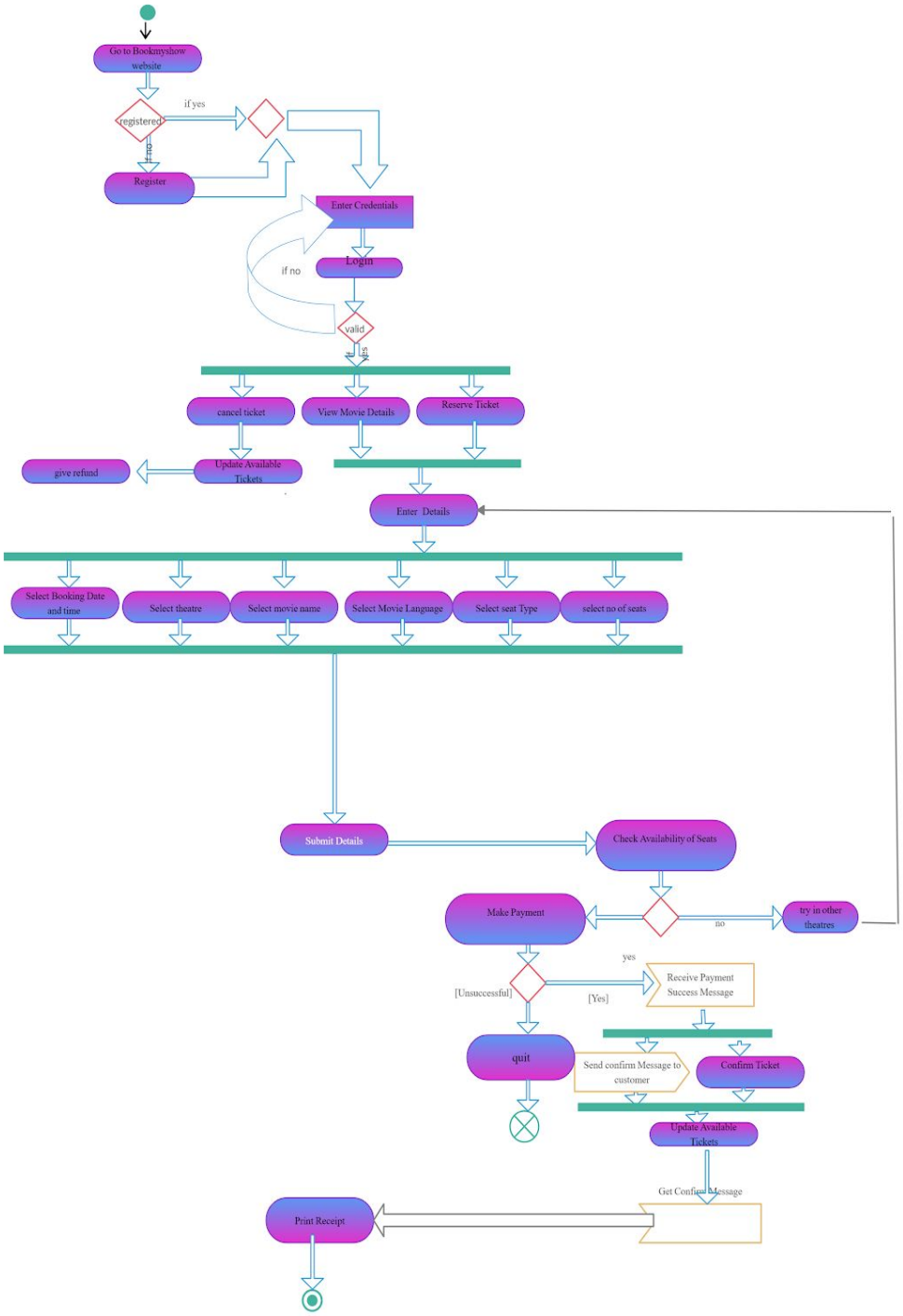
2.2.4 Analysis packages:



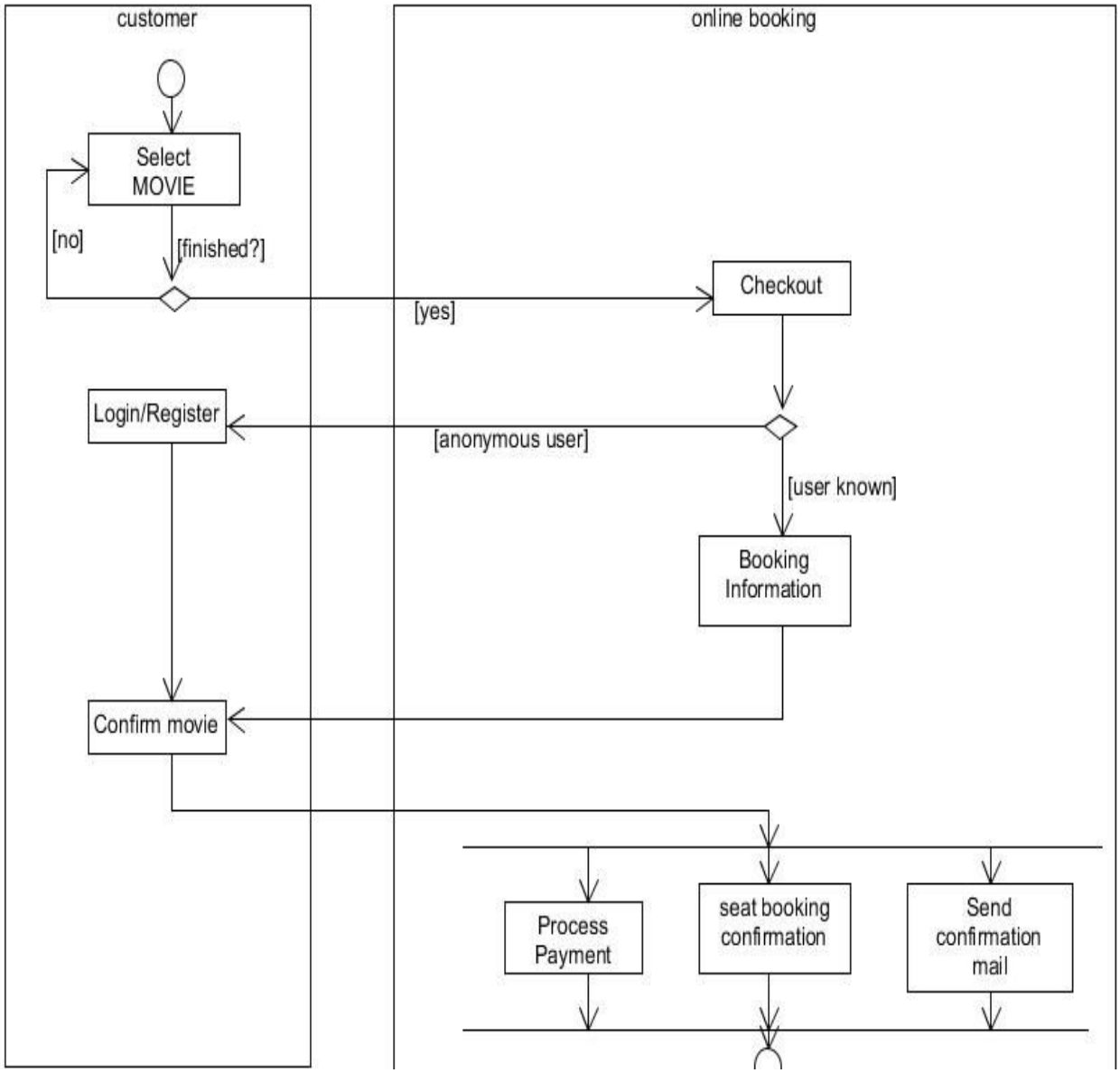


3.3 Scenario based elements:

3.3.1Activity diagram:



3.3.2 SWIMLANE DIAGRAM:



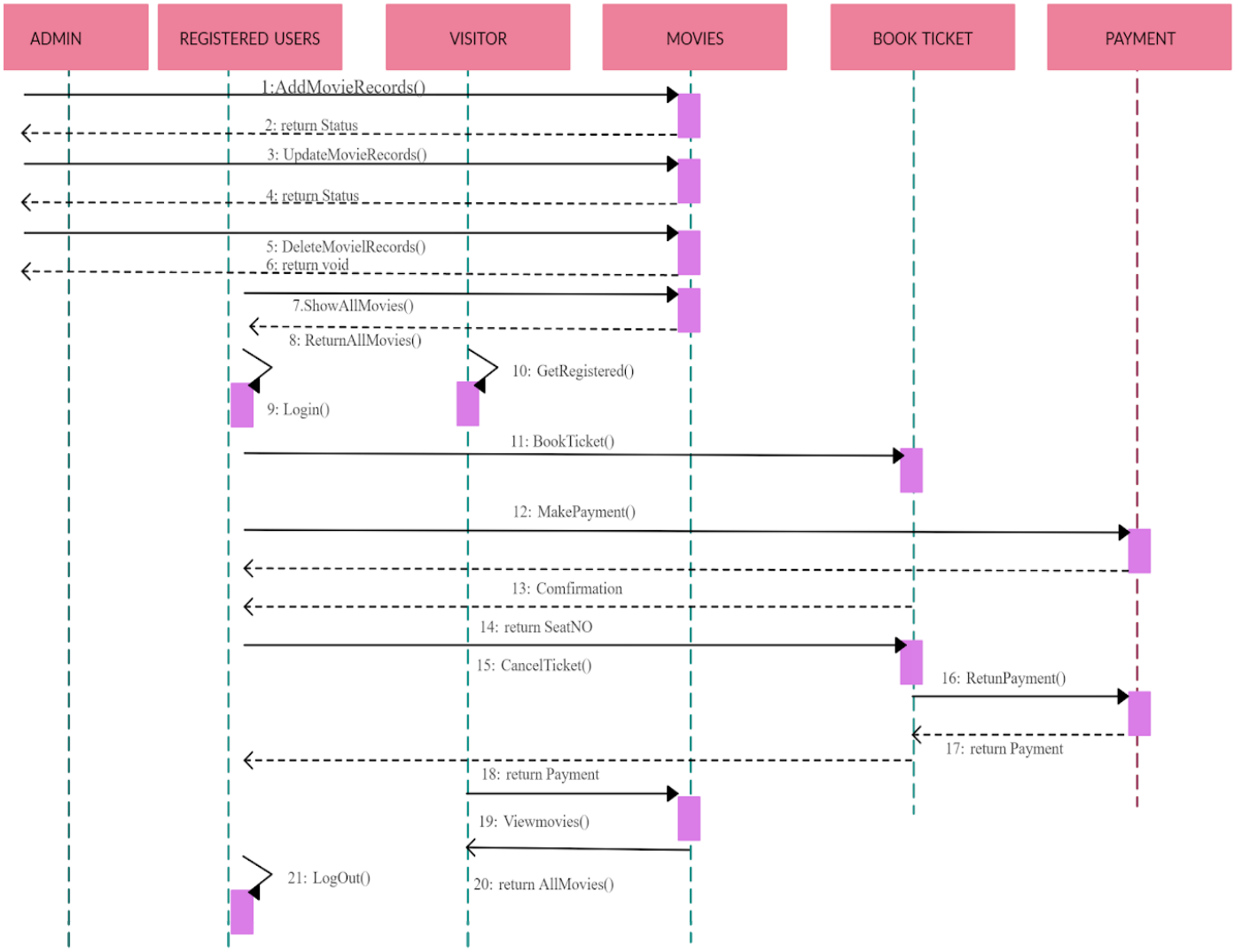
4.BEHAVIOURAL MODELLING:

4.1 CREATING BEHAVIOURAL MODEL:

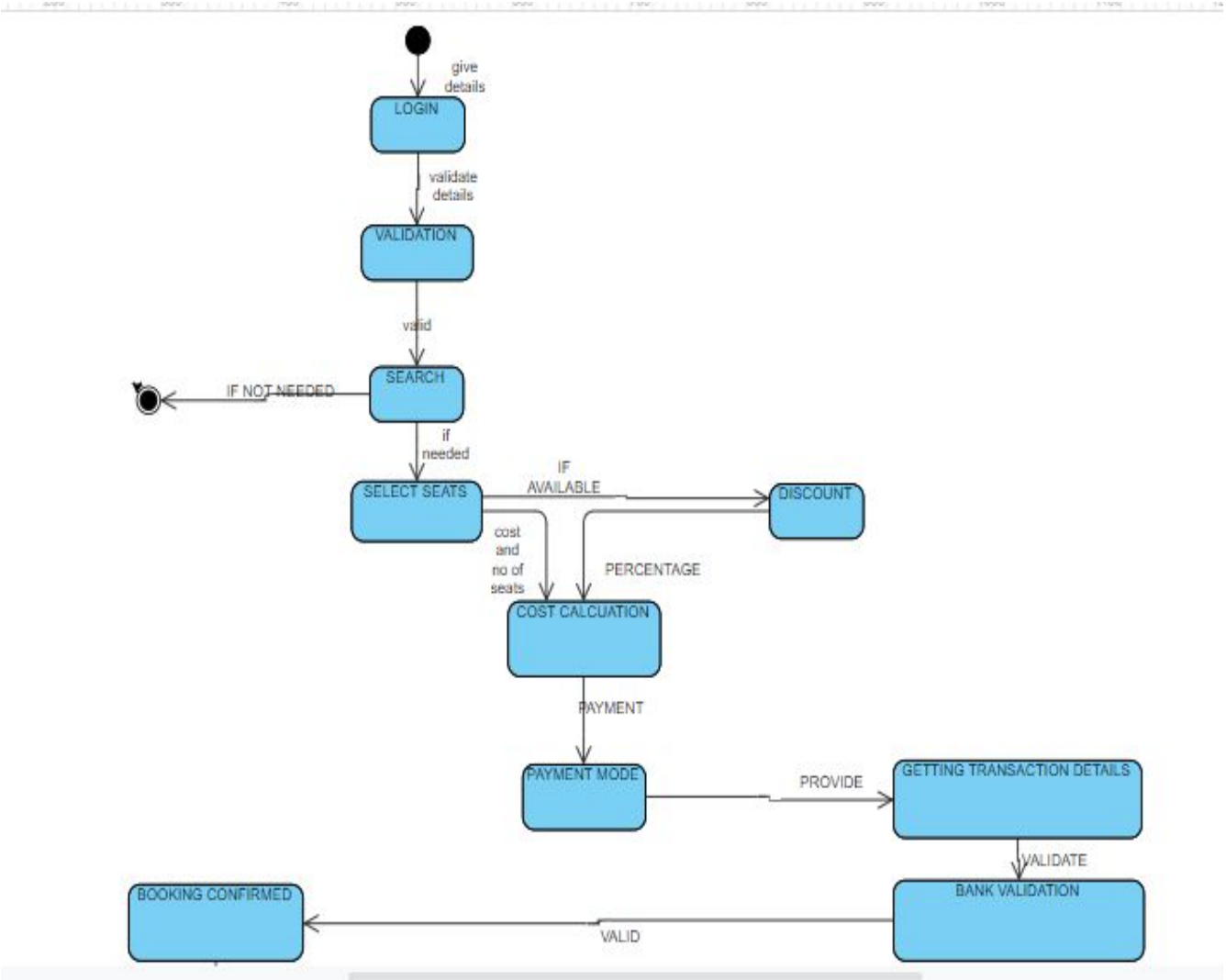
The behavioural model describe the control structure of the system.

So,we use sequence diagram and functional model representing control structure of the system.

SEQUENCE DIAGRAM:



STATE DIAGRAM:

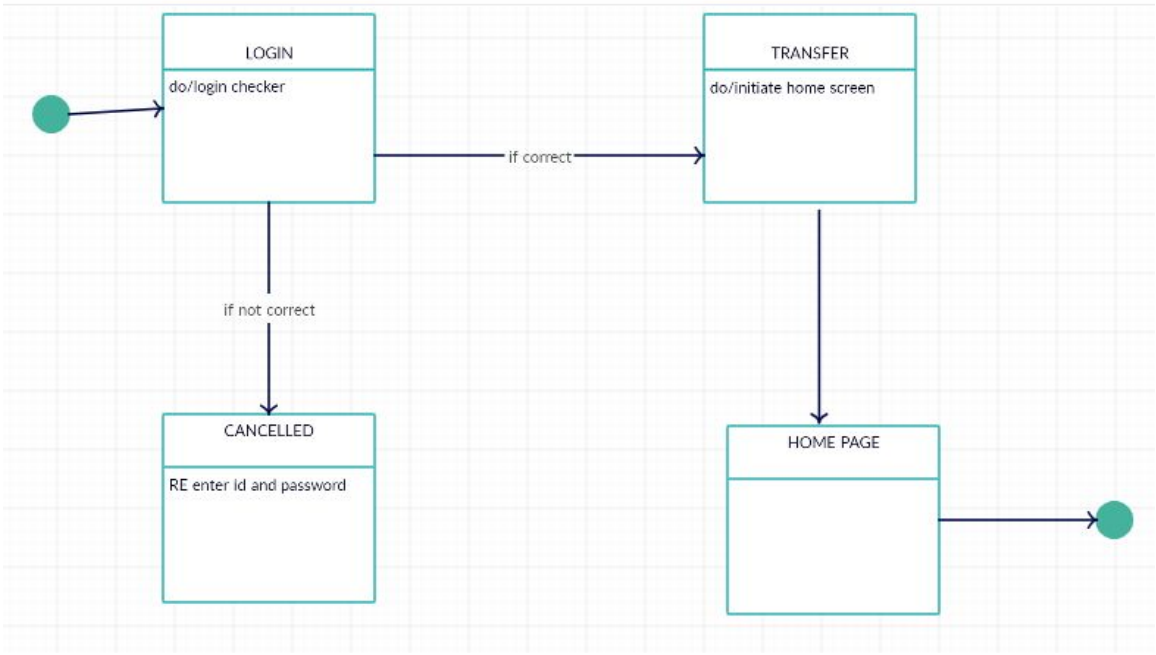


State representations:

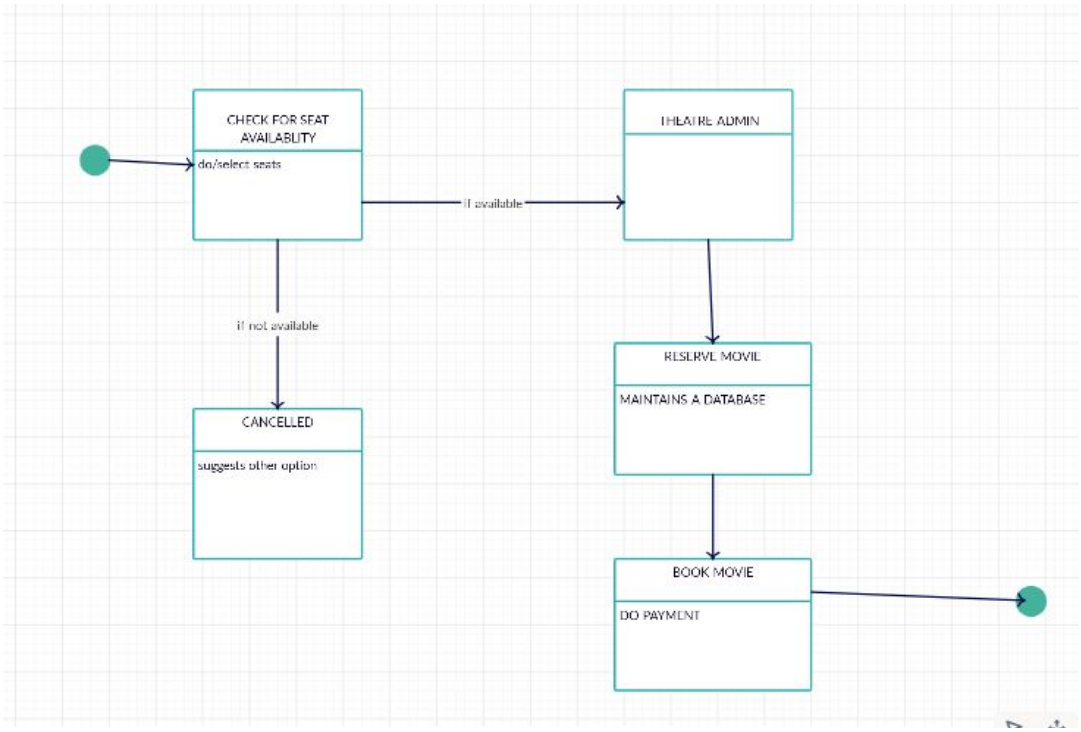
- ❏ Select account
- ❏ Validation
- ❏ Select theatre
- ❏ Select seats
- ❏ Payment
- ❏ Send confirmation message

State diagrams of analysis classes:

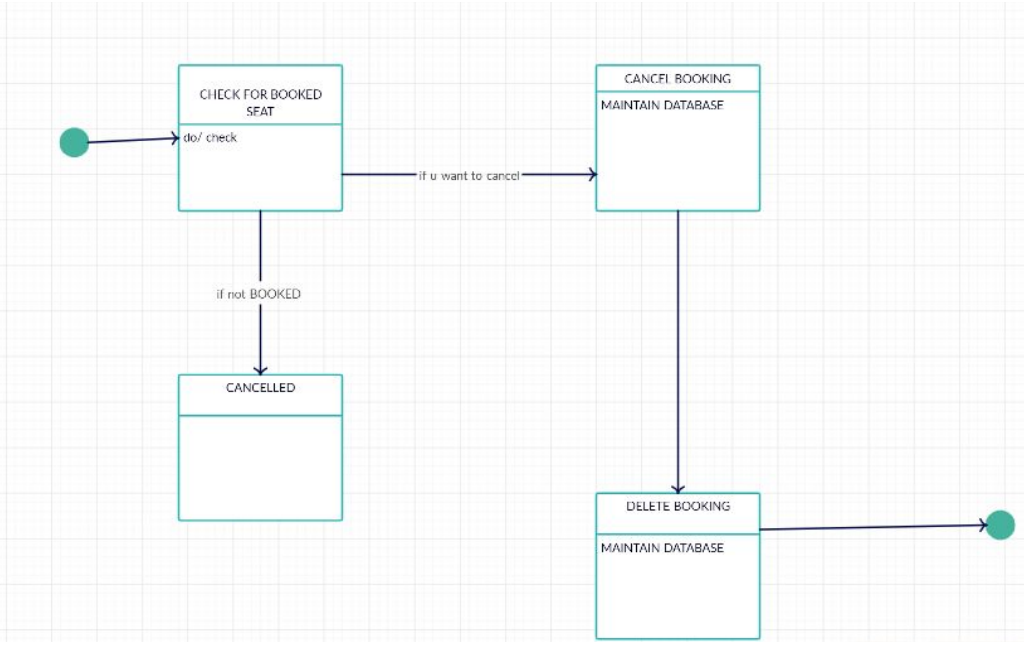
1:login:



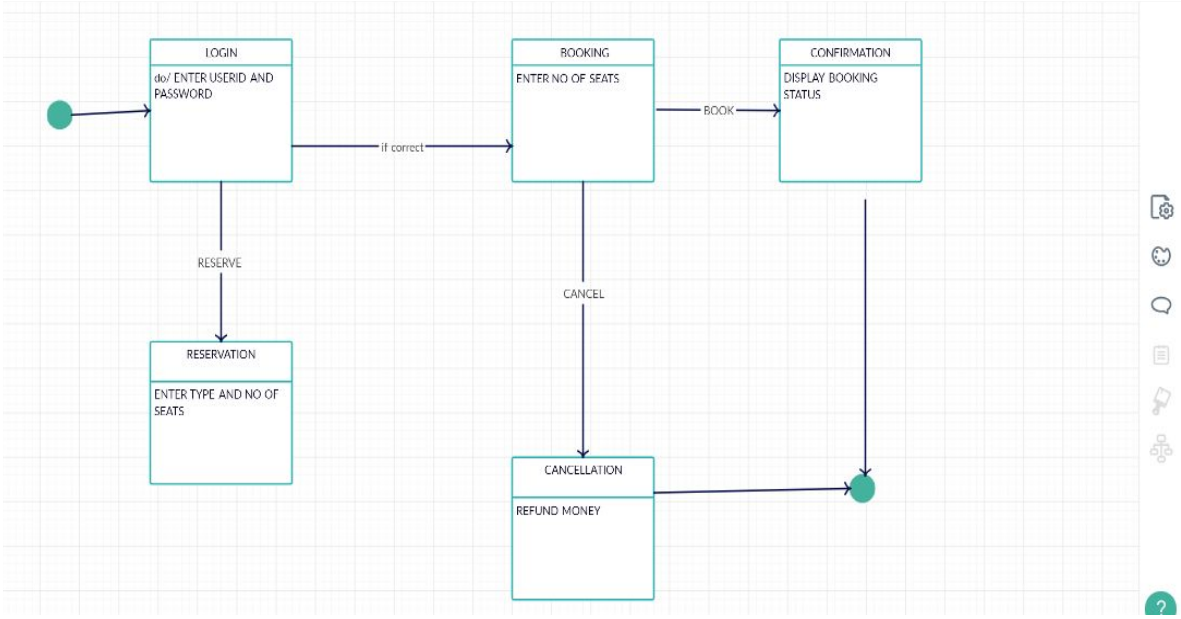
2:Reservation



3:cancellation:



4:system:



3.4.3 ANALYSIS PATTERNS:

Pattern name : queueing system

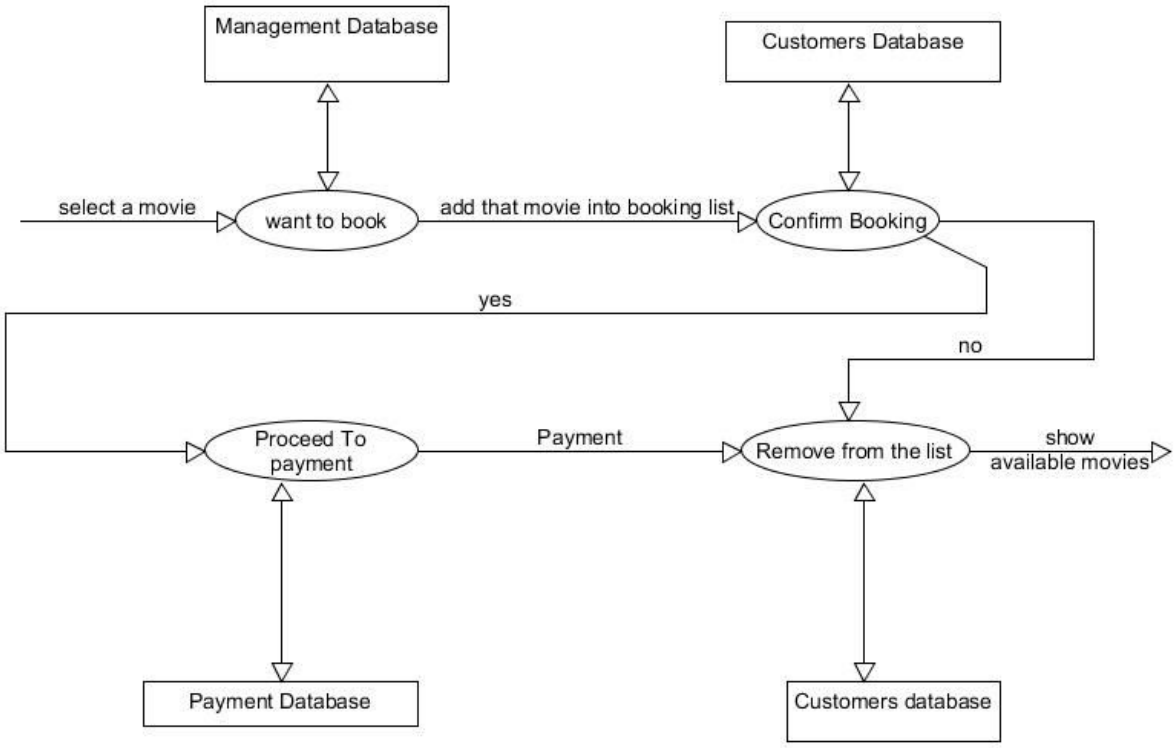
Intent : queueing the customers in a controlled first come-first serve

Motivation : By managing online traffic inflow with an online queue system, you insure against failure along the whole customer journey and keep your site and app available 24/7, no matter the demand.
Context : When online visitors go beyond your website capacity, they are offloaded to a customizable online queue and then passed back into your website or app in a controlled first-come, first-served order. Passing visitors onto your website or app in a first-in, first-out order makes the online queuing process fair and transparent to each visitor. While waiting in the online queue, you can inform your visitors of their number in line, expected waiting time, expected arrival time on the website and any other additional information you would like to share with them.

Solution :
By implementing Queue-it, we were able to process major ticket on sales smoothly, using the virtual queue system as an effective website overload management tool.
Consequences : This can occupy large amounts of storage. It should be updated manually to delete and manage users in the queue.if anything goes wrong customers will starve.

3.4.4 FUNCTIONAL MODELS:

Here are some functional models which describe the function and processes which deal with the information needs,identify opportunities and establish a basis for service costs.



3.5 Flow modelling

3.5.1 Identify data elements:

Entities:

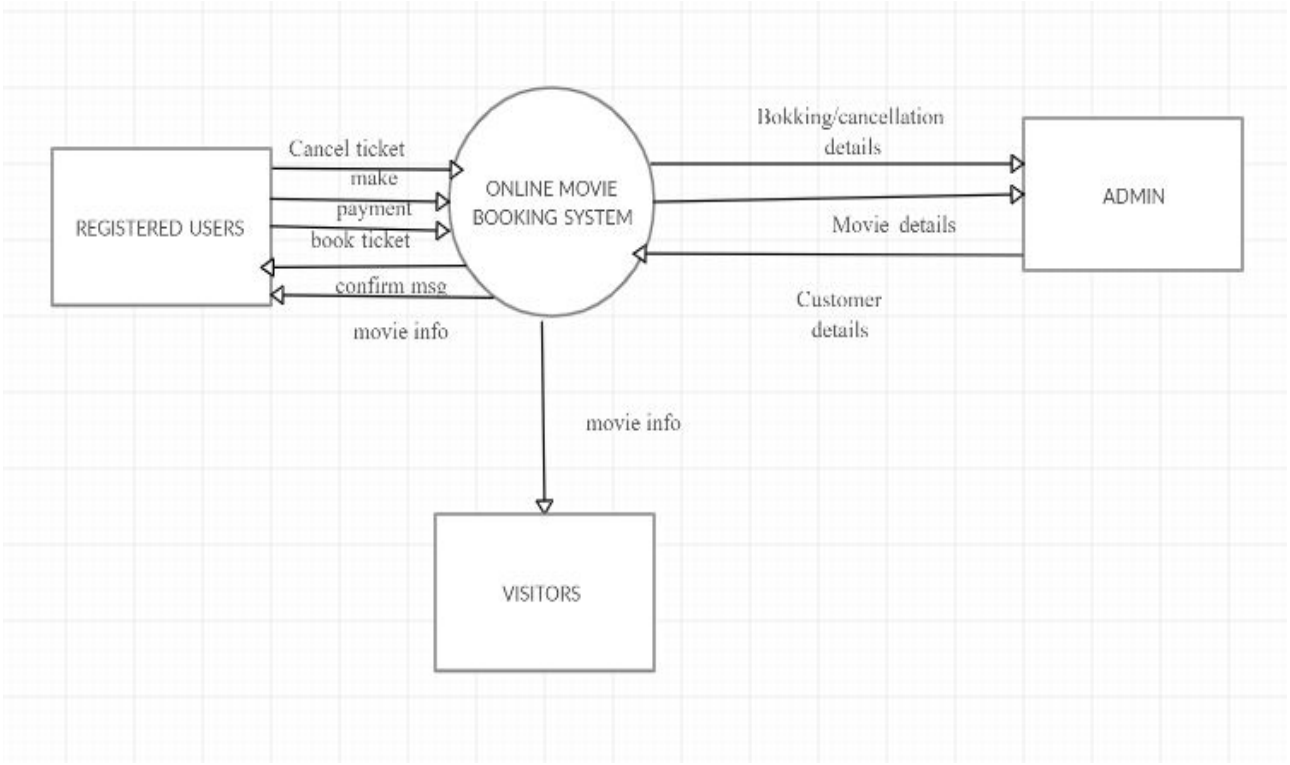
- Customer management
- Administration
- Show management
- Seat management

Processes:

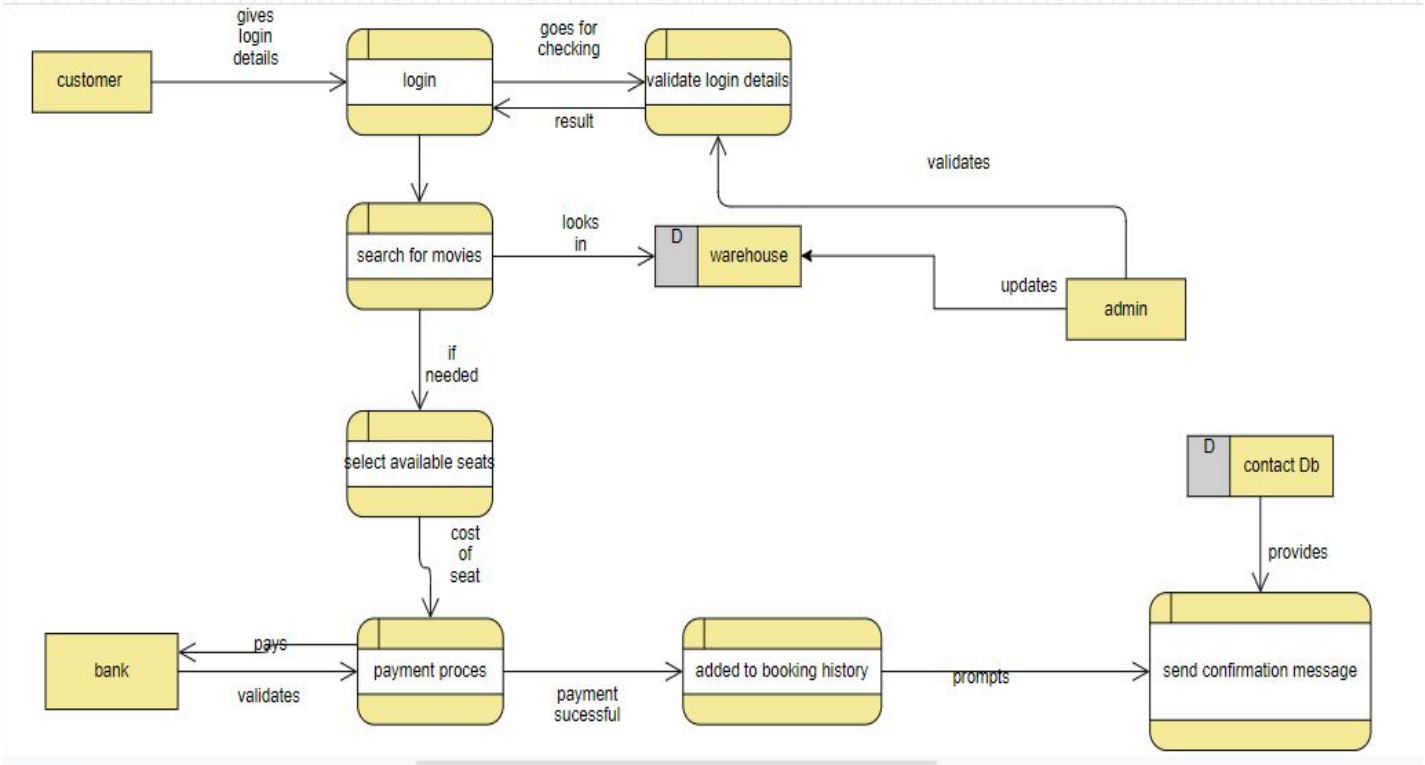
- Login
- Add to cart
- Book seats
- View booking history
- Cancel ticket
- logout

3.5.2DATA FLOW DIAGRAM :

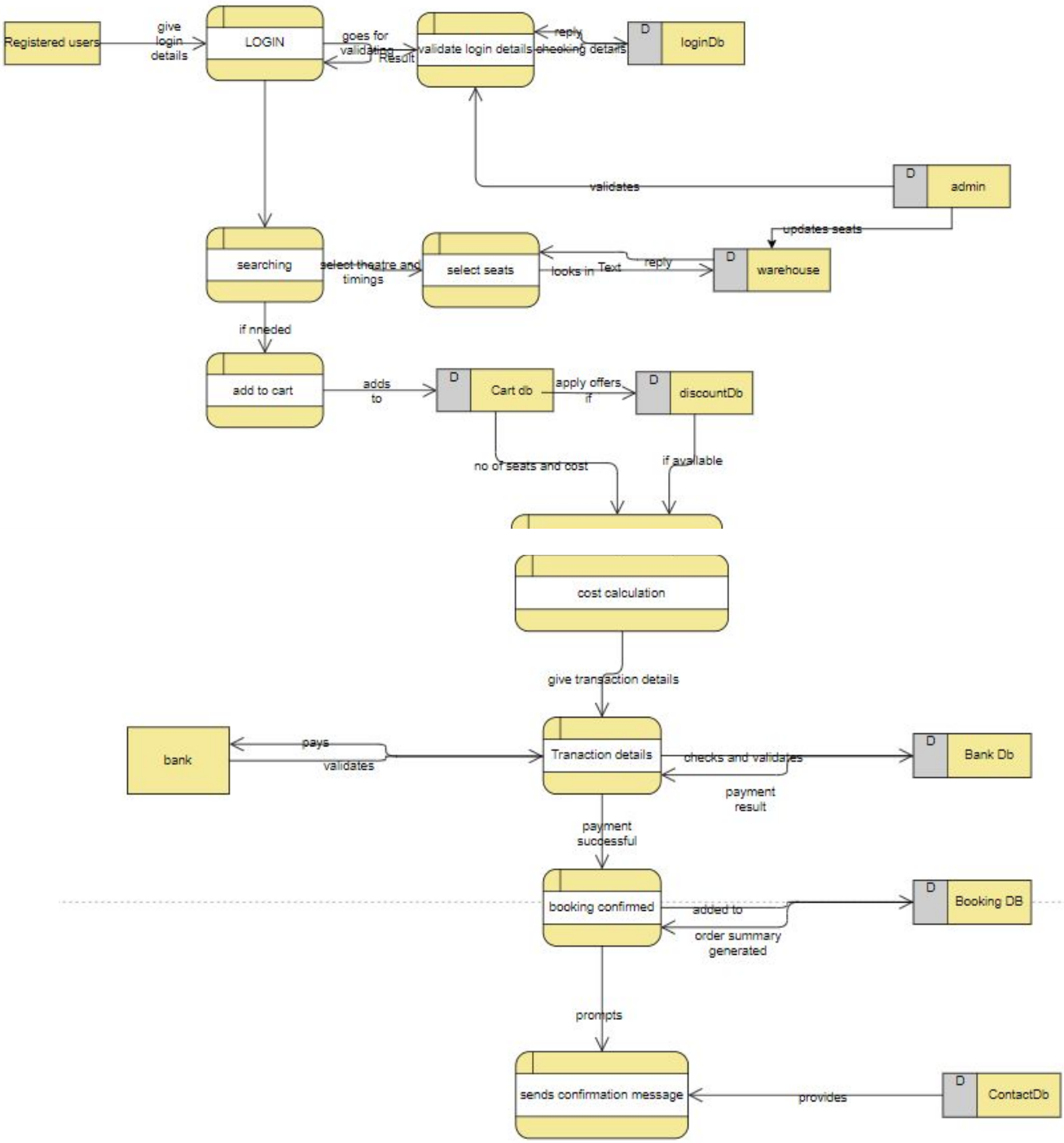
0-level DFD:



1-level DFD:


customer
gives login details
login
goes for checking
validate login details
result
validates
search for movies
looks in
D
warehouse
updates
admin
if needed
select available seats
D
contact Db
provides
cost of seat
bank
pays
validates
payment proces
payment sucessful
added to booking history
prompts
send confirmation message

2-level dfd:


Registered users
give login details
LOGIN
goes for validating
Result
validate login details
reply
checking details
D
loginDb
validates
D
admin
updates seats
searching
select theatre and timings
select seats
looks in
Text
reply
D
warehouse
if nneded
add to cart
adds to
D
Cart db
apply offers if
D
discountDb
no of seats and cost
if available
cost calculation
give transaction details
bank
pays
validates
Tranaction details
checks and validates
payment result
D
Bank Db
payment successful
booking confirmed
added to
order summary generated
D
Booking DB
prompts
sends confirmation message
provides
D
ContactDb

3.6 Design Modelling

3.6.1 Design Concepts

3.6.1.1 Encapsulation:-

- When we consider a modular solution to any problem, many levels of abstraction can be posed. As different levels of abstraction are developed, we work to create data abstraction and procedural abstraction.

- In this context, for procedural abstraction, we consider a procedure booking a seat as an example implies a long sequence of procedural steps like adding seats, updating database, confirming booking, etc.

- For data abstraction, we consider Customers . Each and every online customer has attributes like username, firstname, last name, e-mail ID and password.

- It follows that the procedural abstraction
- Booking tickets would make use of the attributes of the data abstraction customer.

3.6.1.2 Abstraction :-

Abstraction is the process of hiding complex properties or characteristics from the software itself to keep things more simplistic. This allows for a much higher level of efficiency for complex software designs since it allows the developers to list out only the necessary elements or objects required.

In our context of online movie booking system, the module which performs the function of validating User ID and password doesn't need to know about the prices of seats or the current state of the database which stores information about seats available.

3.6.1.3 Modularity :-

a complex problem can be more easily handled if it is subdivided into pieces that can each be solved and/or optimized independently.

In our context, we consider the event of the user in search of a movie. This leads to a complex problem. In the absence of a drop down list, the user has to go through all theatres until he finds his required theatre.

Hence, the problem of searching is divided into two subproblems : searching for a theatre/movie using a search keyword and processing that search keyword to display relevant details.

Thus, the amount of work done by the user is reduced.

3.6.1.4 Aspects :-

An aspect is a representation of a crosscutting concern. In our context, we consider two requirements that crosscut each other.

1)ability of an online customer to book seats using the application.
2)Users can use the application only if the user is logged in using his/her user ID and password.

So, aspect is implemented as a
separate module (component) rather than as software fragments that are “scattered” or “tangled” throughout many components.

3.6.1.5 Refactoring :-

- Refactoring is a reorganization technique that simplifies the design (or code) of a component without changing its function or behaviour. Refactoring is the process of changing a software system in such a way that it does not alter the external behaviour the code (design) yet improves its internal structure.

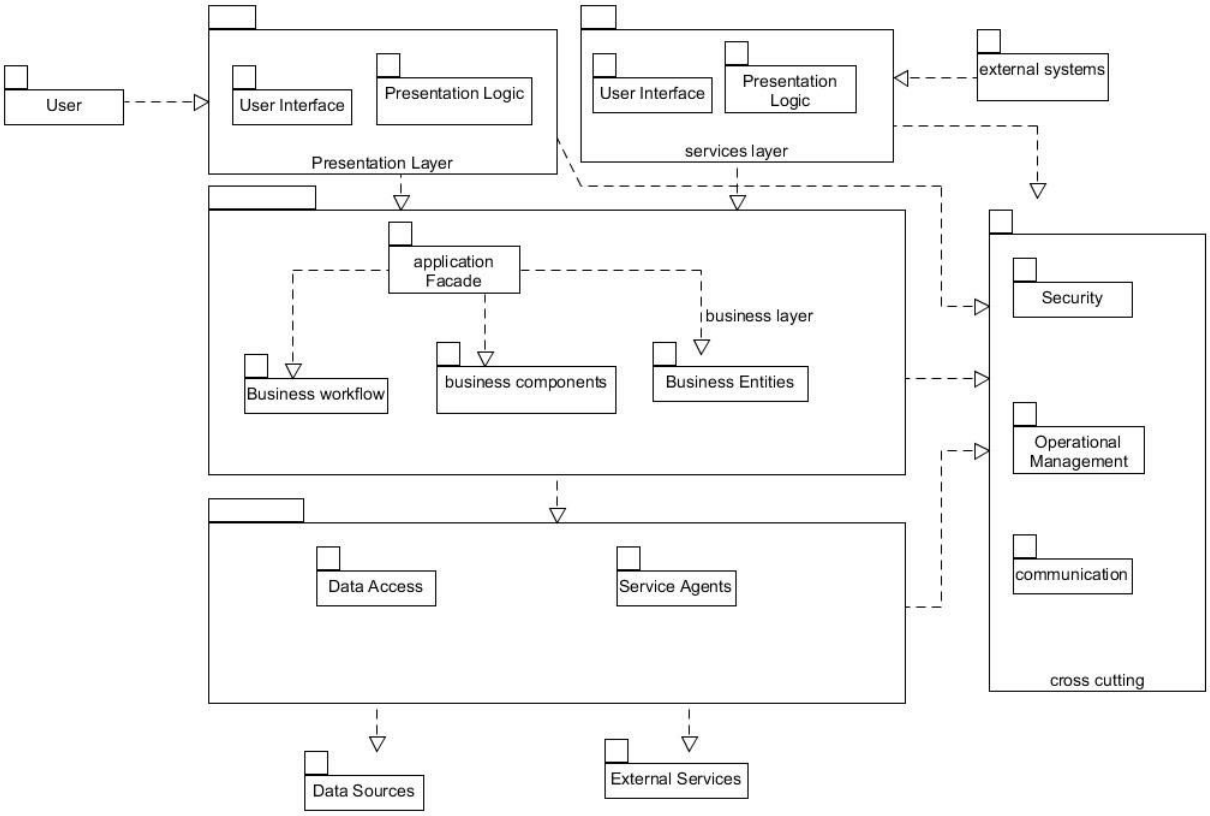
- In our context of online movie booking , during refactoring, unused design elements, inefficient or unnecessary algorithms, poorly constructed or inappropriate data structures, or any other design failure that can be corrected to yield a better design are uncovered so that an efficient system can be built.

- For example, an inefficient search algorithm for searching a product based on user request will be uncovered during refactoring and a programmer will replace it with an efficient algorithm with better run time.

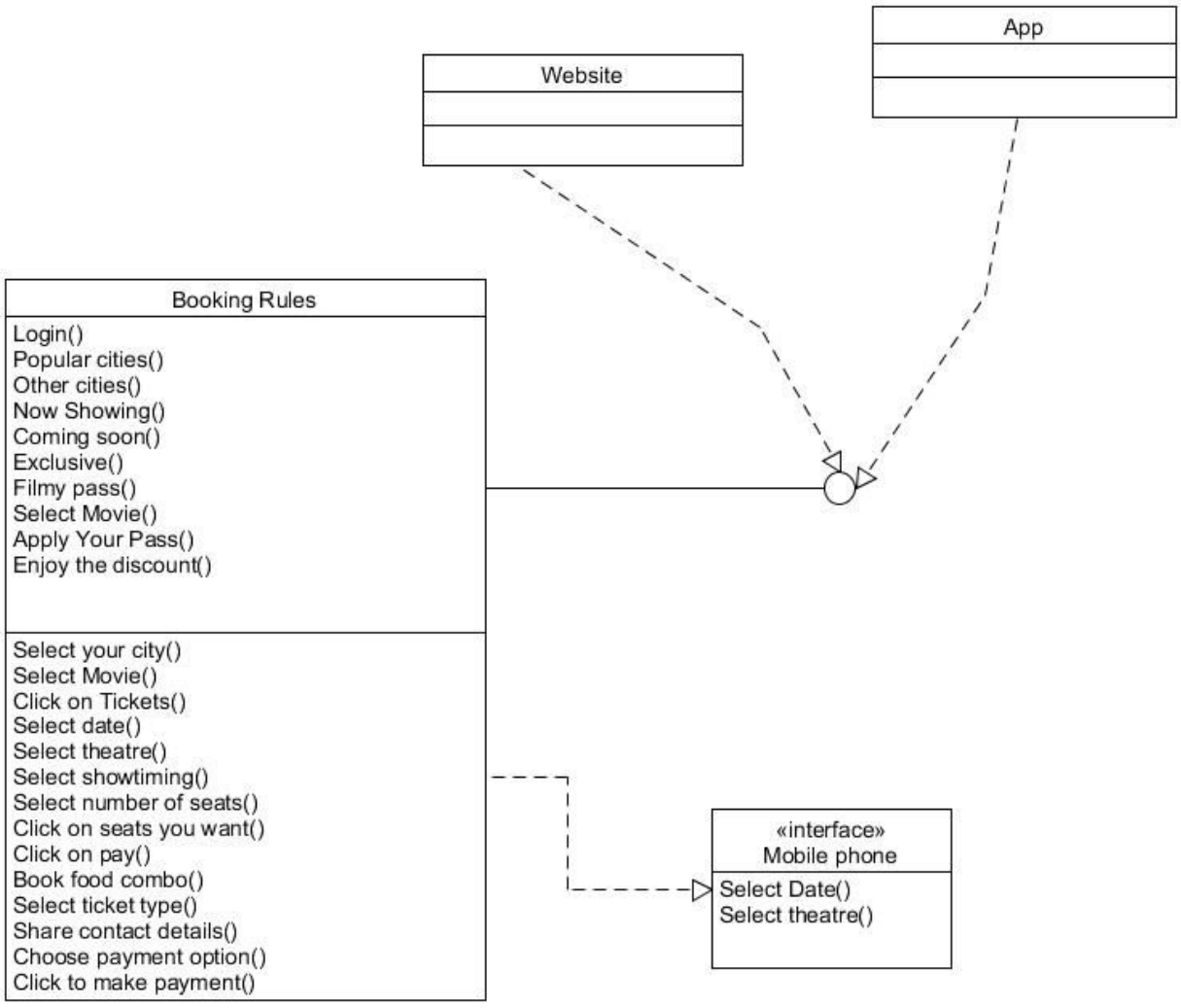
3.6.2 Data Design Elements

Data Structures Used: Array, List , Stack , Hash Table, Map
Data Dictionary: Refe section 2.5
Languages used: Java, Mysql
Packages and libraries used are l abstracted from the user based on OO approach.

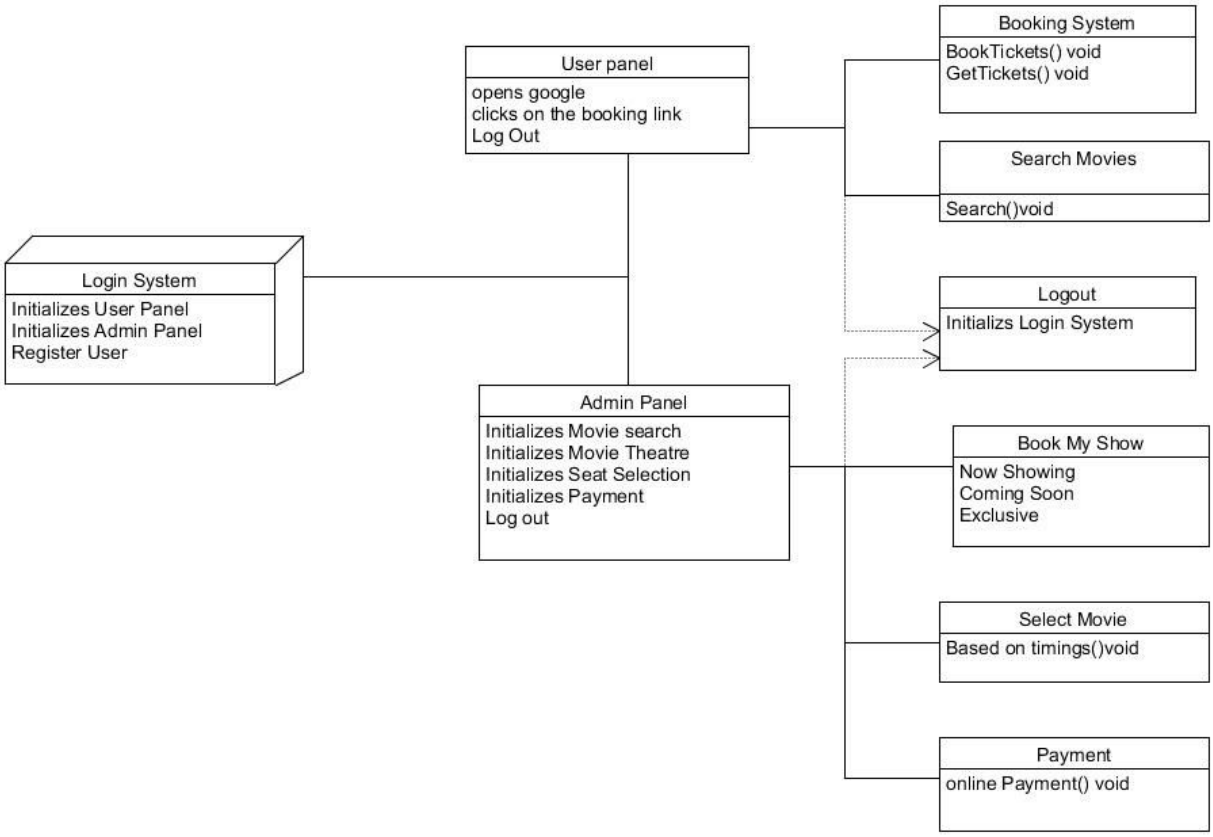
3.3 Architecture design elements:



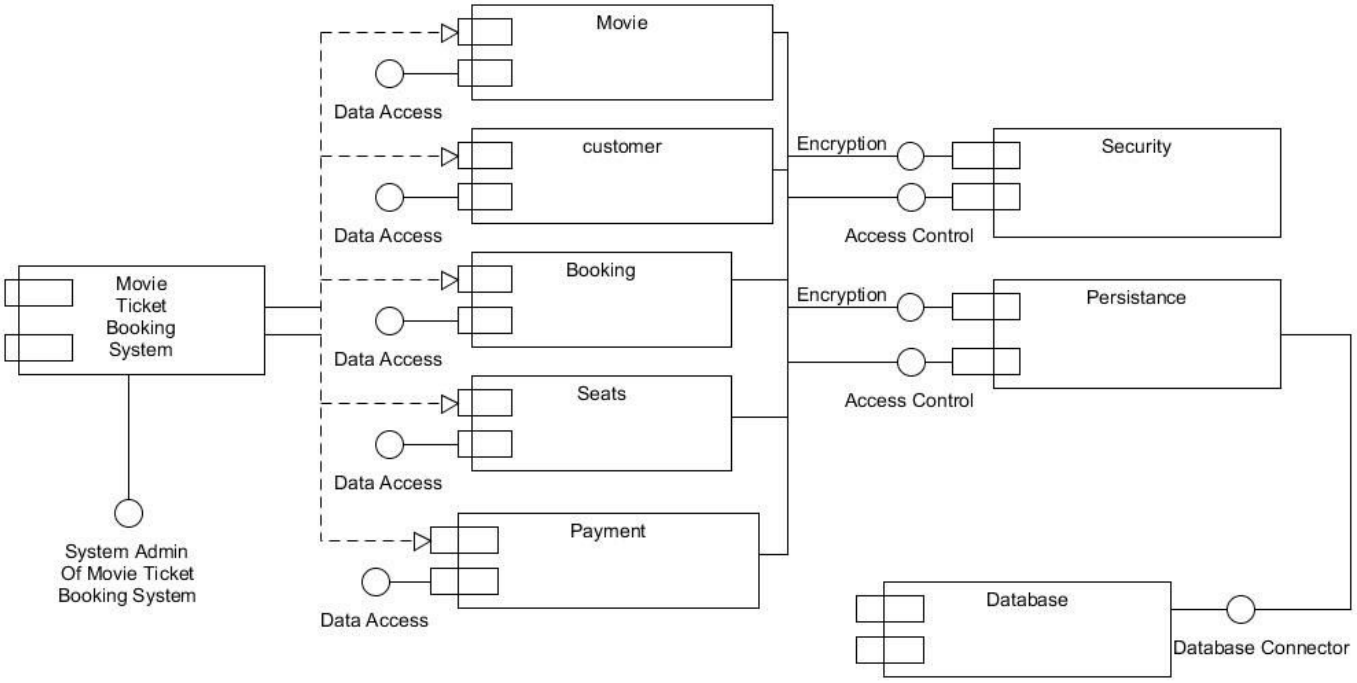
3.4 interface design element:



3.5 deployment design element



3.6 Component design elements



3.7 COCOMO model for Software Cost Estimation

Sub Model used: Basic COCOMO I
Model used: Organic
Formula used:

$$\text{Effort (Man/Month)} = 32 \times (\text{KLOC})^{1.05} \quad (1)$$

$$\text{Time} = 2.5 \times (\text{Effort})^{.38} \quad (2)$$

Calculating total LOC:
Module name Number of Lines of Code

Module name	**No of lines of code**
home	133
Movie details	151
about	76
contact	98
feedback	187
index	350
My account	139
payment	302
Reserve seats	146
search	488
login	106
logout	167
validation	100
sum	2443=2.443 KLOC

Estimating effort: effort=32 * (2.443) ^1.05=8.174 M

Estimating time:
time=2.5 *(8.174)^0.38=5.555 Months

SUB MODEL USED:
Intermediate COCOMO 1
Mode used:organic
Formula used:

$$\text{Effort (Man Month)} = \text{EFA} \times 32 \times (\text{KLOC})^{1.05}$$

$$\text{Time} = 2.5 \times (\text{Effort})^{.38}$$

Cost Drivers:
Product Attributes:

RELY – Required Software Reliability.
DATA – Database Size.
CPLX- Product Complexity.

Computer Attributes:

TIME- Execution Time.
STOR- Main Storage.
VIRT- Virtual Machine Volatility.
TURN- Computer Turn Around Time.

Personal Attributes
o ACAP-Analyst Capability.
o AEXP-Application Experience.
o PCAP- Programmer Capability.
o VEXP- Virtual Machine Experience.
o LEXP- Programming Language Experience.
Project Attributes
o MODP- Use of Modern Programming Practices.
o TOOL- Use of Software Tool.
o SCED- Required Development Schedule

	low	normal	high	Very high
RELY	0.90			

DATA				1.20
STOR				1.20
AXEP		1.0		
DCAP	0.80			
VIRT			1.40	
TURN		1.05		
COMPLEX				1.45
TIME	0.90			
SCED	0.65			
TOOL	0.70			
MODP		1.10		
LEXP		1.0		
VEXP			1.2	
ACAP		1.0		

Cost driver values:

EAF:product of these
0.90*1.20*1.70*1.0*0.80*1.20*1.0*1.45*0.90*0.6*0.70*1.2*1*1.10=1.521

THUS

EFFORT=1.5*32*(2.443)^1.05
effort= 12.15 MM
time=2.5*(12.15)^0.38
time= 6.45 months