COMP-SCI 5552A – Formal Software Specification Case Study

BookMyShow with Amazon EC2

Done By

Bharath Kumar Natesan Arumugam
Sibi Chakravarthy Ramesh

Introduction

Overview

BookMyShow is a largest entertainment-ticketing portal in India with 400 million average page views a month. It was launched in 2007 and owns 90% of the online entertainment ticketing market. The portal contains information about movies and events, show times, venue details, and cast bios. The company sells approximately six million tickets a month. A team of more than 100 engineers spread across three locations to monitor and upgrade the site. As the premier ticketing source for major events across 87 cities, the company's website often experiences record traffic during blockbuster movie releases or major sporting events. Customers eager to grab seats have little patience with a slow website. To make the user experience quick and easy, we suggest them to deploy their application in Amazon EC2 integrated with other Amazon Web Services.

Challenges

- Maintain performance during high profile events such as the release of blockbuster movies.
- Ensure zero downtime for the online ticket ordering application amidst increased demand.
- Provide a solution that could be rapidly deployed without disrupting service and with little or no change to BookMyShow's existing infrastructure and application.
- Servers are segmented into several zones, including servers for customer data and credit card information, database servers, and web application servers. In this complicated environment, maintaining security system rules manually was a challenging task.
- Using multiple open source tools, each with its own management console, for monitoring files and logs.

- Combatted low-profile hacks with a variety of tools and techniques, but found it a challenge to cope with bigger, more sophisticated attacks.

Explanation of the underlying system and how it would work

BookMyShow is exploring Platform as a Service options that would allow them to scale up for weeks or days when popular movie launches or sporting events cause a spike in website traffic. Tight integration with leading cloud service providers such as Amazon Web Services (AWS) make deployment fast and easy. It is a web-based tool designed to optimize live event ticket pricing, improve yield management, and generate incremental revenue.

The user interacts with the website like any other sites but the unique thing about our application is, it is deployed in AWS EC2 (Instance). By Deploying in AWS our application ensures high availability, performance and scalability and reduces bottleneck of the website when hosted in normal server. AWS increase User interaction to our website by providing flexible specifications and pricing. The basic working of our application is when user logins into the system, the Amazon EC2 instance is created. The details of the user will be stored in Elastic Block Store (EBS). After logging into the system, the user will be able to view the list of movies, date, time, and venue of the event. User can select the movie he/she wishes to see and they also have an option to select number of seats to be booked. After filling all the required fields to be booked, now user can proceed to payment option.

Features of the system

- The system ensures High Availability, since it is deployed in AWS EC2.
- Each User will have their own computing platform based on their own preference.
- User can create many instances to book multiple tickets in Amazon Cloud.
- Each instance is independent of one another and can be configured based on users requirements.
- User will have an option to choose movies that they wish to watch later, they will receive notifications based of the movie being run on the theater.
- Eligible Customers will receive offers/exclusives to increase interactions.

Evaluation of the technology

Reliable

Using Amazon EC2 which has registered the highest reliability over the past year with total downtime of just 2.41 hours across all regions. Since customer book tickets at any time, it is necessary to have sufficient servers to respond to all requests. When servers go down, we rely on Amazon EC2 instances which act as some virtual servers so that no request is denied due to server breakdowns. Amazon Web Services has provided many plans per various application usage. So BookMyShow can make use of any of the plans which is suitable for them by allocating sufficient RAMs and memory. Instances are created when the customer login and instances are deleted when the customer finishes booking process. Since AWS can able to allocate many instances at a time, it is enough for BookMyShow to handle requests.

Flexible

- Available in different locations.
- Extensive list of supported operating systems & software.
- Can easily integrate with other AWS services such as Amazon Private Cloud, Amazon Elastic Block Store, Amazon Cloud Watch, etc.
- Provide many purchasing options for cost optimization such as On-demand, Reserved and Spot cost policies.
- We can Import and Export virtual machines.
- Choice of instance families with differing resource ratios.

Available – using Amazon Elastic Block Store and Amazon S3

- EBS volumes are automatically replicated within the Availability Zone (AZ) in which they are created so that there is no single point of failure and data loss.

- Can be shared across AWS accounts or copied across AWS regions which makes data available always for the employees across different locations.
- Data stored at rest on the volume, disk I/O, and snapshots created from the volume are all encrypted.

Scalable – using Auto Scaling and Load Balancer

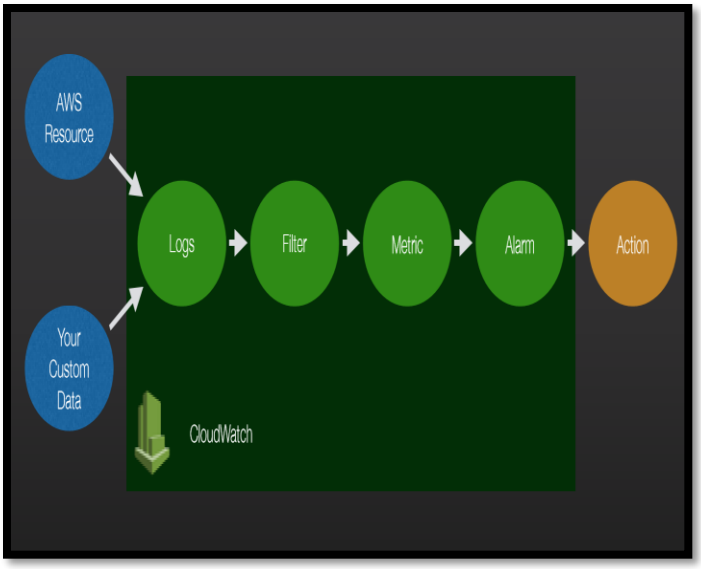
- Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.
- Auto Scaling automatically increase the number of Amazon EC2 instances during release of blockbuster movies to maintain performance.
- Elastic Load Balancing helps to distribute traffic to your instances within Auto Scaling groups.

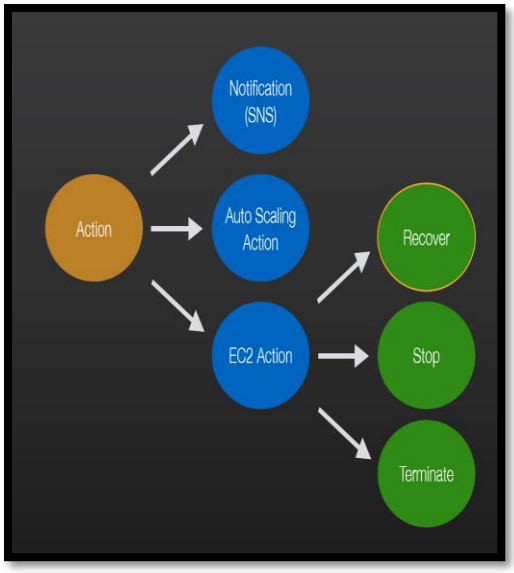
Secure – using Amazon Virtual Private Cloud

- Computed instances are located in a VPC within a given IP range.
- Employees can decide which instances are exposed to the Internet and which remain private. Here the payment options and security rules remain private and user details are exposed to public.
- Access key and secret key can be used to authenticate when accessing AWS APIs.

Maintainable – using Amazon Cloud Watch

It is not possible for employees to monitor all activities, so it is necessary to deploy and automated monitoring service. Using Amazon Cloud Watch, we can able to implement this feature. They collect logs and filter it. Based on filtered content the actions are performed which are predefined actions chosen by employees.



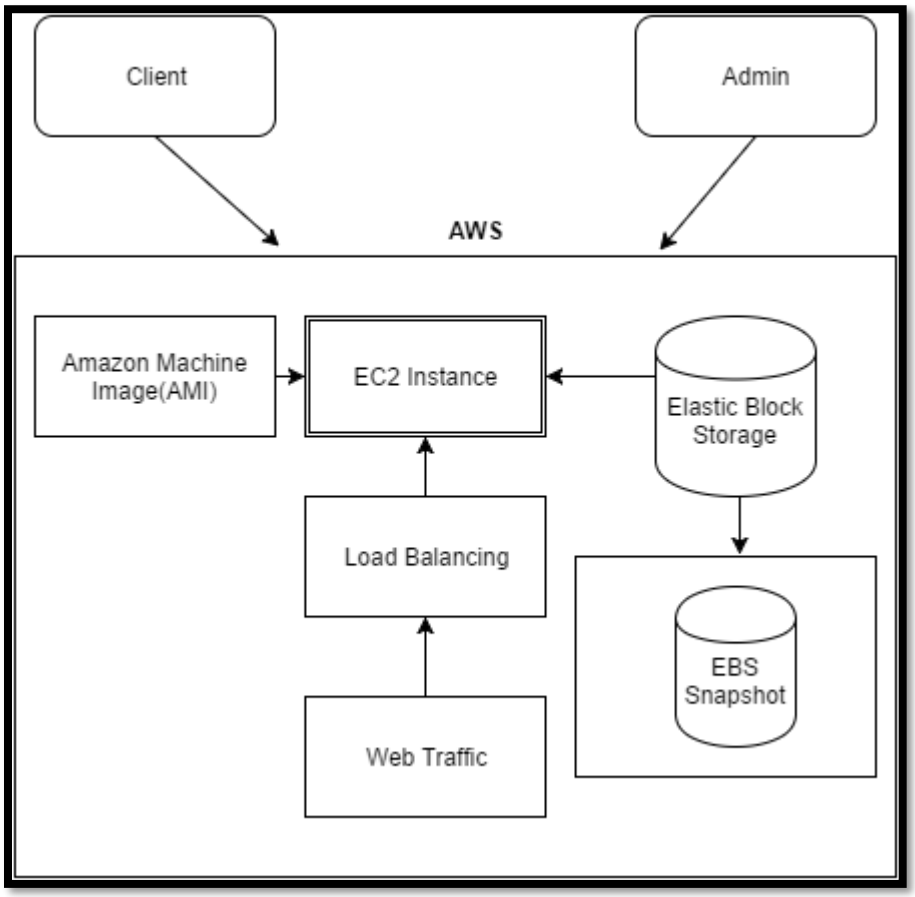


Cost

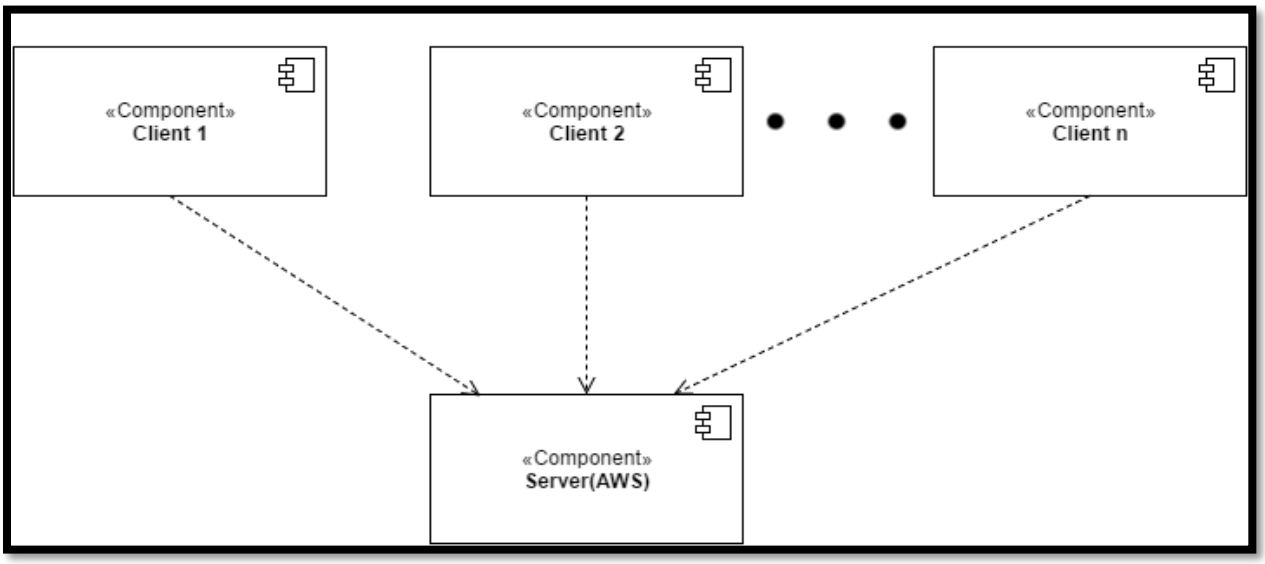
We suggest to use Reserved Instances cost policy which provides significant discount (up to 75%) compared to other pricing policies. The instances are assigned to a specific Availability Zone and provided a capacity reservation. Since we can't predict the start and end time of the use of application, this cost policy gives us confidence that instances and capacity are always available to handle user requests. This can provide significant savings compared to the current computing costs.

Formal Specifications

Process Architecture Diagram



Component Architecture Diagram



Data Flow Diagram

Entities:

* Users
* Admin

Process:

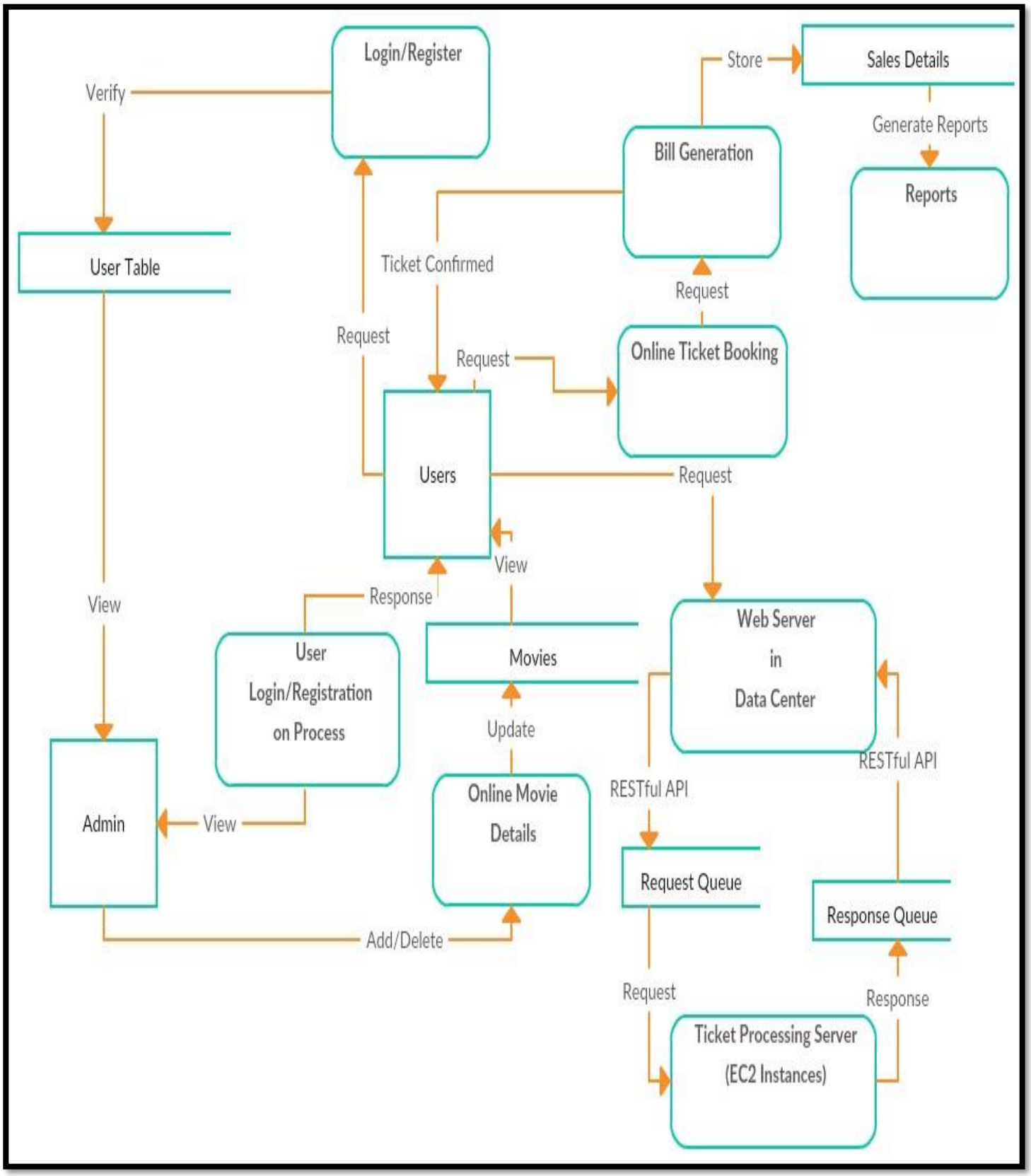
* Login/Register
* User Login/Registration in Process
* Online Movie Details
* Online Ticket Booking
* Bill Generation
* Web Server in Data Center
* Ticket Processing Server (EC2 Instance)
* Reports

Data Stores

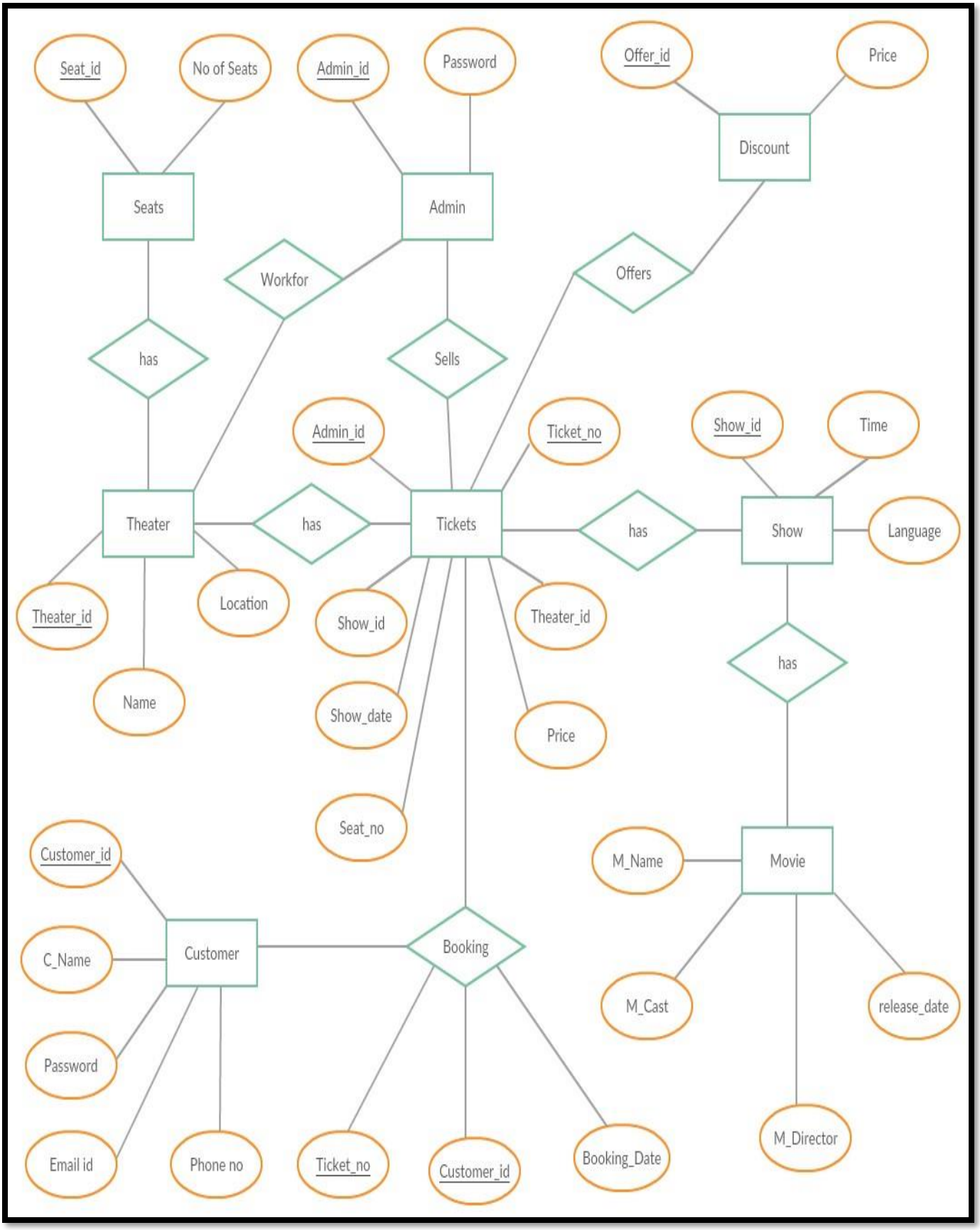
* User Table
* Sales Details
* Movies
* Request Queue
* Response Queue

Data Flows

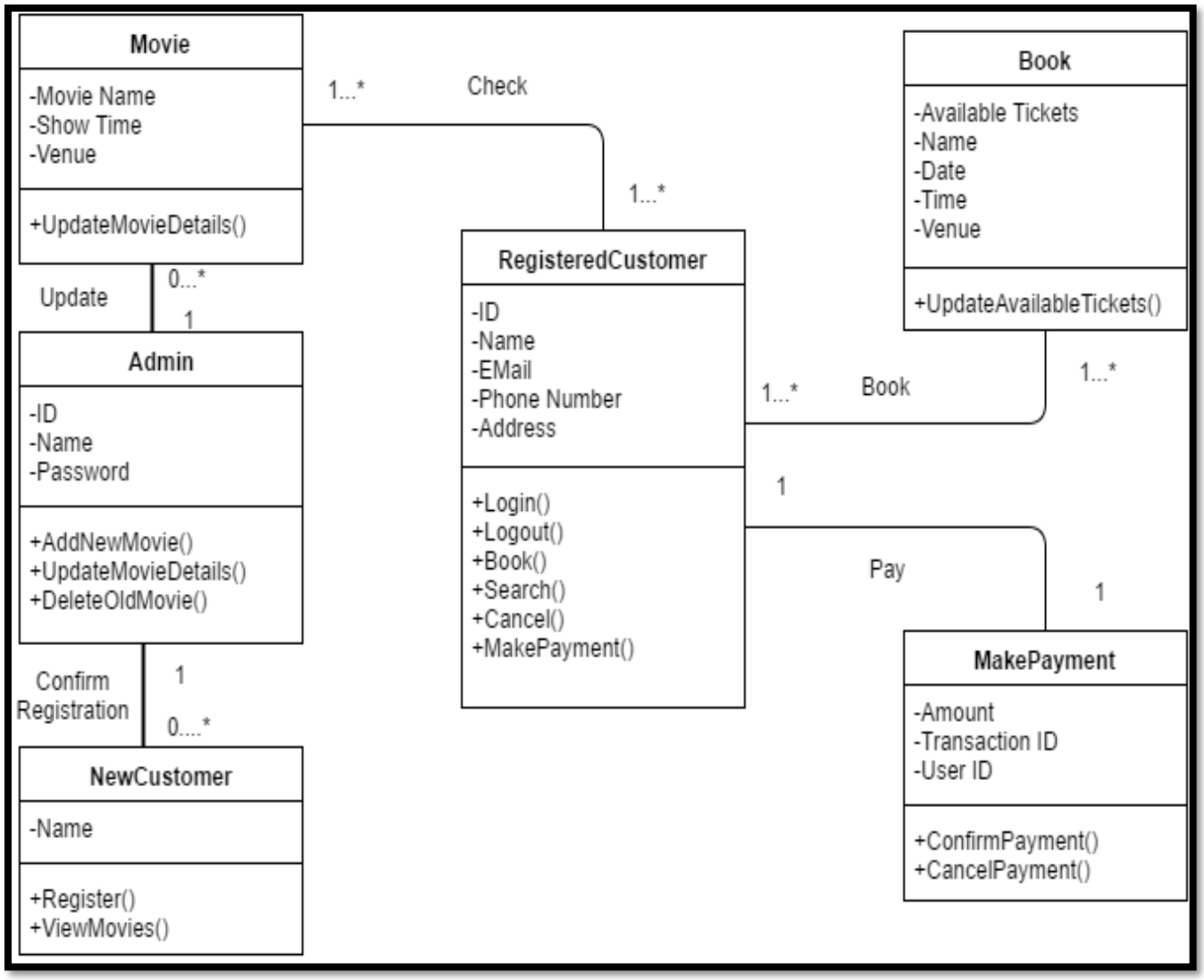
* Verify
* View
* Request
* Response
* Ticket Confirmed
* Update
* Add/Delete
* Store
* Generate Reports
* RESTful API


Login/Register
Verify
Store
Sales Details
Generate Reports
Bill Generation
Reports
User Table
Ticket Confirmed
Request
Request
Request
Online Ticket Booking
Users
Request
View
Response
View
Web Server
in
Data Center
User
Login/Registration
on Process
Movies
Update
Admin
View
Online Movie
Details
RESTful API
RESTful API
Request Queue
Response Queue
Add/Delete
Request
Response
Ticket Processing Server
(EC2 Instances)

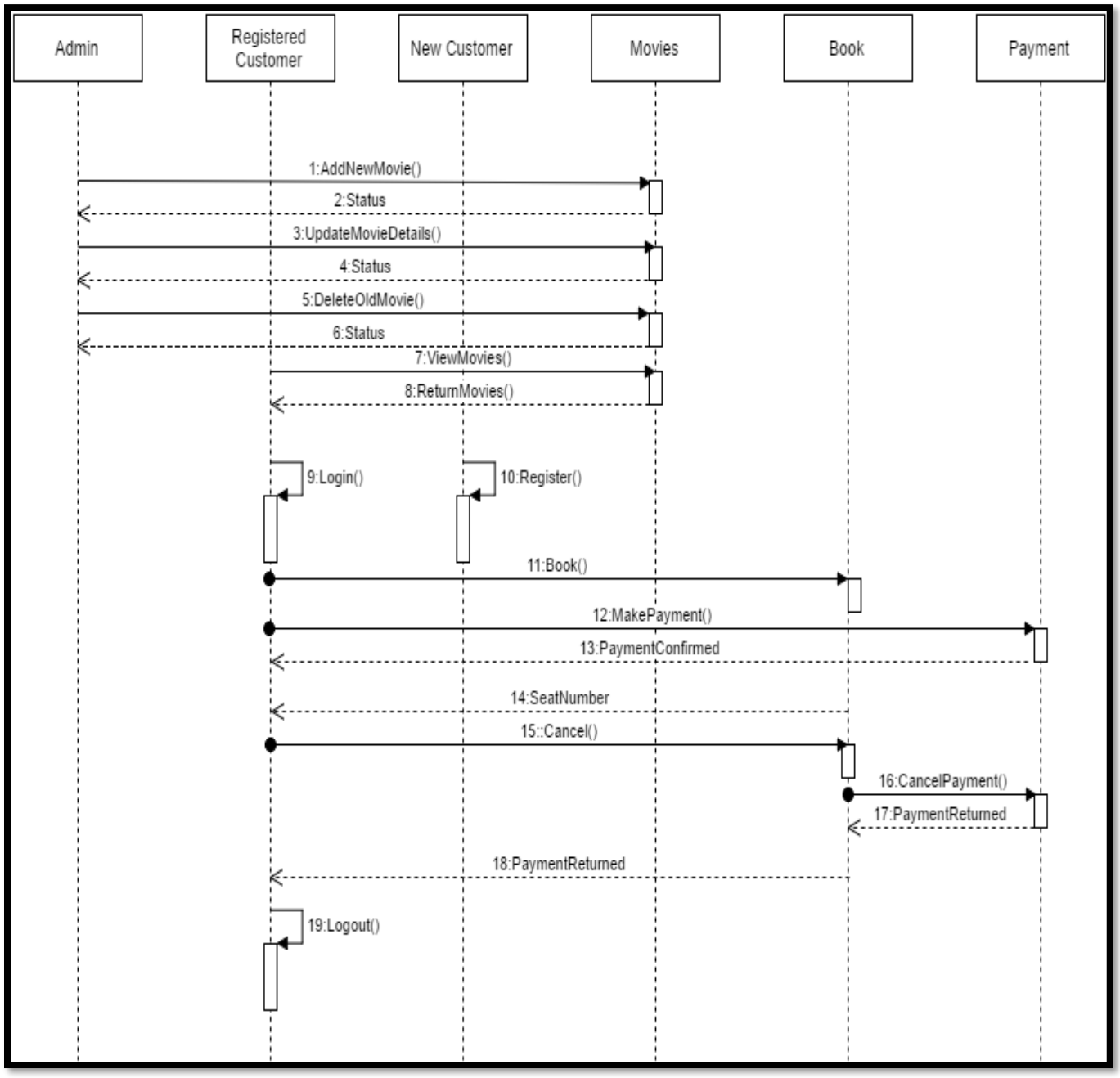
ER Diagram


Seat_id
No of Seats
Admin_id
Password
Offer_id
Price
Discount
Seats
Admin
Workfor
Offers
has
Sells
Admin_id
Ticket_no
Show_id
Time
Theater
has
Tickets
has
Show
Language
Theater_id
Location
Show_id
Theater_id
has
Name
Show_date
Price
Seat_no
Customer_id
M_Name
Movie
C_Name
Customer
Booking
M_Cast
release_date
Password
M_Director
Email id
Phone no
Ticket_no
Customer_id
Booking_Date

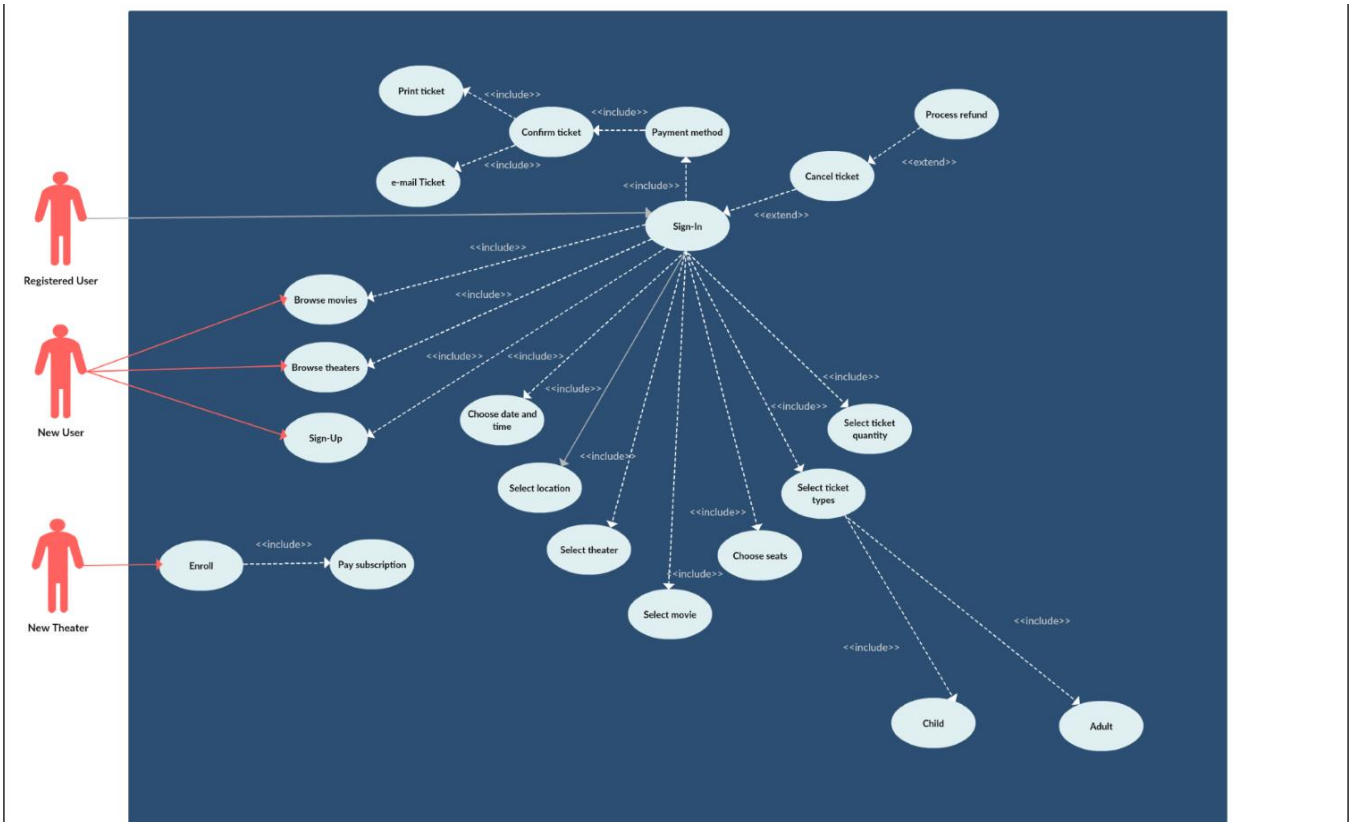
UML Class Diagram

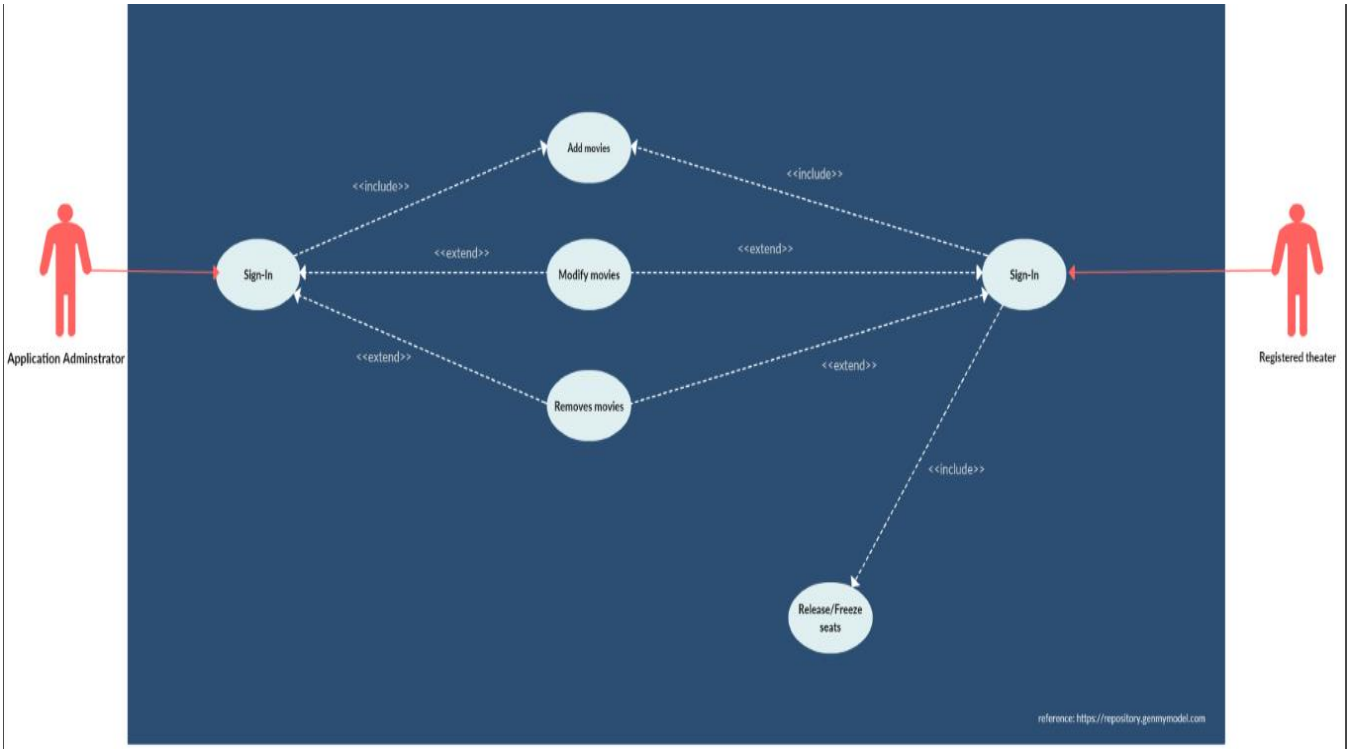


UML Sequence Diagram

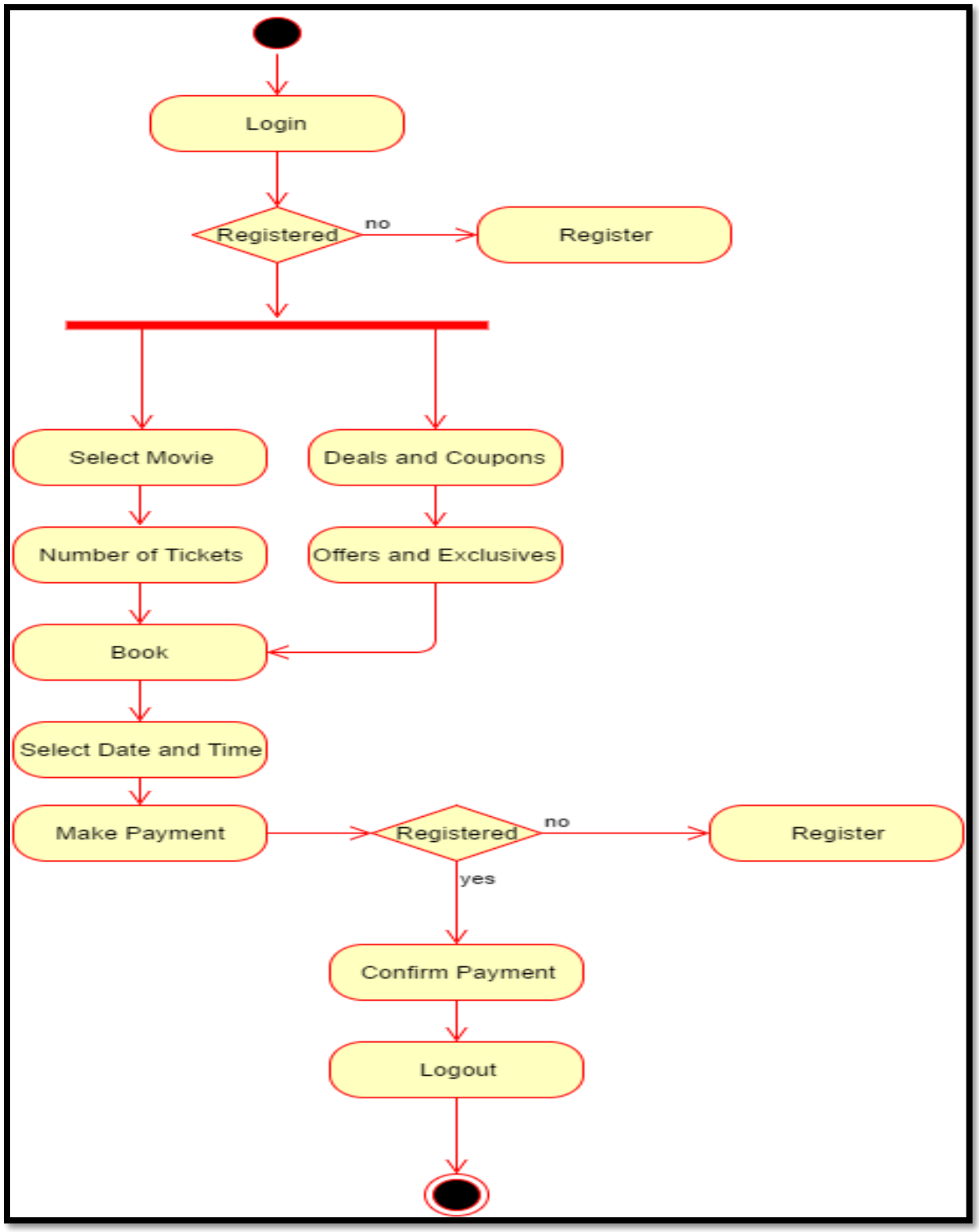


UML Use Case Diagram

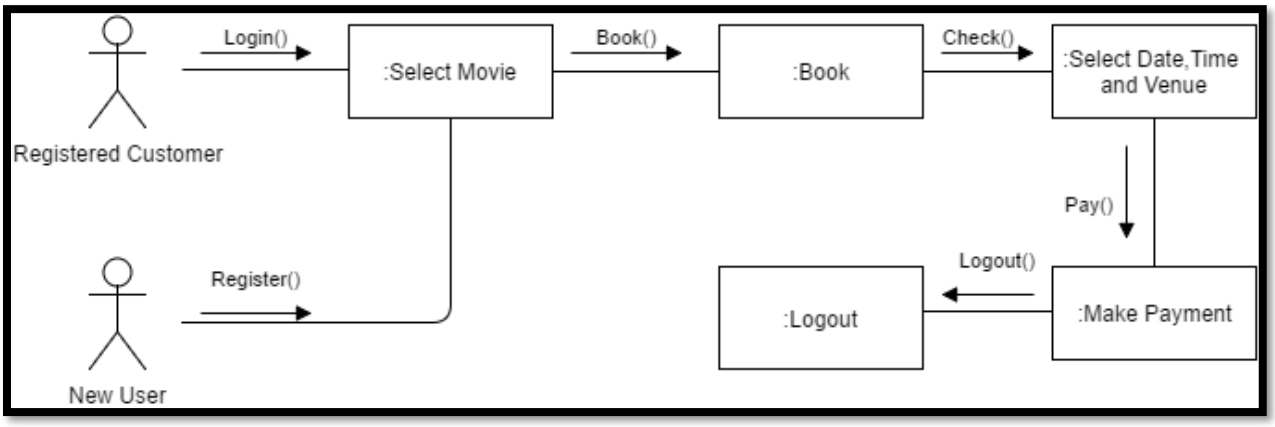




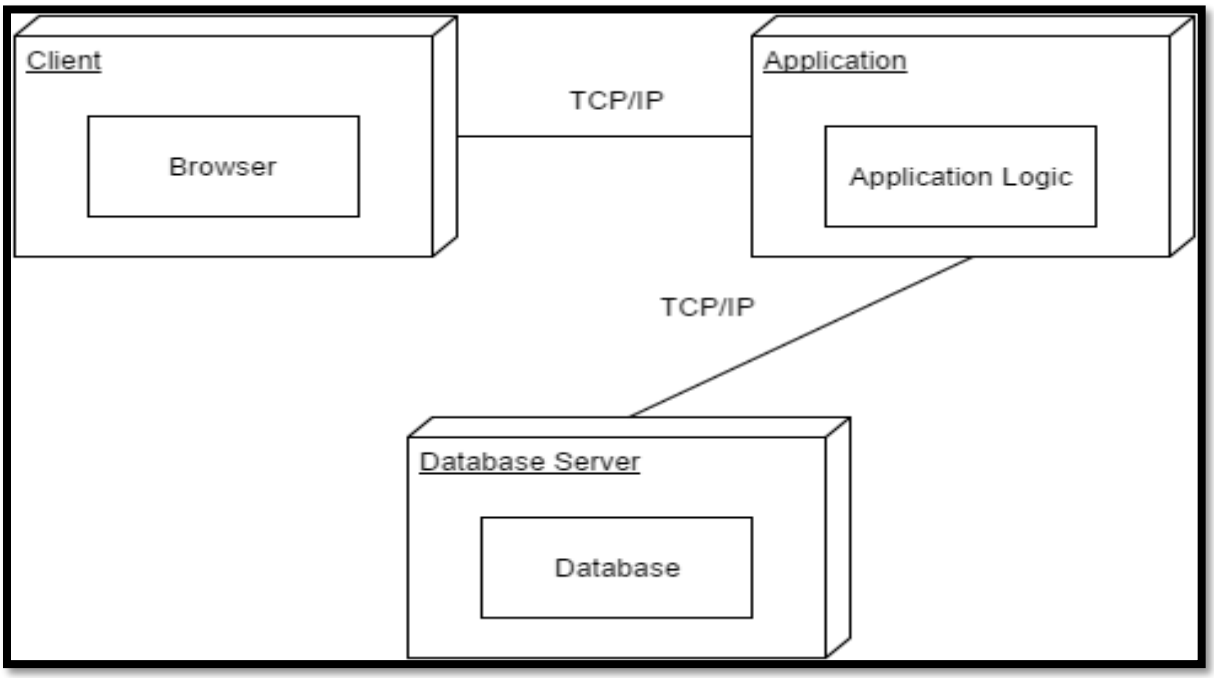
UML Activity Diagram



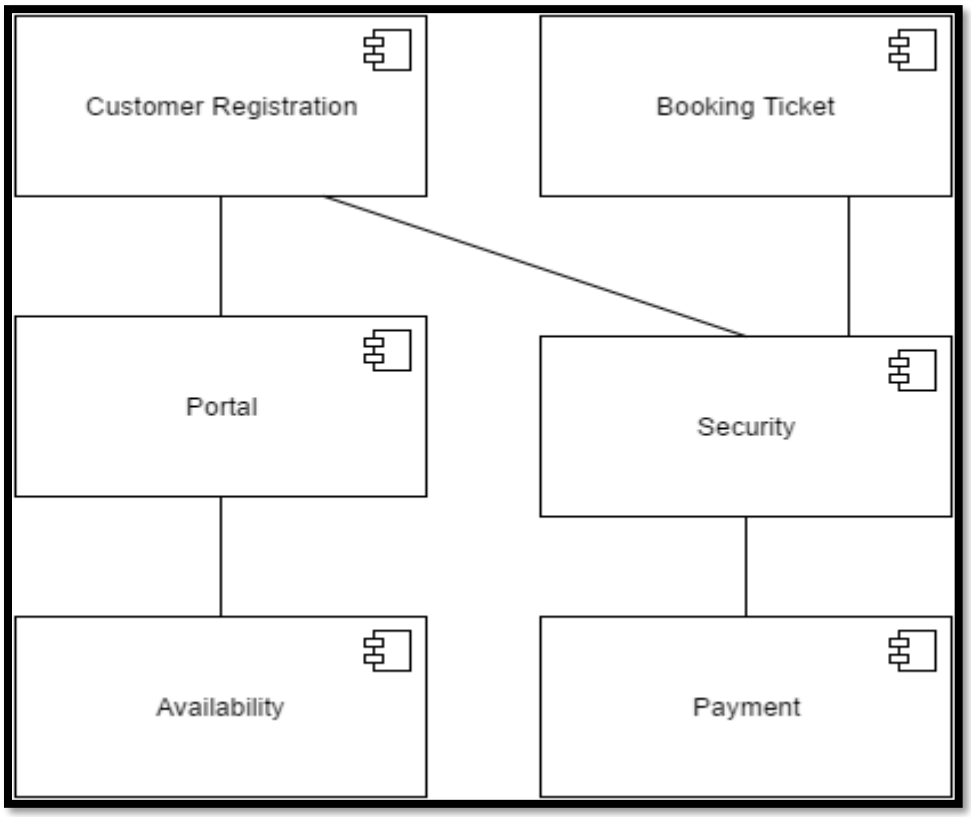
UML Collaboration Diagram



Deployment Diagram



Component Diagram



Determination

We strongly recommend to use technology AWS EC2 for this project because of its flexible options to perform consistently without any interruptions that may cause the entire system to fail. It can adapt itself based on the current environment by using auto scaling feature inbuilt in each instance.

It satisfies all the necessary requirements that is needed for the system to perform efficiently and it overcomes the challenges faced by the current technology which is used. We have included additional features which can reduce computing cost and increase performance.

AWS EC2 acts as a complete computing platform to support various operations. AWS EC2 is the best option for security, load balancing, cost effective and performance when compared to its competitors.

Appendix

Name	Work Completed
Bharath Kumar Natesan Arumugam	Data Flow Diagram, Use Case Diagram, ER Diagram, Evaluation of Technology, Challenges, and Explanation of Underlying System
Sibi Chakravarthy Ramesh	Architecture Diagram, UML Diagrams (Class, Sequence, Activity, Deployment, and Component), and Features of System
Both	Overview, and Determination of using Technology