E-Ticketing

Exp1:- Write down the problem statement for a suggested system of relevance.

Description:

E-ticketing software is a platform that allows organizations to manage support requests via email or phone and streamline their ticketing processes. It is an online-based platform that enables employees and customers to provide accurate information about problems, complaints, or returns. E-ticketing software is easy to use, intuitive, easy to implement, and requires minimal maintenance. It can be used for various events, including sports games, music concerts, theatre, and family events.

Software Description:

E-ticketing software is used to promote events and sell tickets for events. It receives, logs, and sorts incoming tickets to streamline problem resolution. Simple ticketing software is the price of entry for any professional customer service team, even if it's a team of one.

Here are some features that a good e-ticketing software should have:

- Personalized ticket management: The software should allow for personalized ticket management, including the ability to customize fields and create automatic knowledge base article suggestions relevant to a customer's issue.
- Mobile check-in app: The software should have a mobile check-in app that allows for easy admission to events.
- Event management: The software should have features for event management, including print-at-home ticketing, assigned seating, and mailing lists.

Problem in existing system:

- Training personnel: One of the challenges associated with e-ticketing is the need to train personnel on novel software and hardware. This can be time-consuming and costly for organizations.
- Integration with existing systems: E-ticketing systems need to be fully integrated into the brand and should look, act, and feel the same as the rest of the online presence. This can be challenging if the system does not integrate well with existing systems.

Requirements:

- Security and privacy: The software should be secure to protect customer data and prevent fraud.

- Integration capabilities: E-ticketing software should be able to integrate with existing systems to avoid compatibility issues.
- Customer self-service: The software should allow customers to manage their own tickets, including the ability to view and print tickets, transfer tickets, and request refunds.
- Reporting and analytics: The software should have basic reporting and pre-built dashboards to help organizations track ticket sales, customer satisfaction, and agent productivity.
- Easy implementation: The software should be web-based and easy to implement, requiring minimal maintenance and support.

Cost Analysis:

- Some e-ticketing software is available for free, while others can cost hundreds or even thousands of dollars per month.
- The cost of e-ticketing software can be offset by the benefits it provides, such as reduced operating costs, improved business performance, and increased revenue.
- E-ticketing software can help organizations save money by reducing the need for manual ticketing processes and paper tickets.
- E-ticketing software can also help organizations increase revenue by providing a seamless and positive customer experience, which can lead to repeat business and positive reviews.
- The cost of e-ticketing software should be weighed against the potential benefits to determine if it is a worthwhile investment for the organization.

Limitation:

- Technical difficulties: Any problems that arise in e-ticketing systems need to be taken care of swiftly by developers. Technical difficulties can cause delays and frustration for customers.
- Cost: E-ticketing software can be costly, depending on the features and functionality of the software. This can be a limitation for smaller organizations with limited budgets.
- Security concerns: E-ticketing systems need to be secure to protect customer data and prevent fraud. Any security breaches can lead to loss of customer trust and financial loss for the organization.

Future Scope:

The future scope of e-ticketing software includes increased adoption, expansion into new industries, improved functionality, increased security, and integration with emerging technologies.

- Increased adoption: The e-ticketing market is expected to grow in the coming years, with more organizations adopting e-ticketing software to streamline their ticketing processes.
- Expansion into new industries: E-ticketing software is currently used in industries such as entertainment and transportation. However, there is potential for e-ticketing software to expand into new industries, such as construction.

- Improved functionality: E-ticketing software is likely to continue to improve in functionality, with new features and capabilities being added to meet the evolving needs of organizations and customers
- Increased security: As e-commerce fraud continues to rise, e-ticketing software is likely to place an increased emphasis on security and privacy to protect customer data and prevent fraud
- Integration with emerging technologies: E-ticketing software is likely to integrate with emerging technologies, such as blockchain and artificial intelligence, to improve functionality and provide a better customer experience.

These are just a few examples of the potential future scopes of e-ticketing software. As technology continues to evolve, e-ticketing software is likely to evolve as well, providing new opportunities for organizations to streamline their ticketing processes and provide a seamless and positive customer experience.

EXP2 DO REQUIREMENT ANALYSIS AND DEVELOP SOFTWARE REQUIREMENT SPECIFICATION SHEET (SRS) FOR THE SUGGESTED SYSTEM.

1) Introduction

1.1) Purpose

The E-Ticketing Software Requirements Specification (SRS) is a detailed description of the software system to be developed for managing e-ticketing for movies. This SRS outlines both functional and non-functional requirements, as well as potential use cases.

The primary purposes of this e-ticketing software include:

A) Facilitating Ticket Booking: Providing a platform for moviegoers to browse movie listings, select showtimes, and purchase tickets online.

B) Streamlining Theatre Operations: Assisting theatre administrators in managing seat allocations, show schedules, and ticket sales.

1.2) Scope

The scope of the E-Ticketing Software for Movie Theatres includes:

A) Movie Listings: Displaying current movies, showtimes, and available theatres.

B) User Registration: Allowing users to create accounts and manage their profiles.

C) Ticket Booking: Enabling users to select seats, purchase tickets, and receive e-tickets.

D) Payment Processing: Handling secure online payments for ticket purchases.

E) Admin Panel: Providing theatre administrators with tools to manage movie listings, show schedules, and seat allocations.

1.3) Definitions, Acronyms, Abbreviations

In this SRS for e-ticketing software, the following terms are used:

- E-Ticketing: Electronic ticketing, the process of booking and receiving tickets digitally.
- GUI: Graphical User Interface.
- DBMS: Database Management System.
- SQL: Structured Query Language, used for database operations.
- API: Application Programming Interface, for integrating with payment gateways and movie databases.

1.4) Overview

Functional Overview:

- Movie Listings: Displaying movie titles, showtimes, theatre locations, and available seats.
- User Registration and Login: Allowing users to create accounts, log in, and manage their profiles.
- Ticket Booking: Providing a user-friendly interface for selecting seats, purchasing tickets, and receiving e-tickets.
- Payment Integration: Securely processing payments using various payment gateways.
- Admin Panel: Offering theatre administrators the ability to update movie listings, manage show schedules, and allocate seats.

User Experience:

- User-Friendly Interface: Ensuring an intuitive and visually appealing interface for both moviegoers and administrators.
- Secure Transactions: Guaranteeing the security of user data and payment information.
- Real-time Updates: Providing real-time information on seat availability and showtimes.

2) Productive Prospective

- Streamlined Booking Process: Simplifying the ticket booking process for moviegoers, reducing queues and wait times.
- Revenue Generation: Increasing theatre revenue through online ticket sales.
- Data Insights: Collecting valuable data on user preferences and movie popularity for future marketing and scheduling decisions.

2.1) System Interface

- API Integration: Integration with movie databases and payment gateways through APIs.
- Database: Use of a relational database system to store user information, movie listings, and booking details.

2.1.2) User Interfaces

- Customer Interface: The user interface for moviegoers to browse movies, select seats, and make bookings.
- Admin Interface: The interface for theatre administrators to manage movie listings, showtimes, and seats.

2.1.3) Hardware Interfaces

- Server Hardware: The software will run on theatre servers.
- Client Hardware: Users will access the software via web browsers on their devices.

2.1.4) Communication Interfaces

- The software will communicate with external payment gateways over secure HTTPS connections.
- User interactions with the software will be facilitated through standard web protocols.

2.1.5) Memory Constraints

- The software should be optimized for efficient memory usage, especially on mobile devices.

2.1.6) Operation

- The software will operate on standard web browsers, ensuring compatibility with popular browsers such as Google Chrome, Mozilla Firefox, and Safari.

2.1.7) Site Adaptation Requirements

- The software should be compatible with various web browsers and responsive on different screen sizes and devices.

2.2) Product Functions

The e-ticketing software will perform the following functions:

- Display Movie Listings
- User Registration and Login
- Seat Selection and Booking
- Payment Processing
- Admin Panel for Theatre Management

2.3) User Characteristics

- Moviegoers: Users may range from tech-savvy individuals to those with limited technical proficiency.
- Theatre Administrators: Admin users should have access to advanced features for efficient theatre management.

2.4) Constraints

- Performance: The software must provide fast and responsive ticket booking experiences, even during high-demand periods.
- Integration: Integration challenges may arise when connecting with third-party movie databases and payment gateways.
- Scalability: The software should be able to handle increasing numbers of concurrent users and theatres.

2.5) Assumptions and Dependencies

a) Stable Internet Connection: Users are assumed to have a stable internet connection to access the e-ticketing platform.

b) User Proficiency: Users are expected to have basic computer literacy for navigating the software.

c) Third-Party Integration: The software depends on APIs provided by movie databases and payment gateways for real-time data and payment processing.

3) Specific Requirements

3.1) External Interfaces

- Integration with Movie Database API for fetching movie details and showtimes.
- Integration with Payment Gateway APIs for processing payments.

3.2) Functional Requirements

a) User Management:
   - Registration of new users.
   - User login and authentication.
   - Profile management.

b) Movie Booking:
   - Display of available movies, showtimes, and theatres.
   - Seat selection and booking. o Generation and delivery of e-tickets.

3.3) Performance Requirements

a) Response Time:
   - Quick display of movie listings and seat availability.
   - Smooth and fast payment processing.

b) Load Handling:
   - Ability to handle multiple concurrent users during peak booking times.

3.4) Logical Database Requirements

a) Data Model:
   - Define the data structure for user profiles, bookings, theatres, and movies.
   - Establish relationships between data entities.

b) Data Integrity:
   - Enforce data integrity constraints to maintain accurate and consistent data.

3.5) Design Constraints

- The software will use a MySQL database.
- It will be a web-based application accessible via standard web browsers.

3.6) Software System Attributes

a) Reliability:
   - The software should operate without failure, ensuring ticket bookings and payments are secure.

- Critical for tasks like data storage, transaction handling, and user management.

b) Usability:

- The user interface should be intuitive and user-friendly for both moviegoers and administrators.

3.7) Other Requirements

- Data Protection: Sensitive user data and payment information should be encrypted and secure.
- Authentication and Authorization: Robust authentication mechanisms should be in place, and users should only have access to authorized functions.
- Third-Party Software: The software may need to integrate with third-party services for additional features.

EXP 4:- TO DRAW ER DIAGRAMS FOR E TICKETING MANAGEMENT SYSTEM

ER Diagrams are composed of entities, relationships, attributes, and cardinality. Entities are things of interest in a database, relationships are connections between entities, attributes are descriptors of entities, and cardinality indicates how many entities can be related to each other. ER Diagrams also depict entity categories, entity keys, recursive relationships, attribute categories, and descriptive attributes.

ER Diagram Components and Features

Entities: Things of interest in a database, such as customers, products, or orders.

Relationships: Connections between entities, such as a customer ordering a product.

Attributes: Descriptors of entities, such as a customer's name or a product's price.

Cardinality: Indicates how many entities can be related to each other.

ER Diagram Features

Entity categories: Strong, weak, and associative entities.

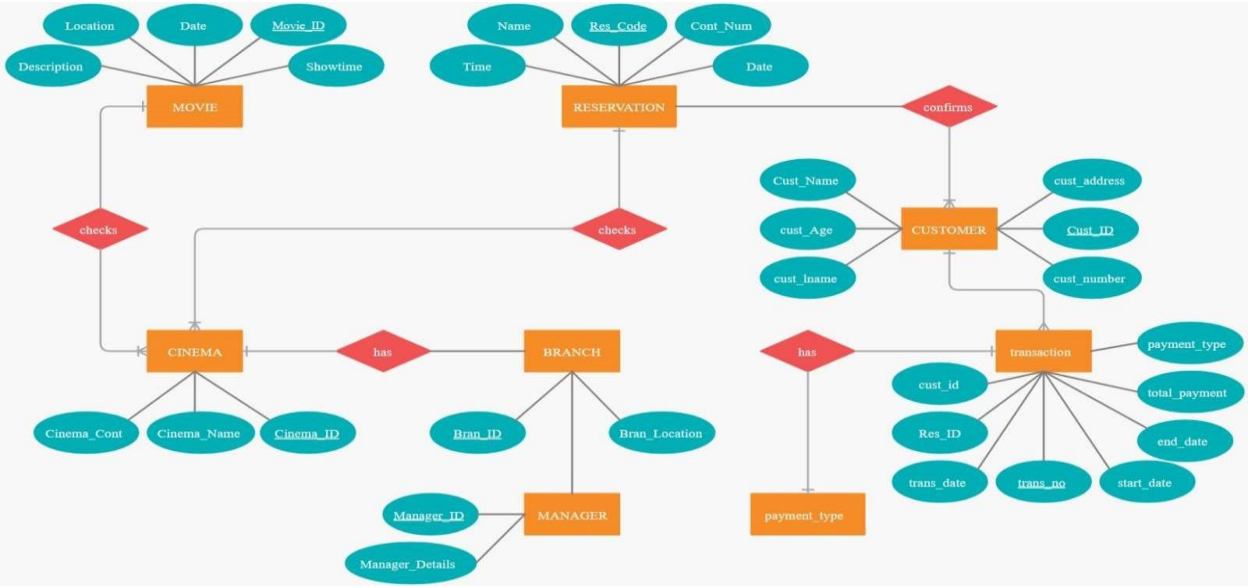
Entity keys: Super, candidate, and primary keys.

Recursive relationships: The same entity participates more than once in the relationship.

Attribute categories: Simple, composite, derived, single-value, and multivalue attributes.

Descriptive attributes: Properties of a relationship, not of an entity

ERD FOR THE E-TICKETING SYSTEM



The above ER diagram illustrates the key information about the E-ticketing system, including entities like Movie, Cinema, Branch, Manager, Reservation, Customer, transaction, payment_type. This diagram also shows the relationships between entities.

Entities and their Attributes –

- Movie Entity: It has Description, Location, Date, Movie_id, Showtime. Movie_id is the primary key for the Movie entity.
- Cinema Entity: It has Cinema_Cout, Cinema_Name, Cinema_Id in which the Cinema_Id is the primary key.
- Reservation Entity: It has Name, Time, Date, Res_code, Cout_Num in which the Res_code is the primary key.
- Branch Entity: It has the Bran_Id and Bran_Location
- Manager Entity: It has the Manager_ID and Manager_details with thre Manager_id as the primary key.
- Customer Entity: It has cust_name, cust_age, cust_lname,cust_address, cust_id, cust_number . Name is composite attribute of firstname and lastname and cust_id is the primary key.
- Transaction Entity : It has the cust_id, Res_id, trans_date, trans_no, start_date, end_date, total_payment, payment_type as the attributes.

Relationships between Entities –

- Cinema has multiple movies but one movie can be run in one cinema so the relationship is 1:N
- Cinema has a single branch and each branch is related to one cinema, so its relation is 1:1
- Branch has a single manager in it so its relation is 1:1
- Multiple customer give the multiple reservation so its relationship is M:N
- Each transaction has multiple payment type so its relationship is 1:M

Exp 3:- To perform the function oriented diagram: Data Flow Diagram (DFD) and Structured chart.

Data Flow Diagrams (DFD) are used to represent the sequence of process steps and flow of information using a graphical representation or visual representation rather than a textual description. DFDs can range from simple overviews to complex, granular representations of a system or process with multiple levels, starting with level 0. The following are the three levels of DFDs:

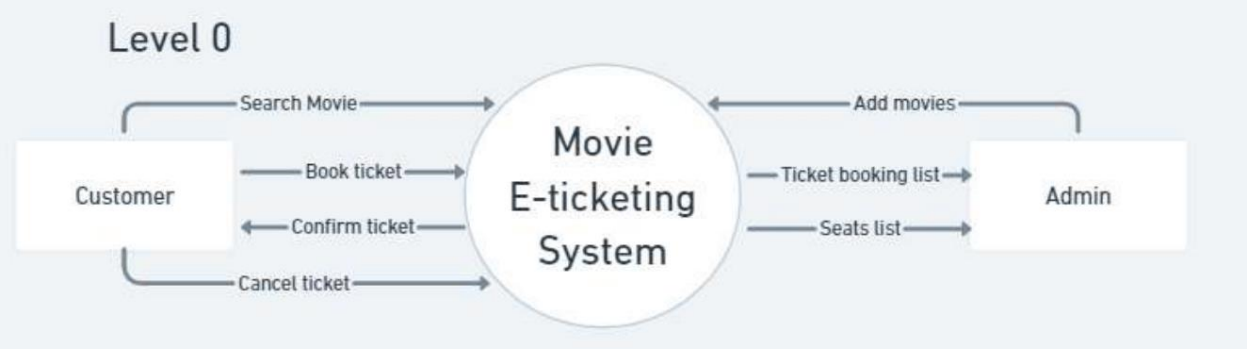
1. Level 0 DFD:- : Also known as a "context diagram," this is the highest level and represents a very simple, top-level view of the system being represented. It shows the major processes, data flows, and data stores in the system, without providing any details about the internal workings of these processes. It is designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts, and developers.

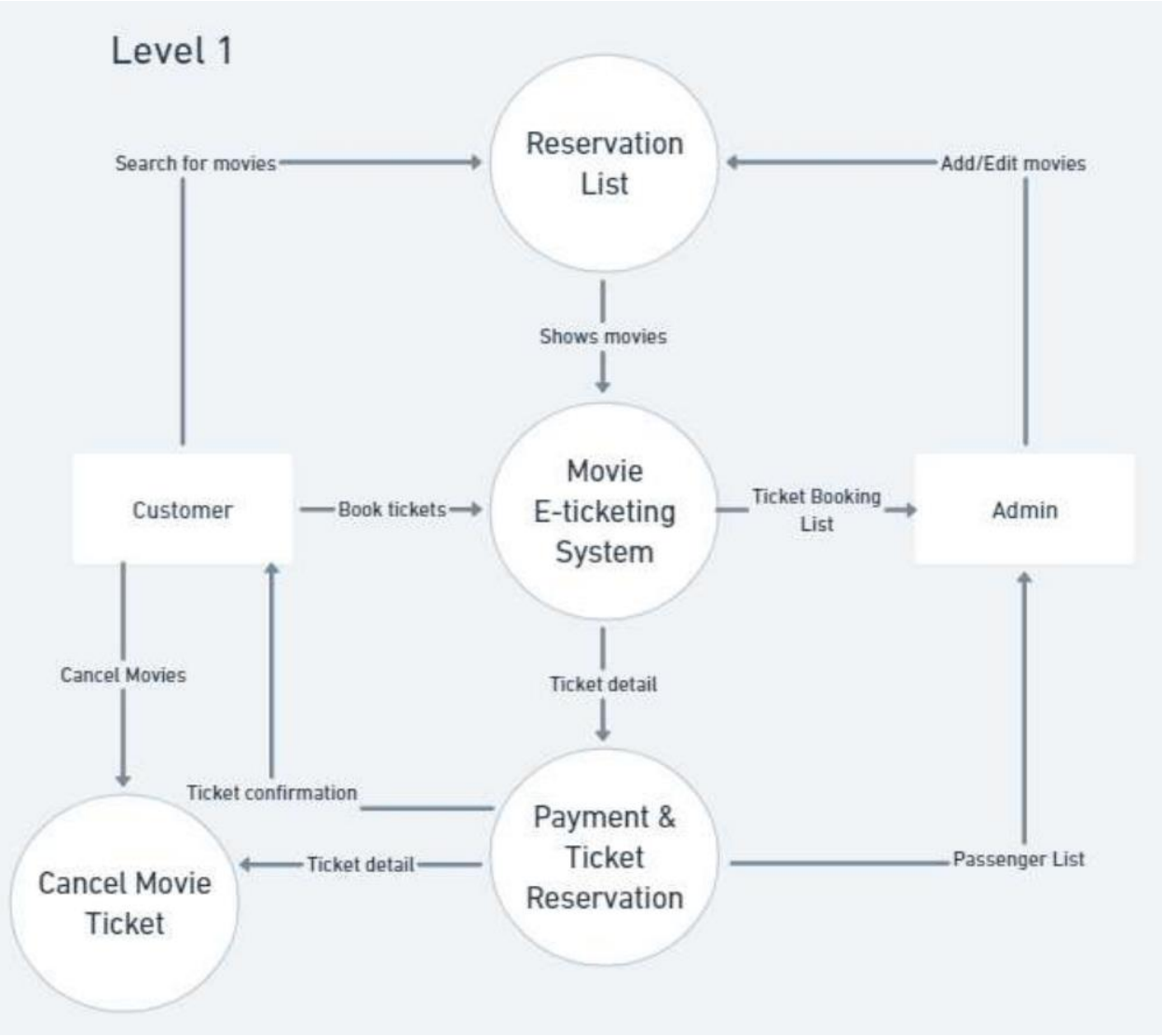
2. Level 1 DFD: This level provides a more detailed breakout of pieces of the context level diagram. It breaks down the system's single process node into subprocesses. Each sub-process is depicted as a separate process on the level 1 DFD. The data flows and data stores associated with each sub-process are also shown.

3. Level 2 DFD: This level goes one step deeper into parts of level 1. It may require more text to reach the necessary level of detail. It breaks processes down into more detailed sub-processes.

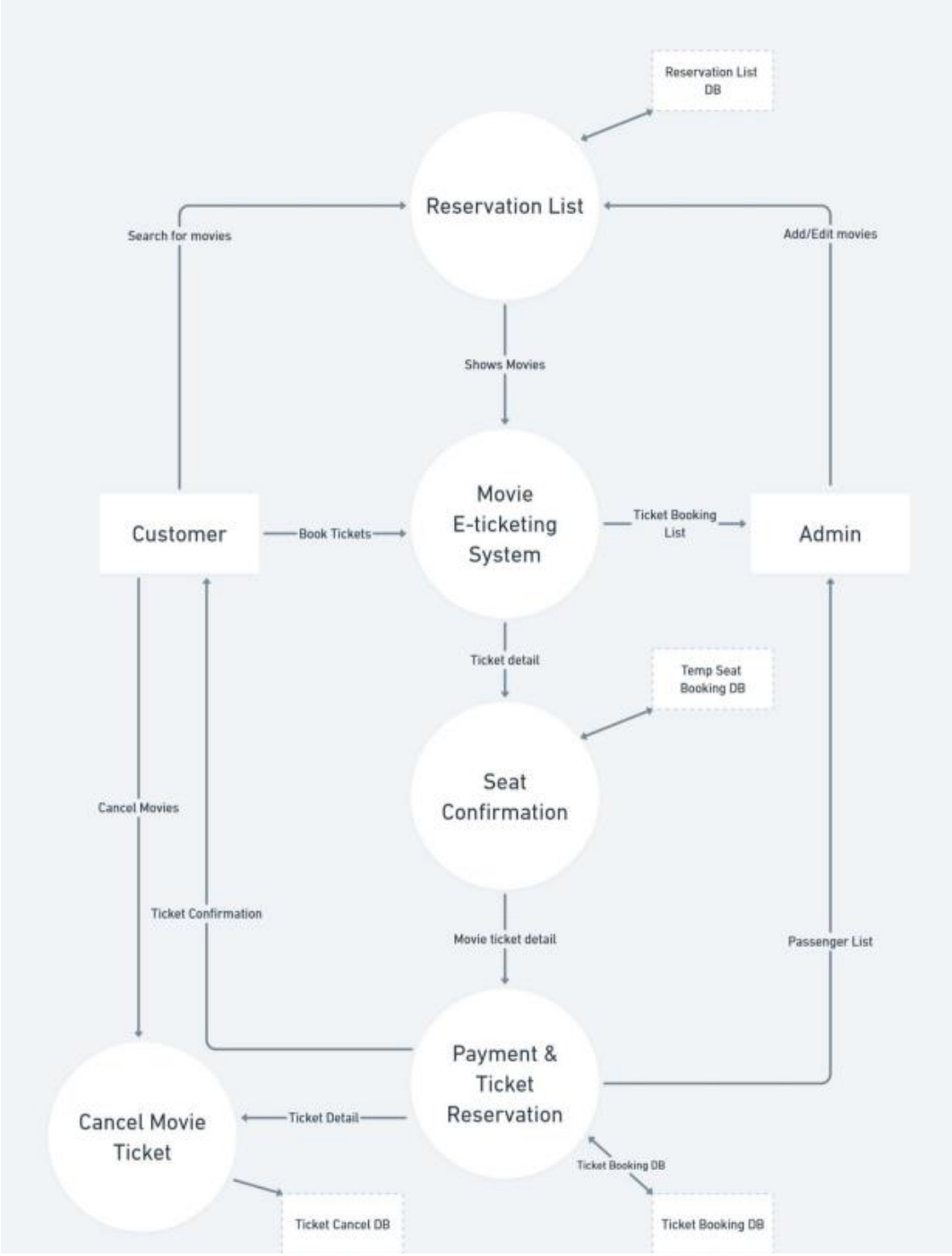
Each process on the level 2 DFD is depicted with a detailed description of its input, processing, and output. The data flows and data stores associated with each process are also shown.

The choice of DFD level depends on the complexity of the system and the level of detail required to understand the system. Higher levels of DFD provide a broad overview of the system, while lower levels provide more detail about the system's processes, data flows, and data stores. A combination of different levels of DFD can provide a complete understanding of the system.


Level 0
Search Movie
Add movies
Customer
Book ticket
Movie E-ticketing System
Ticket booking list
Admin
Confirm ticket
Seats list
Cancel ticket


Level 1
Reservation List
Search for movies
Add/Edit movies
Shows movies
Customer
Book tickets
Movie E-ticketing System
Ticket Booking List
Admin
Cancel Movies
Ticket detail
Ticket confirmation
Payment & Ticket Reservation
Cancel Movie Ticket
Ticket detail
Passenger List

Level 2


Reservation List DB
Reservation List
Search for movies
Add/Edit movies
Shows Movies
Customer
Book Tickets
Movie E-ticketing System
Ticket Booking List
Admin
Ticket detail
Temp Seat Booking DB
Seat Confirmation
Cancel Movies
Ticket Confirmation
Movie ticket detail
Passenger List
Payment & Ticket Reservation
Cancel Movie Ticket
Ticket Detail
Ticket Booking DB
Ticket Cancel DB
Ticket Booking DB

Exp 5:- To perform the user's view analysis for the suggested system: Use case diagram

Software Used: StarUML

Theory

Use case diagrams are a type of behavioral diagram that show the interactions between a system and its users, or actors. They are used to represent and model the functionality of a system. The main purpose of a use case diagram is to capture the functional requirements of a system. They are often used during the analysis and design phases of software development.

The components of a use case diagram are:

Actor: An actor is a person or entity that interacts with the system to achieve a goal. Actors can be people, organizations, or external systems.

System: The system is the software or hardware that is being modeled. The system is represented by a rectangle in a use case diagram.

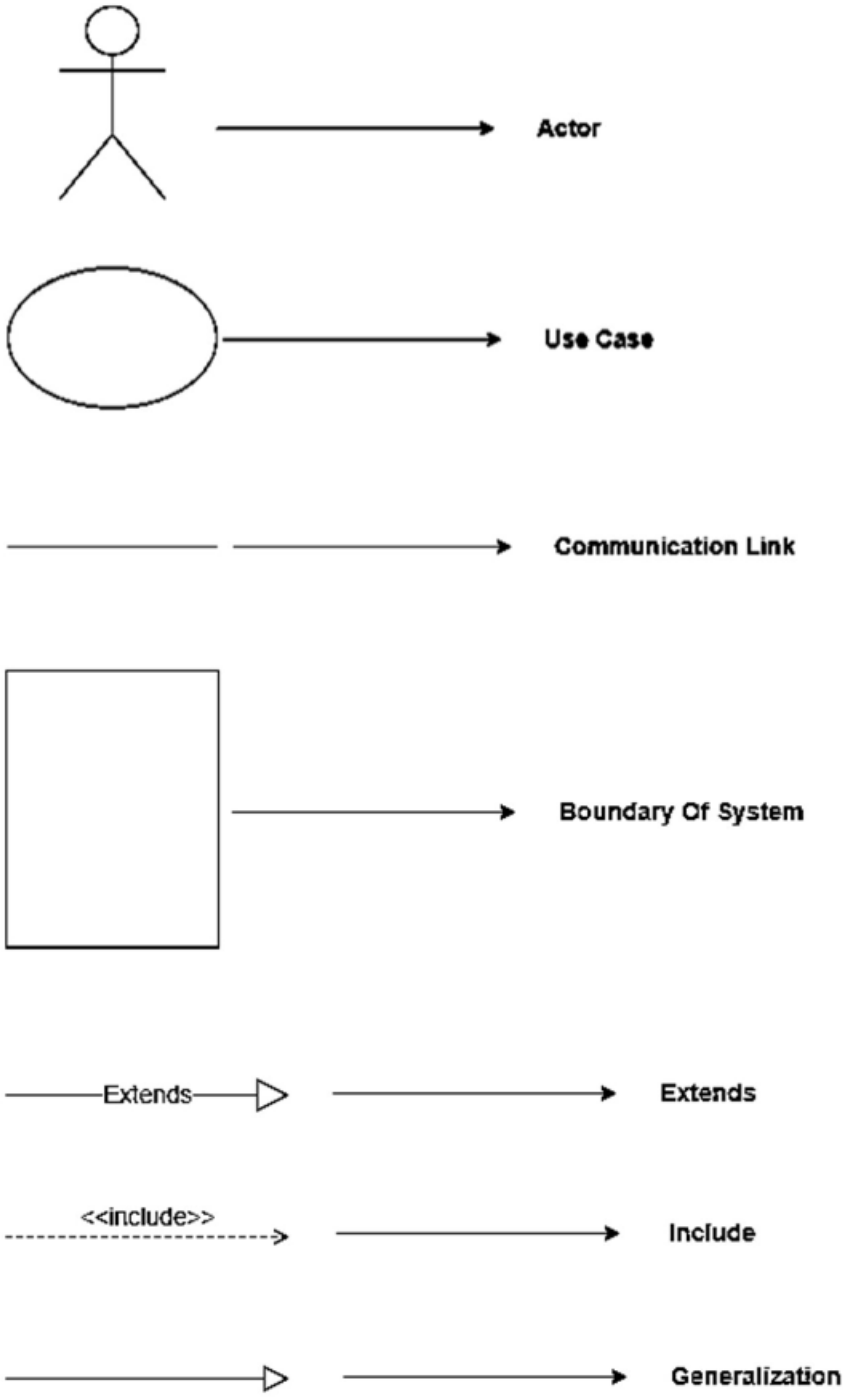
Use Case: A use case is a sequence of interactions between an actor and the system that results in the actor achieving a goal. Use cases are represented by ellipses in a use case diagram.

Relation: A relation shows how two actors or use cases interact with each other. Relations can be communication relations or association relations. Communication relations are shown by lines with arrows. Association relations are shown by lines with diamonds.

Generalization: A generalization relation shows that a use case is a more specialized version of another use case. Generalization relations are shown by lines with an open triangle at the end.

Include: An include relation shows that one use case includes the functionality of another use case. Include relations are shown by dashed lines with an "include" label at the end.

Exclude: An exclude relation shows that one use case excludes the functionality of another use case. Exclude relations are shown by dashed lines with an "exclude" label at the end.

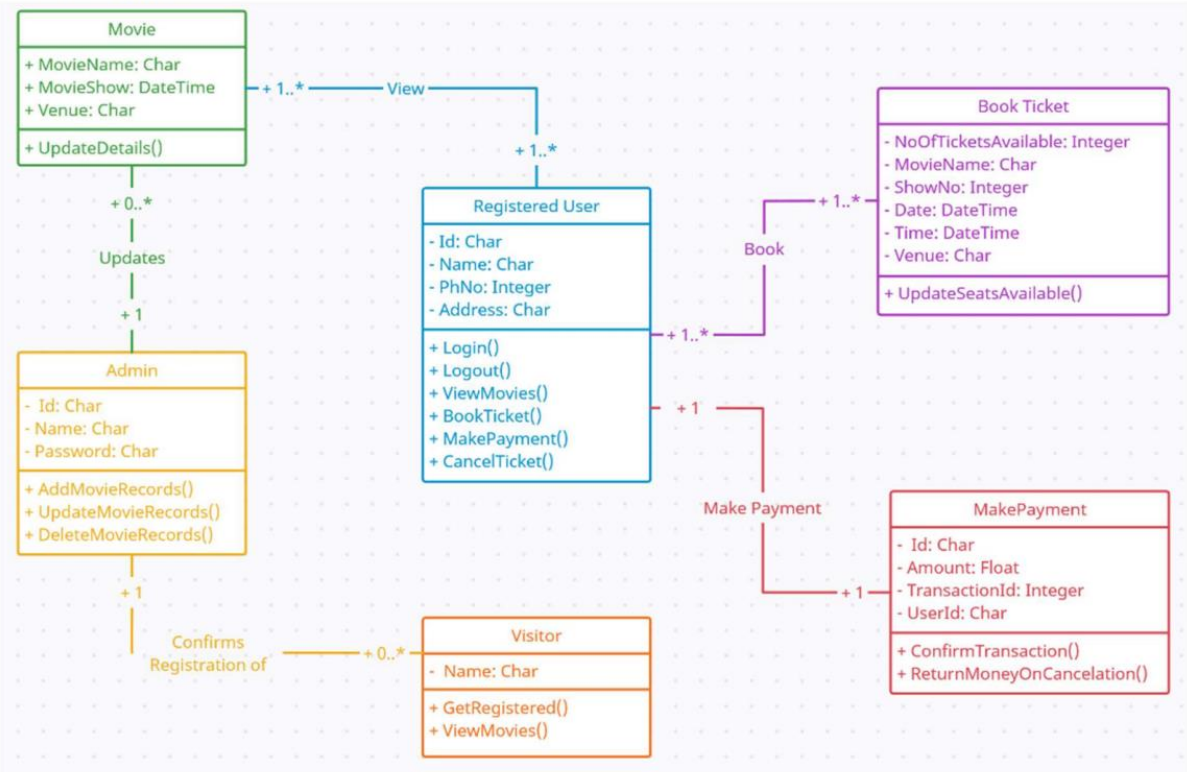


Exp6:- -

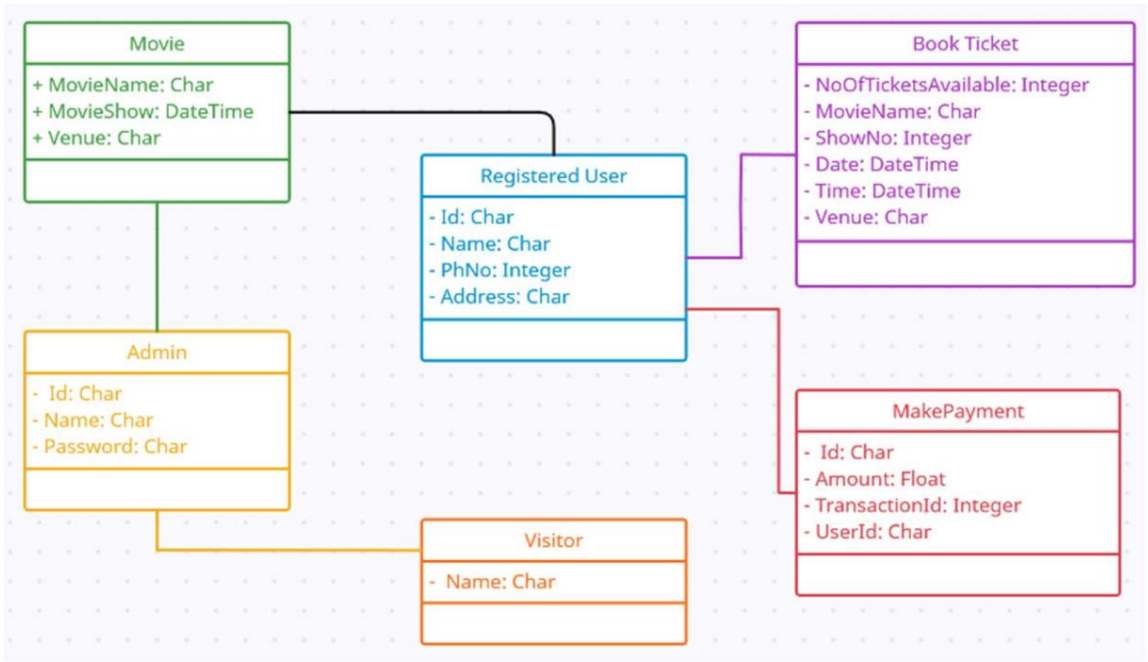
To draw the structural view diagram for the system: Class diagram, object diagram.

A class diagram is a type of UML structural diagram that represents the structure of a system at the level of classes and interfaces. It shows the features, constraints, and relationships (associations, generalizations, dependencies, etc.) between classes and interfaces. Class diagrams are widely used in object-oriented systems as they can be directly mapped with object-oriented languages An object diagram, on the other hand, is a type of UML structural diagram that represents a snapshot of the structure of a system at a specific moment in time. It shows the instance specifications of classes and interfaces (objects), slots with value specifications, and links (instances of association) between objects. Object diagrams can be considered as instance-level class diagrams, as they show the relationships between objects without representing the classes themselves. They are useful for explaining smaller portions of a system when the system class diagram is complex or for modeling recursive relationships

Class Diagram :-



Object Diagram :-



Experiment 7

Aim:- To draw the behavioral view diagram: State-chart diagram, Activity diagram.

Software Used :- Star UML

Theory:-

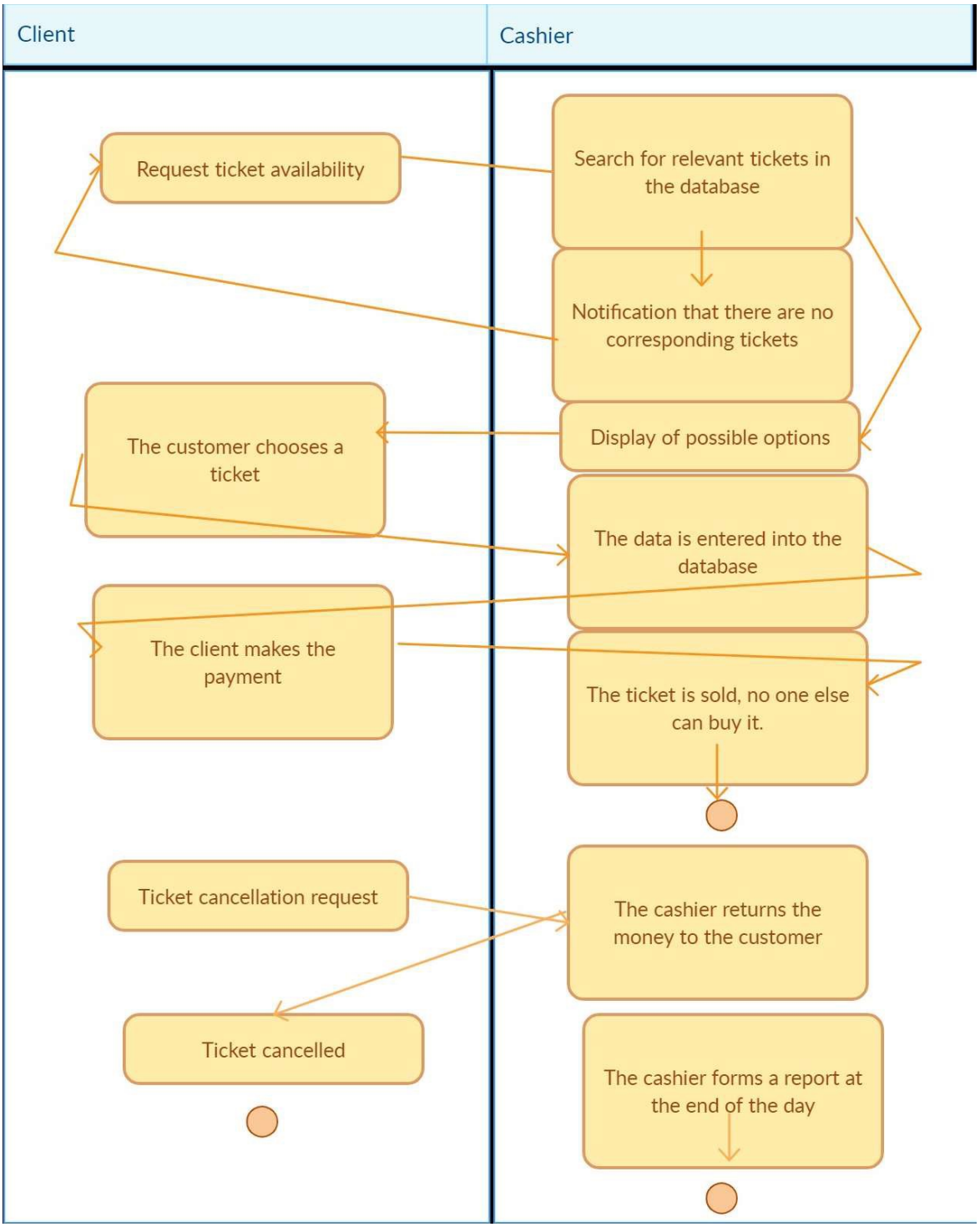
State chart Diagram :- These terms are often used interchangeably. So simply, a state diagram is used to model the dynamic behavior of a class in response to time and changing external stimuli. We can say that each and every class has a state but we don't model every class using State diagrams. We prefer to model the states with three or more states.

Activity Diagram :-In UML, the activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities.

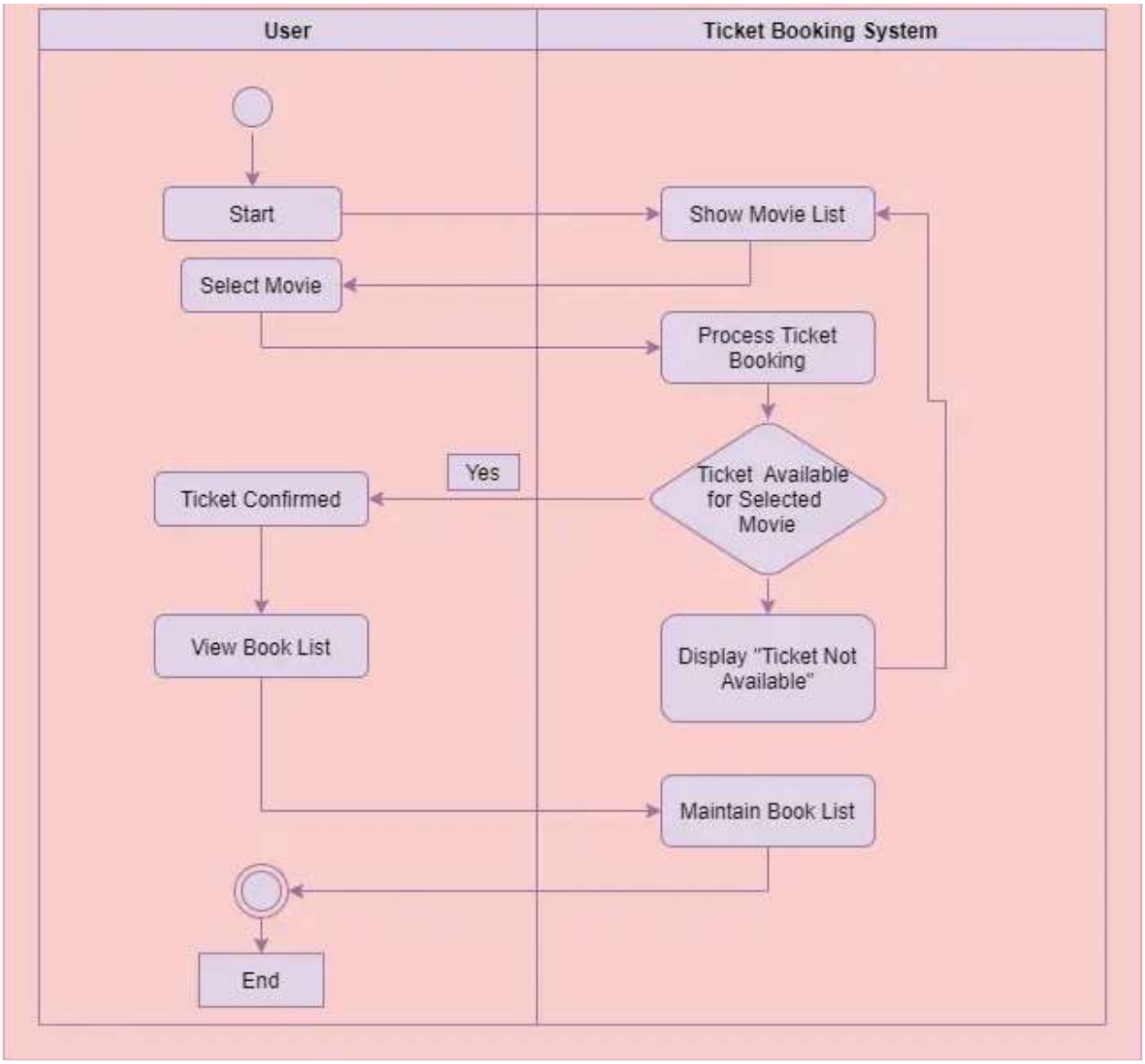
The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

It is also termed as an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are applied to model the behavioral diagram.

STATE DIAGRAM



ACTIVITY DIAGRAM



Experiment 8

Aim:- To perform the behavioral view diagram for the suggested system: Sequence diagram, Collaboration diagram.

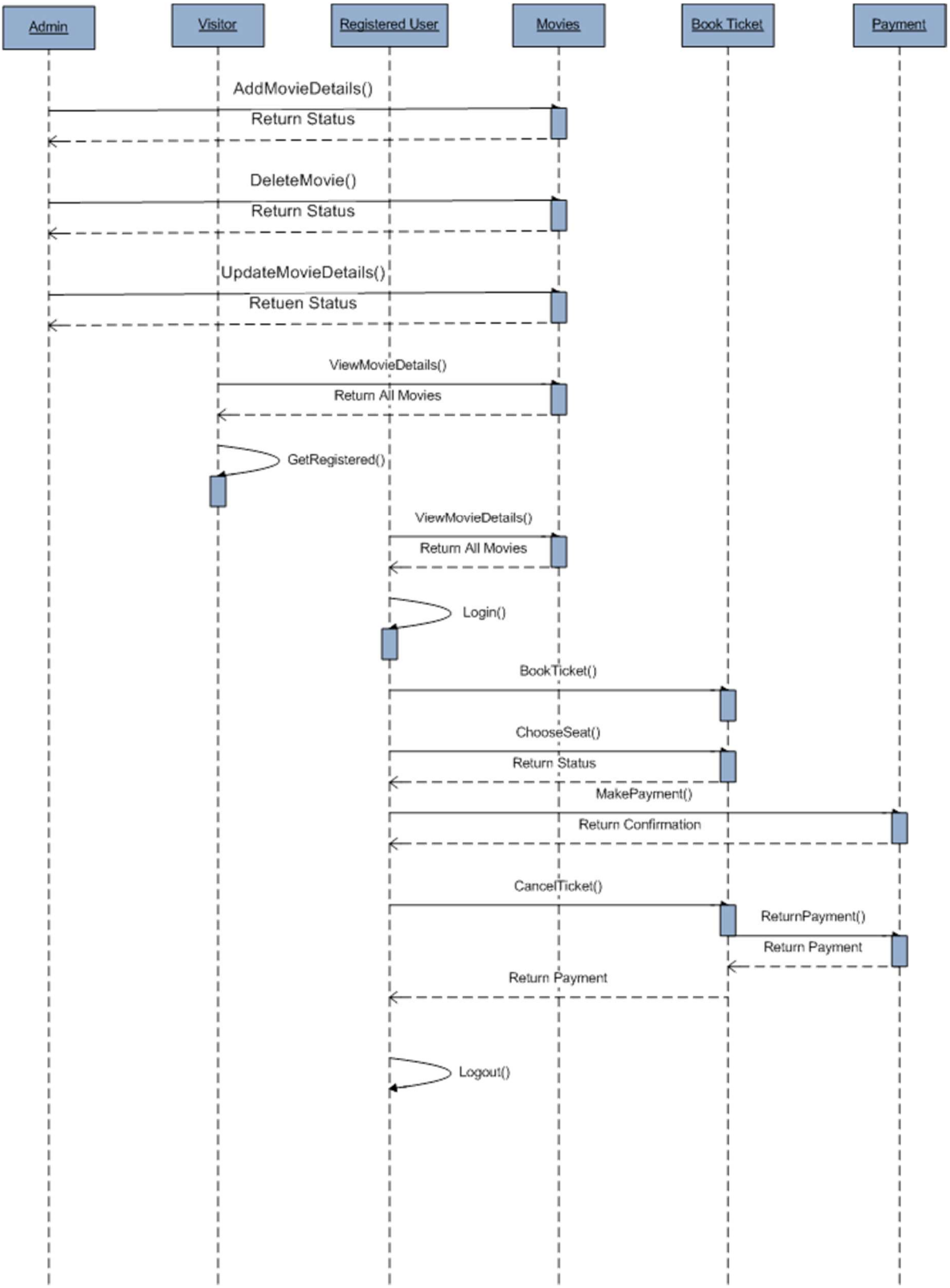
Software Used :- Star UML

Theory:-

Sequence Diagram :- A sequence diagram is the most commonly used interaction diagram. Interaction diagram – An interaction diagram is used to show the interactive behavior of a system. Since visualizing the interactions in a system can be a cumbersome task, we use different types of interaction diagrams to capture various features and aspects of interaction in a system. Sequence Diagrams – A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function.

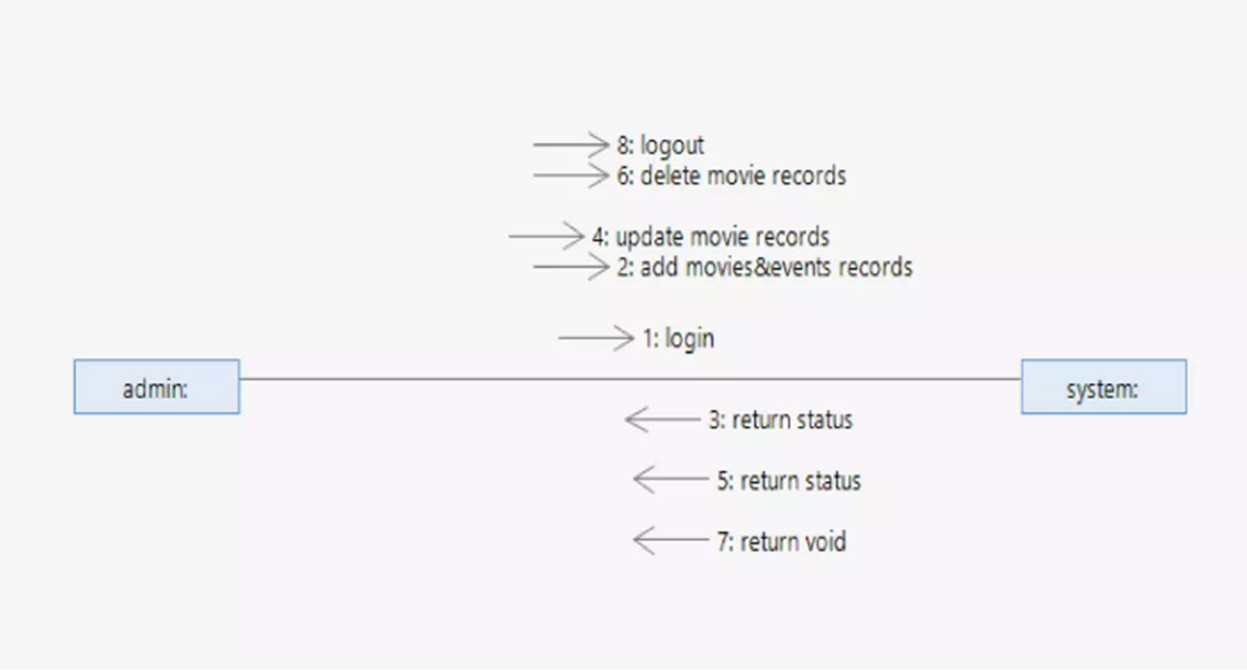
Collaboration Diagram:- The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

Sequence Diagram Diagram For E-Ticketing management system

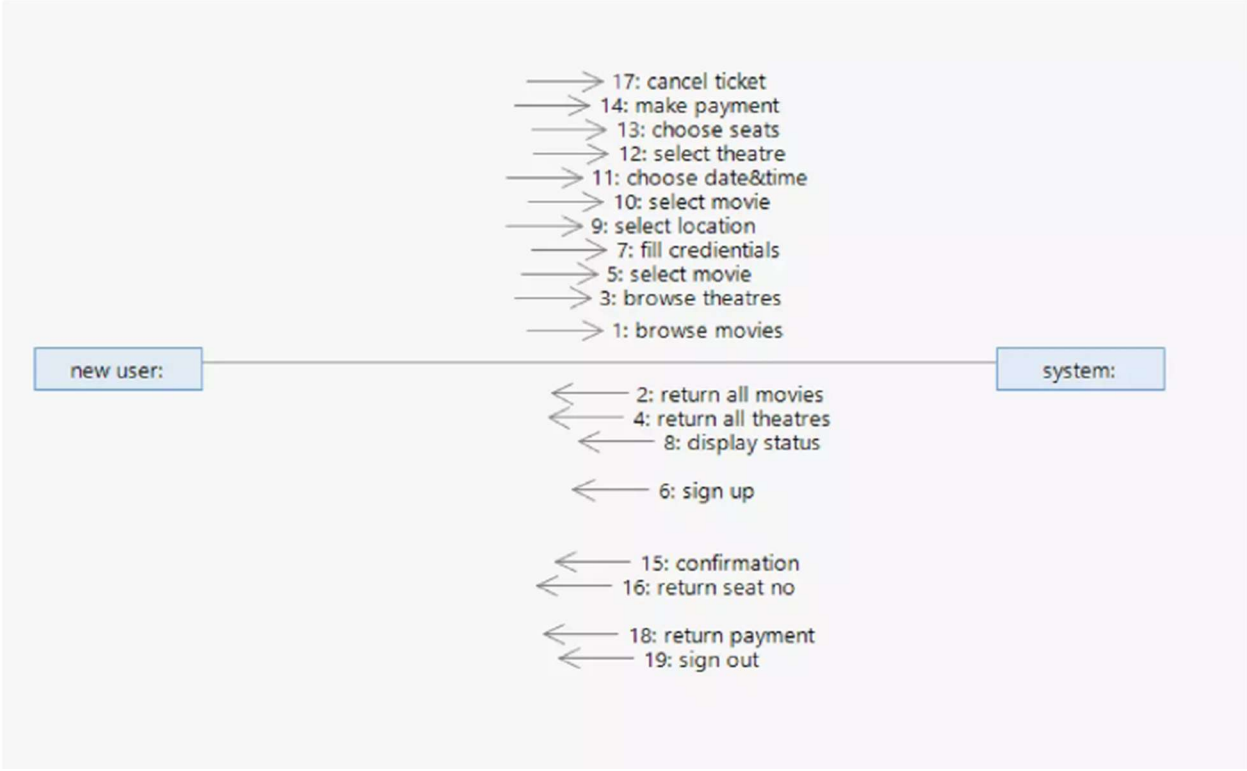


Collaboration Diagram For E-Ticketing management system

Admin Side



User Side



CBS 1

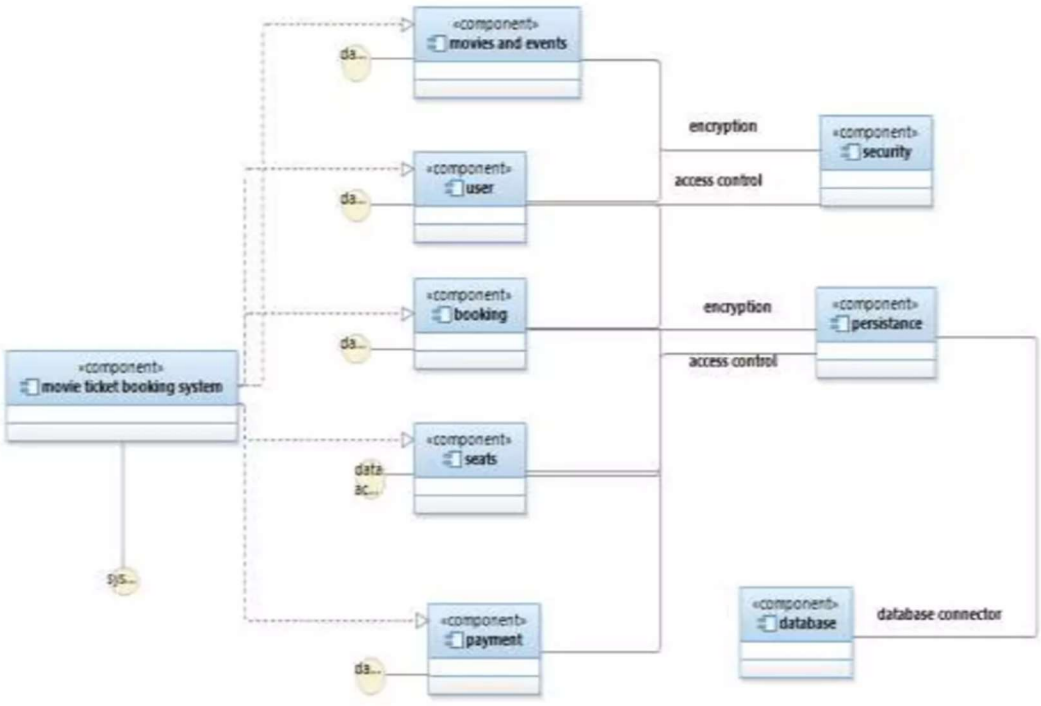
Aim:- To perform the implementation view diagram: Component diagram for the system.

Software Used :- Star UML

Theory:-

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behavior is explained by the provided and required interfaces.



CBS 2

Aim To perform the environmental view diagram: Deployment diagram for the system.

Software Used :- Star UML

Theory:-

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships.

It ascertains how software is deployed on the hardware. It maps the software architecture created in design to the physical system architecture, where the software will be executed as a node. Since it involves many nodes, the relationship is shown by utilizing communication paths.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behavior is explained by the provided and required interfaces.