Topic :- EXPERIMENT - 1 Date : 12/2/25

AIM:

To study the different aspects of software engineering and SDLC model.

DEFINITION:

Software Engineering is a systematic, disciplined and quantifiable approach to the development, operation and maintenance of software. It applies engineering principles (planning, designing, testing, management) to build high quality, reliable, cost effective software systems

The needs of Software Engineering:

1. Increasing complexity
2. Quality demands
3. Cost and time control
4. Team collaboration
5. Maintenance

SDLC (Software Development Life Cycle) is a systematic process used by software developers to design, develop, test and deploy high quality software. The main goal of SDLC is to produce software that meets or exceeds customer expectations, is completed within time and budget, and is efficient and maintenance.

Topic :- ..... Date : __/__/____

Stages of SDLC:

SDLC breaks down the complex process of software creation into manageable phases, ensuring that each step is carefully planned and executed.

1. Planning and Requirement Analysis:

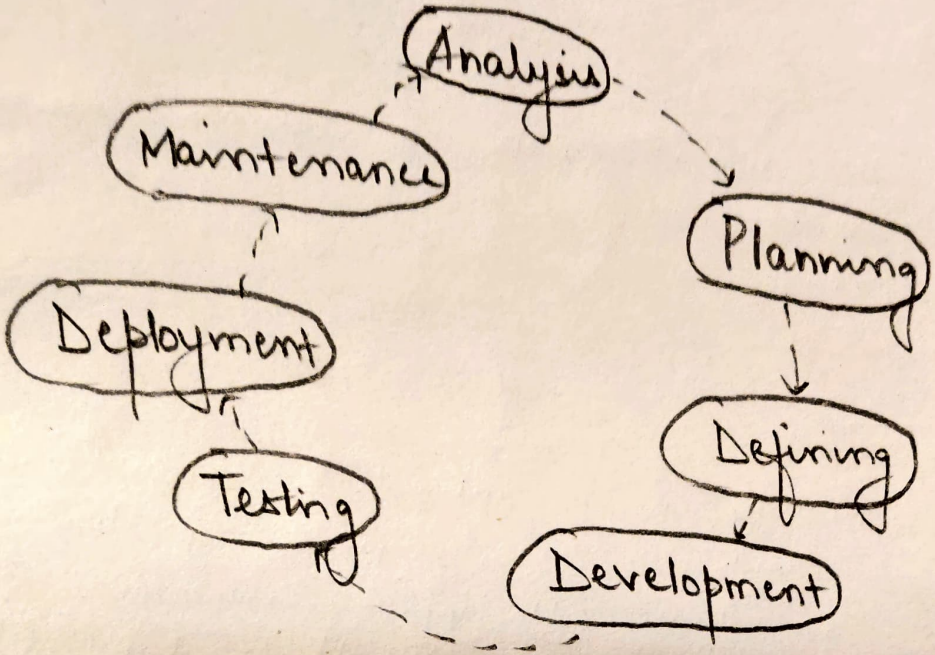
To understand the problem, define scope of the project, and gather the requirements from stakeholders.
It includes: - Meeting with stakeholders
- Identifying risks
- Estimating time and resources

2. Planning:

To analyze whether the project is feasible from a technical, operational and financial perspective.
It includes - Analyzing project viability
- Analyzing tech constraints and resources

3. Design:

To create the architecture and design of the system based on requirements.
It includes - High level & detail design
- Prototyping & mock ups
- Defining system components


Analysis
Planning
Defining
Development
Testing
Deployment
Maintenance

SDLC

Topic :- ..... Date : __/__/____

4. Development:

To translate the design into executable code.
It includes – Writing code
– Building databases
– Implementing features

5. Testing:

To ensure that the software is from defects & meets the requirements.
It includes – Unit testing, system testing
– Identifying & fixing bugs

6. Deployment:

To release the software to users or to production.
It includes – Preparing the environment for deployment
– User training and documentation.

7. Maintainenance:

To make updates to the software after deployment to fix issues, enhance functionality or improve performance.
It includes – Patching security vulnerabilities
– Adding new features
– Monitoring & optimization

Topic :- ..... Date : __/__/____

SDLC Models:

SDLC models provide a structured approach to plan, design, develop, test and deliver software projects. Different models offer various ways to organise these phases, each with unique advantages and suited to different project types and requirements.

1. Waterfall Mode:

A linear sequential approach where each phase must be completed before moving to the next. Its simple and easy to manage but inflexible to changes once a phase is finished. Best suited for projects with well defined requirements.

2. Agile Model:

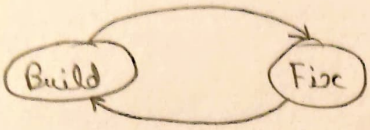
An iterative and incremental approach that delivers small, usable software pieces in short cycles called sprints. It emphasizes collaboration, flexibility and customer feedback, making it ideal for projects with changing requirements.

3. Iterative Model:

Builds the software through repeated cycles, gradually refining and expanding the system based on feedback. It helps identifying issues early but requires careful scope management to avoid uncontrolled growth.

Good Write

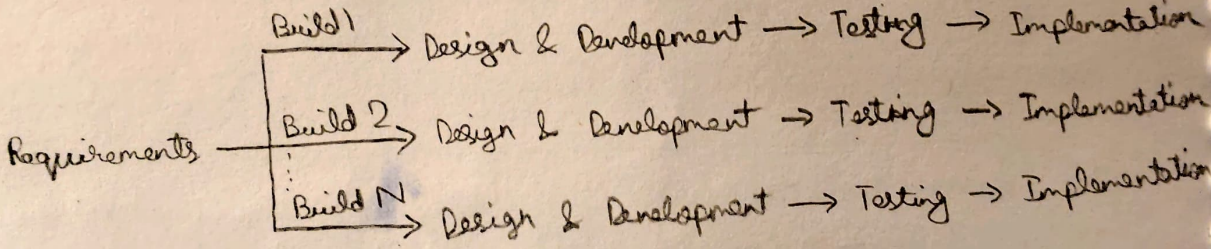
Teacher's Sign. .....



Build & Fix model

Feasibility study → Requirement analysis and specification → Design → Coding & Unit testing → System testing & integration → Maintenance

Waterfall Model



Incremental Process Model

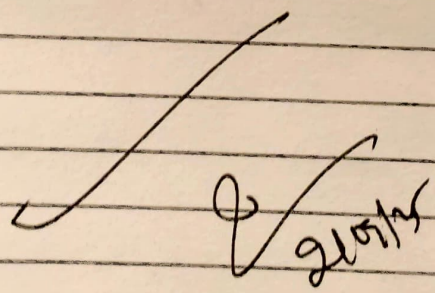
Topic :- ..... Date : __/__/____

4. V - Model :

An extension of Waterfall that pairs each development phase with a corresponding testing phase. This model emphasizes verification and validation, ensuring defects are detected early but remains relatively rigid.

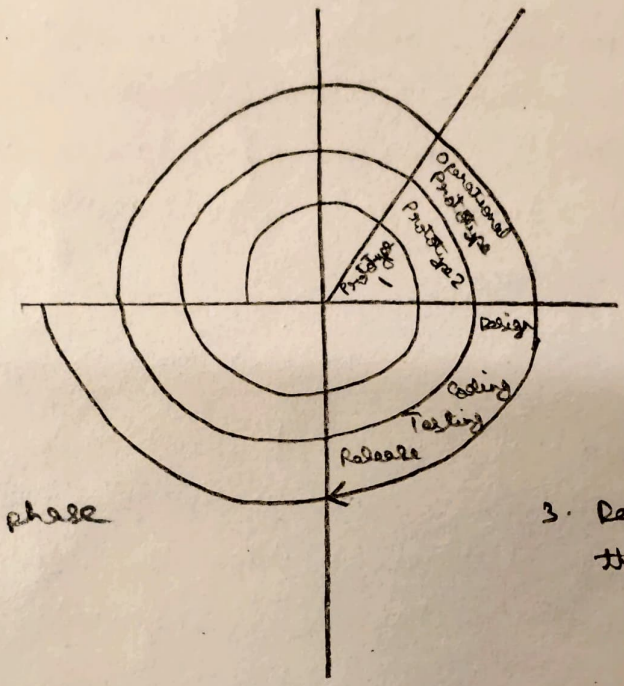
5. Spiral Model :

Combines iterative development with risk analysis, focusing on identifying and mitigating risks throughout the project. It's flexible and suitable for large, high risk projects but can be complex and costly to implement.



1. Plan objectives and find alternate solutions.

2. Risk analysis and resolving



4. Plan the next phase

3. Develop the next version of the product.

Spiral Model

Topic :- EXPERIMENT - 2 Date : 12/8/25

AIM:

To analyse the problem statement, description, functional and non-functional requirements.

PROBLEM STATEMENT:

In today's fast paced world, booking tickets via physical counters or phone calls is time consuming, error prone and inconvenient. Manual processes cause overbooking, lack of real time seat availability, long queues and poor data management for organisers. Users increasingly expect an instant, secure and seamless way to book tickets anywhere, anytime. The absence of a centralised automated ticketing platform leads to lost sales and dissatisfied customers.

PROJECT DESCRIPTION:

The Event Ticket Booking System is an online platform that automates and streamlines the entire ticket booking process for events of all kinds. It enables users to browse and select events, choose seats, make secure payments and receive electronic tickets instantly. Event organisers can efficiently manage event details, ticket inventory and sales analytics from an admin interface. The system aims to improve user convenience, increase sales, reduce errors and provide real-time updates on ticket availability.

Topic :- ..... Date : __/__/____

FUNCTIONAL REQUIREMENTS:

1. User Registration & Authentication: Users should be able to register, login and securely access their accounts.
2. Event Catalog: Display a list of events with detailed information such as venue, date, time, ticket categories and prices.
3. Seat Selection: Provide a visual seat map or ticket categories for users to select their preferred seats.
4. Booking and Payment: Enable users to book tickets, process payments securely, and generate electronic tickets.
5. Tickets Confirmation & Delivery: Automatically send booking confirmations and e-tickets via email or in-app notifications.
6. Admin Management: Allow event organisers to create, update and delete events, manage ticket inventories and monitor sales.
7. Search & Filter: Users can search events by keyword and filter by date, location or category.
8. Booking History: Users can view their past bookings and download tickets.
9. Cancellation & Refunds: Users can cancel bookings within a specified timeframe and request refunds according to policies.

NON - FUNCTIONAL REQUIREMENTS:

1. Performance: The system should handle multiple concurrent users without significant delays.

Topic :- ..... Date :__/__/____

especially during high-demand ticket sales.

2. Security: Protect user data and payment information with encryption and secure authentication mechanisms.
3. Scalability: The system must support an increasing number of users, events, and transactions as usage grows.
4. Reliability: Ensure high availability and minimal downtime, particularly during ticket sale launches.
5. Usability: The user interface should be intuitive and accessible on various devices, including desktops, tablets and smartphones.
6. Maintainability: The system should be designed to allow easy updates, bug fixes, and feature enhancements.
7. Compliance: Adhere to relevant data privacy regulations and payment processing standards.
8. Backup & Recovery: Implement data backup and recovery plans to prevent data loss.

21/04/25

Topic :- EXPERIMENT - 3 Date : __/__/____

AIM:

To draw the use diagram of the project.

THEORY:

A use case diagram is a part of Unified Modelling language (UML) used to represent the functional requirements of a system. It visually shows how different actors (users, admins and external systems) interact with the system.

Functions of Use Case diagrams:

- Requirement Analysis: What the system should do from user perspective.
- User - System Interaction: How users and other actors interact with system.
- System Scope Definition: Boundaries of the system.
- Communication Tool: Communicate requirements to stakeholders.
- Documentation: Part of system documentation for design and development.

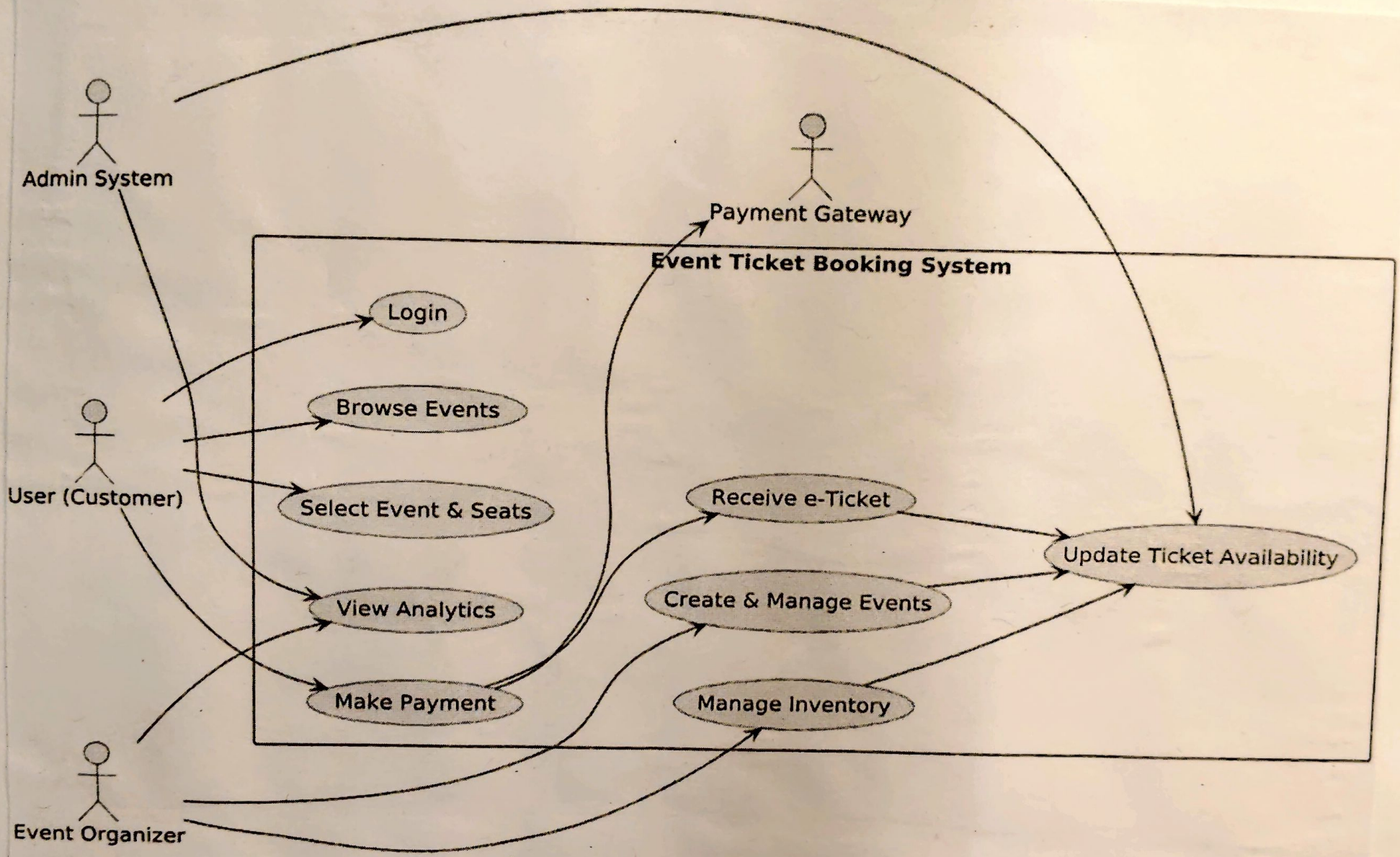
USE CASE:

Actors:

- User : → Login
  → Browse events
  → View Seat availability

Good Write

Teacher's Sign. _____


Admin System
Payment Gateway
Event Ticket Booking System
Login
Browse Events
User (Customer)
Select Event & Seats
Receive e-Ticket
Update Ticket Availability
View Analytics
Create & Manage Events
Make Payment
Manage Inventory
Event Organizer

Topic :- ..... Date : __/__/____

→ Select Seats
→ Make payment
→ Recieve e-ticket

- Event Organiser : → Create and manage events
  → Manage ticket inventory and pricing
  → View sales and analytics.

- Admin System : → Manage platform
  → Ensure availability
  → Monitor transactions
  → Provide analytics.

- Payment Gateway : → Process secure payments

Preconditions :

- User has internet access and can log into the platform.
- Events are already listed on the system database.
- Payment gateway is active and functional.

Main Flow :

1. User logs into the system.
2. Browses and selects an event.
3. System displays real time availability of seats.
4. Selects seats and proceeds to checkout.
5. Redirected to payment gateway.
6. Payment is verified and confirmed.

Good Write

Teacher's Sign. .....

Topic :- ..... Date : __/__/____

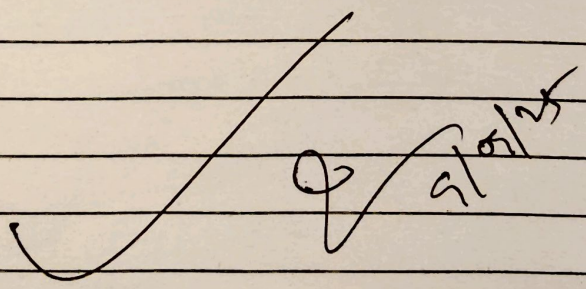
7. Electronic ticket / QR / PDF is generated.
8. User recieves the e-ticket via email / SMS / app.
9. System updates ticket availability in the database.
10. Event organizer view ticket sales and analytics via admin interface.

Postconditions :

- User recieves valid e-ticket.
- Ticket inventory is updated.
- Organizer can track sales and analytics.

Exceptions :

- Payment failure → User is notified, booking is cancelled.
- Seat already booked → System prompts user to select another seat.
- Event cancelled → Automatic refund is triggered.



Good Write

Teacher's Sign. .....

Topic :- EXPERIMENT - 4 Date : 2/9/25

AIM:

To draw 0-level DFD.

THEORY:

A data flow diagram (DFD) is a graphical tool that shows how data moves through a system. It represents the interaction between processes, entities and data flows.

The 0-level DFD is the topmost view of the system. It is also called a context diagram. It shows the entire system as a single process, interacting with external entities. It focuses only on who interacts with the system and what data flows in and out, not the internal details.

Functions of 0-level DFD:

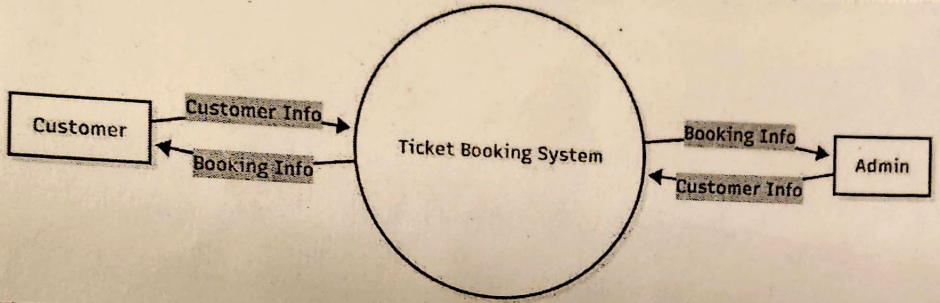
- Provides a broad overview of the entire system
- Identifies external entities interacting with the system.
- Describes data inflows and outflows.
- Serves as a foundation for creating lower level DFDs.

0-level DFD of the project:

1. External Entities:

- Customer : Interacts with the system by providing booking requests and customer details.
- Admin : Manages bookings and accesses customer information

Good Write Teacher's Sign. _____


Customer
Customer Info
Booking Info
Ticket Booking System
Booking Info
Customer Info
Admin

Topic :- ..... Date : __/__/____

from the system.

2. Central Process:

- Ticket Booking System: Acts as the main process that handles all requests and processes related to event ticket booking.

3. Data Flows:

- Customer → System: Customer information and booking details flow into the system.
- System → Customer: Confirmation of booking and related details are sent back.
- System → Admin: Provides booking information and customer details for management and analysis.
- Admin → System: Admin may update or verify customer and booking data.

Topic :- EXPERIMENT 5 Date : 9/9/25

AIM:

To draw 1- level DFD.

THEORY:

A level - 1 DFD is a detailed diagram that expands on the 0- level DFD by breaking down the main process into multiple sub-processes. It shows the flow of information between the system's key functions, data stores, external entities, and the interactions that occur step by step. It explains how the system actually works internally.

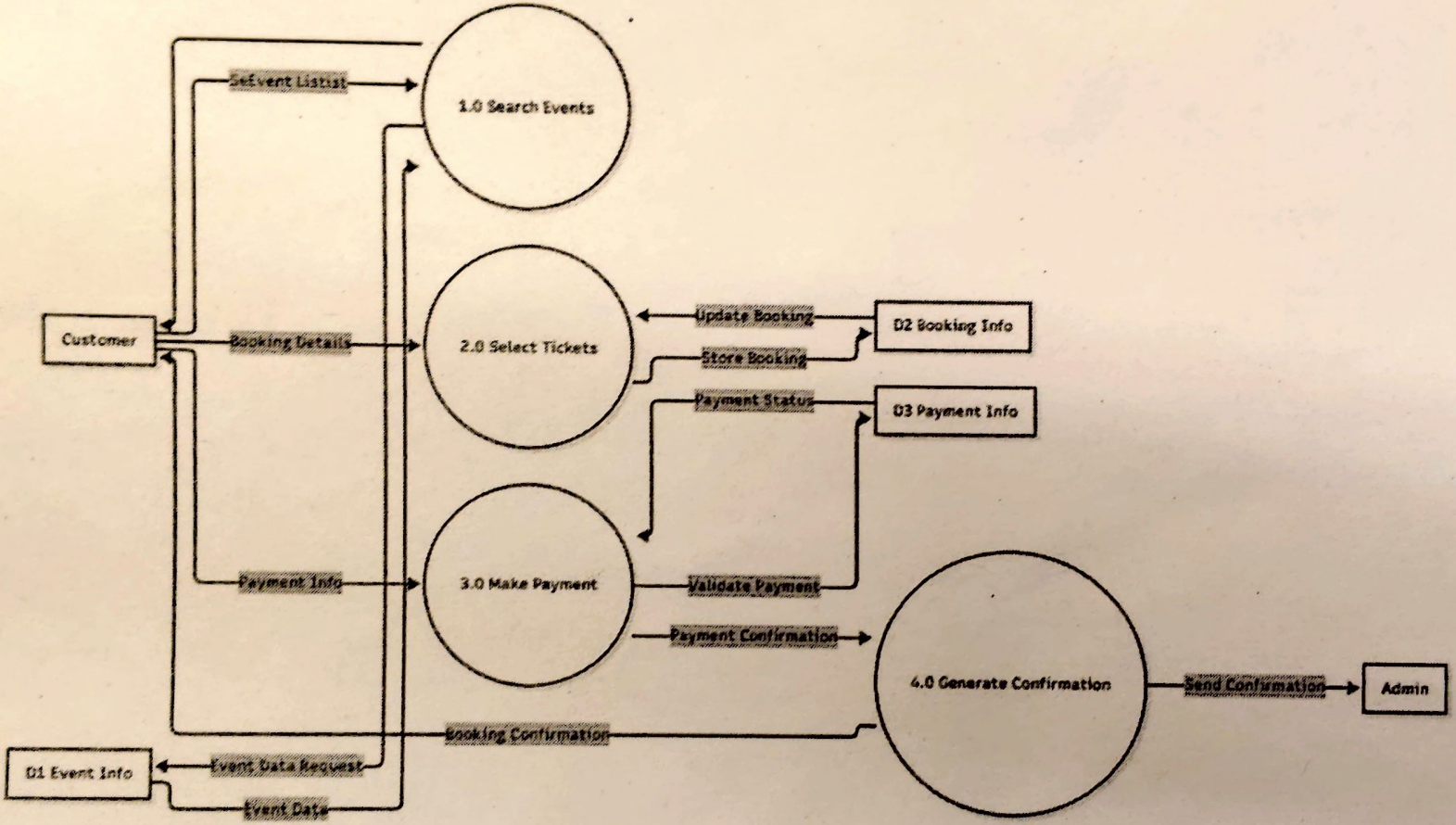
level 1 DFD of the Project:

1. External Entities:

- Visitor: Customer: The user who searches for events, book tickets and makes payments.
- Admin: An internal user or system administrator who receives confirmation details for record keeping or management.

2. Processes:

- Search Events: Handles the functionality of searching for available events.
- Select Tickets: Manages the user's selection and temporary reservation of tickets.
- Make Payment: Processes the financial transaction for the


Customer
Selvent Listist
Booking Details
Payment Info
1.0 Search Events
2.0 Select Tickets
3.0 Make Payment
4.0 Generate Confirmation
Update Booking
Store Booking
D2 Booking Info
Payment Status
D3 Payment Info
Validate Payment
Payment Confirmation
Send Confirmation
Admin
Booking Confirmation
D1 Event Info
Event Data Request
Event Data

Topic :- ..... Date : __/__/____

selected tickets.

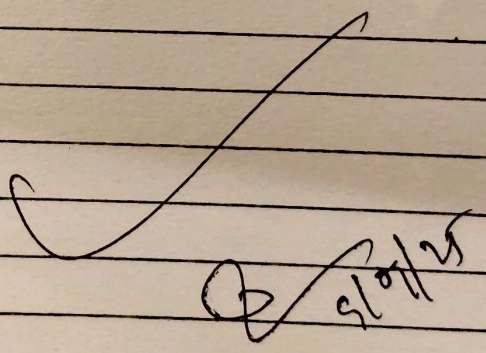
- Generate Confirmation: Finalizes the booking and distributes confirmation notices.

3. Data Stores:

- Event Info: Stores all information related to events.
- Booking Info: Stores details of customers' bookings.
- Payment Info: Stores payment transaction records and status.

4. Data Flows:

The pipelines through which data moves between entities, processes and data stores. They are represented by arrows and labelled with the name of the data being transferred.



Topic :- EXPERIMENT - 6 Date : 16/9/25

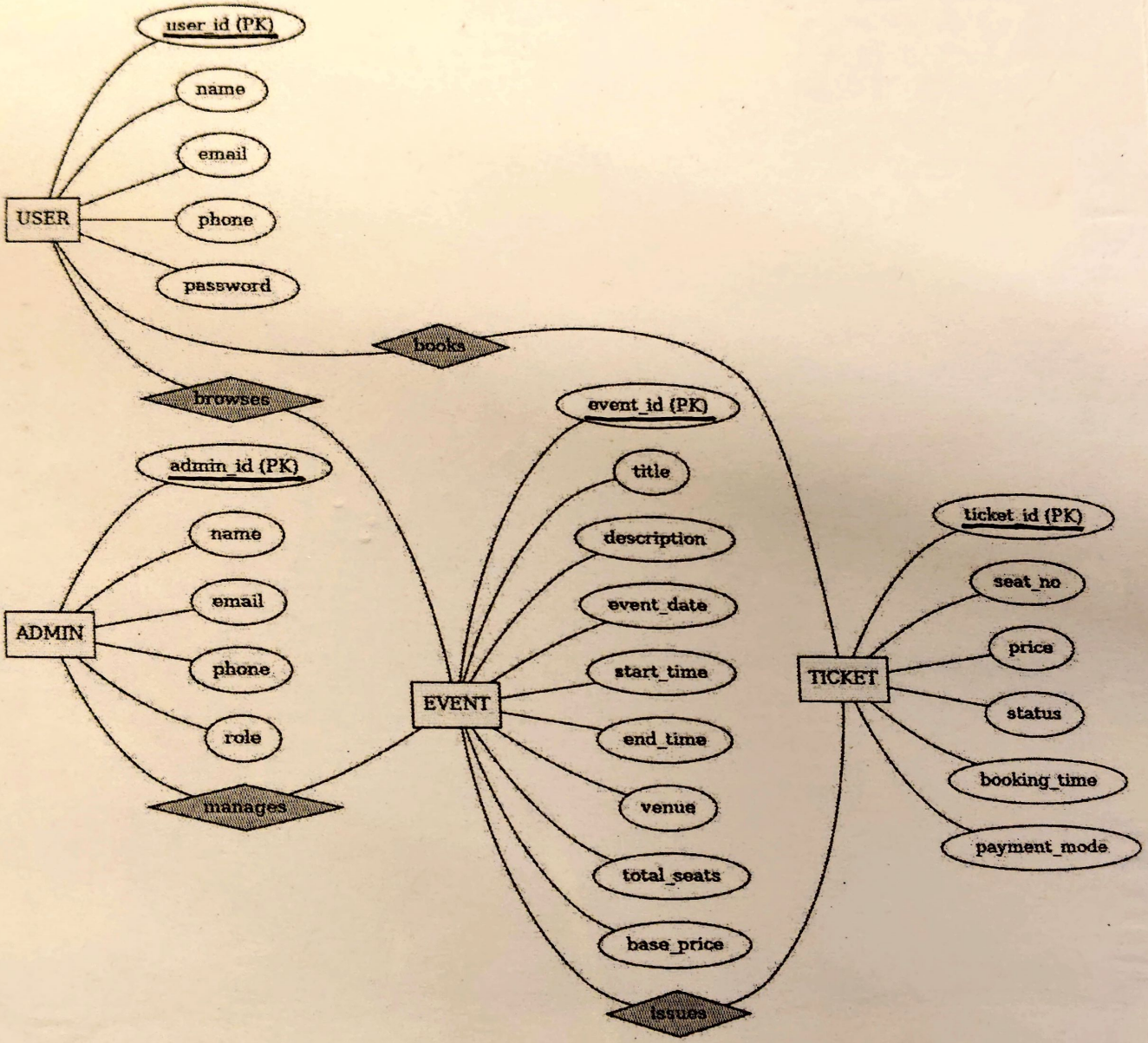
AIM:

To draw ER diagram.

THEORY:

An ER diagram is a visual representation of the data and its relationships within a database system. It is used during the database design phase to model the structure of the data and ensure efficient organisation. It shows how entities, their attributes and relationships are connected.

- Entity : Real world Object
  Shown as a rectangle
  Eg: Student, Teacher
- Attribute : Characteristics of an entity
  Shown as an oval
  Eg: Name, Age, Number
- Relationship : Link between entities
  Shown as diamond
  Eg: Student - enrolls in - Course
- Cardinality : Defines how many instances of one entity relate to another (1:1, 1:N, N:1, M:N)
- Participation : Shows whether all or some entities take part in relationship (total or partial)


user_id (PK)
name
email
USER
phone
password
books
browses
admin_id (PK)
name
email
ADMIN
phone
role
manages
event_id (PK)
title
description
event_date
start_time
EVENT
end_time
venue
total_seats
base_price
issues
ticket_id (PK)
seat_no
price
TICKET
status
booking_time
payment_mode