SOFTWARE ENGINEERING PROJECT

COMPUTER SCIENCE & ENGINEERING
(Artificial Intelligence & Machine Learning)

Submitted By

24WH1A6629 MS. E. MANUSREE
24WH1A6630 MS.Y. SPANDANA
24WH1A6631 MS.A. SIVA MANASA
24WH1A6632 MS. B. NAVYA
24WH1A6633 MS. E. SAI SPOORTHY
24WH1A6634 MS. A. NAVYASUSHMASRI
24WH1A6635 Ms. K. SAHASRA

under the esteemed guidance of
Ms. V. ASHA
Assistant Professor CSE (AI & ML)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
(Artificial Intelligence & Machine Learning)
BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with A Grade
Bachupally, Hyderabad – 500090

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with A Grade
Bachupally, Hyderabad – 500090

Department of Computer Science & Engineering
(Artificial Intelligence & Machine Learning)

CERTIFICATE

This is to certify that the **Software Engineering Project** is a bonafide work carried out by **MS.E. MANUSREE (24WH1A6629), MS.Y. SPANDANA (24WH1A6630), Ms. A. SIVAMANASA (24WH1A6631), MS.B.NAVYA (24WH1A6632), MS.E. SAI SPOORTHY (24WH1A6633), MS.A.NAVYASUSHMASRI (24WH1A6634),MS.K.SAHASRA (24WH1A6635)** in partial fulfilment for the award of BTech degree **in Computer Science & Engineering (AI & ML), BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad,** affiliated to Jawaharlal Nehru Technological University Hyderabad, under my guidance and supervision. The results embodied in the project work have not been submitted to any other**.**

Internal Guide	**Head of the Department**
MS.V. ASHA \	**Dr. B Lakshmi Praveena**
Assistant Professor	**HOD & Professor**
Dept of CSE(AI&ML)	**Dept of CSE(AI&ML)**

E-TICKECTING

Problem Statement:
To develop an online e-ticketing system that enables users to book, cancel, and manage tickets for events or transport services. The system also provides admin features for managing schedules and generating reports.
Scope:
The system will be web-based and accessible to both users and administrators. It supports real-time ticket booking, secure payments, user authentication, and ticket management.

Software Requirement Specification:

Functional requirements:

1. **User Registration and Authentication**
   - Users must be able to create an account using email or phone number.
   - Users can log in and log out securely.
   - Password recovery must be available.
2. **Search and Browse Events/Tickets**
   - Users can search for events/trips by name, location, or date.
   - Users can filter and sort search results.
3. **Booking and Ticket Purchase**
   - Users can select seats or ticket types (e.g., economy, VIP).
   - System calculates total price including fees and taxes.
   - Users can pay via credit card, debit card, or digital wallets.
4. **E-Ticket Generation**
   - After successful payment, the system generates a unique e-ticket.
   - The e-ticket must contain a QR code or barcode for validation.
5. **Ticket Cancellation and Refund**
   - Users can cancel tickets within a set time frame.
   - Refund amount and policy are displayed during cancellation.
6. **Admin Features**

- Admins can add, edit, or delete events/routes.
- Admins can view booking statistics and revenue reports.

7. **Notifications**
    - Users receive email/SMS notifications for booking confirmation, cancellation, or reminders.
8. **Ticket Validation**
    - Ticket checkers/scanners can validate e-tickets via QR code/barcode scanning.
9. **Multi-language and Multi-currency Support**
    - Users can select language and currency preferences (if system is global).

Non Functional requirements:

1. **Performance**
    - The system should respond to user actions (e.g., search, booking) within 2 seconds.
    - Ticket generation should not exceed 1 second per request.
2. **Scalability**
    - System should handle high loads during peak times (e.g., 100,000 users during event sales).
3. **Availability**
    - System must be available 99.9% of the time, except during scheduled maintenance.

4.**Security**

- All transactions must be encrypted (HTTPS, SSL).
- User data and payment information must follow industry security standards (e.g., PCI DSS).
- Implement 2-factor authentication for users and admins.

5. **Usability**
    - The system interface should be intuitive and user-friendly for users of all ages.
    - Support accessibility (e.g., screen readers, keyboard navigation).
6. **Maintainability**
    - System should support modular code for easy updates and bug fixes.
    - Admin dashboard should allow non-technical staff to manage content.

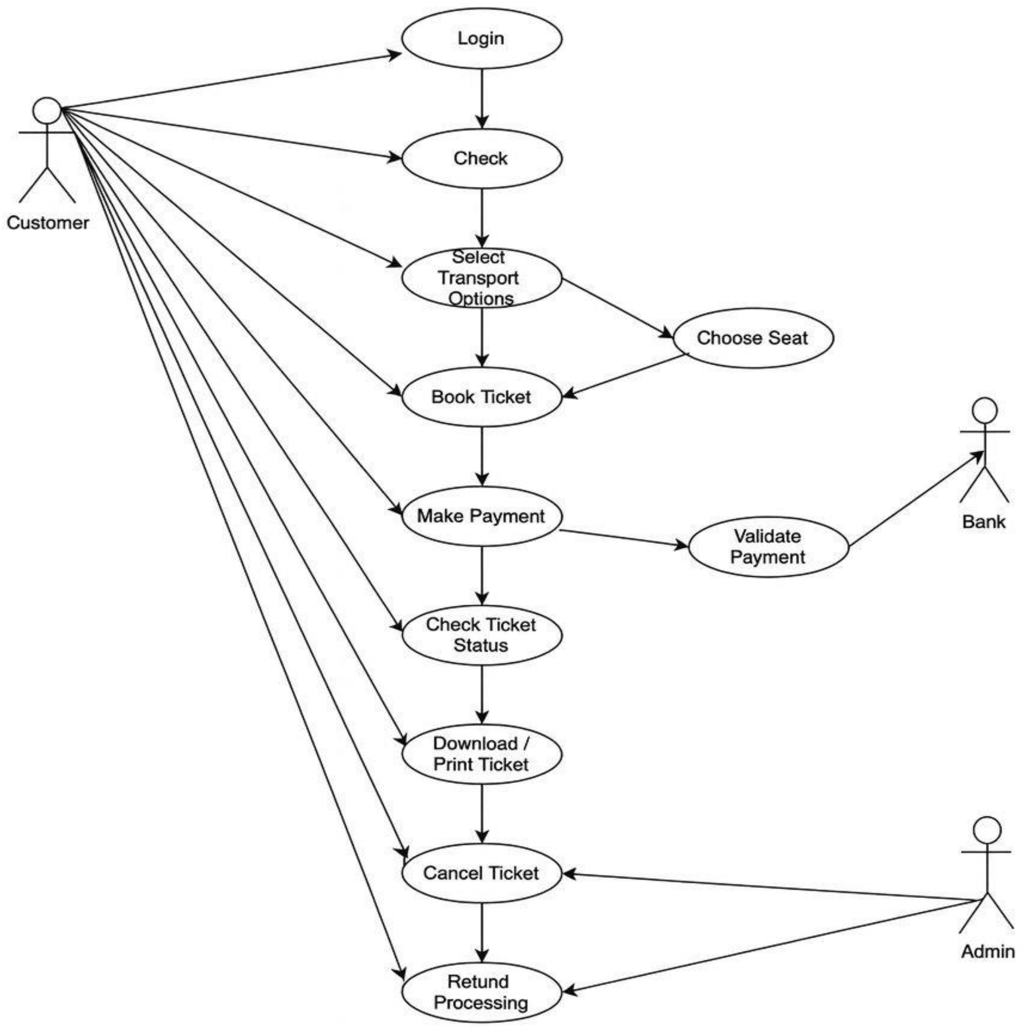
7. **Portability**

- System should run on web browsers (Chrome, Safari, Firefox) and mobile devices

UML Diagrams

Use Case Diagram:

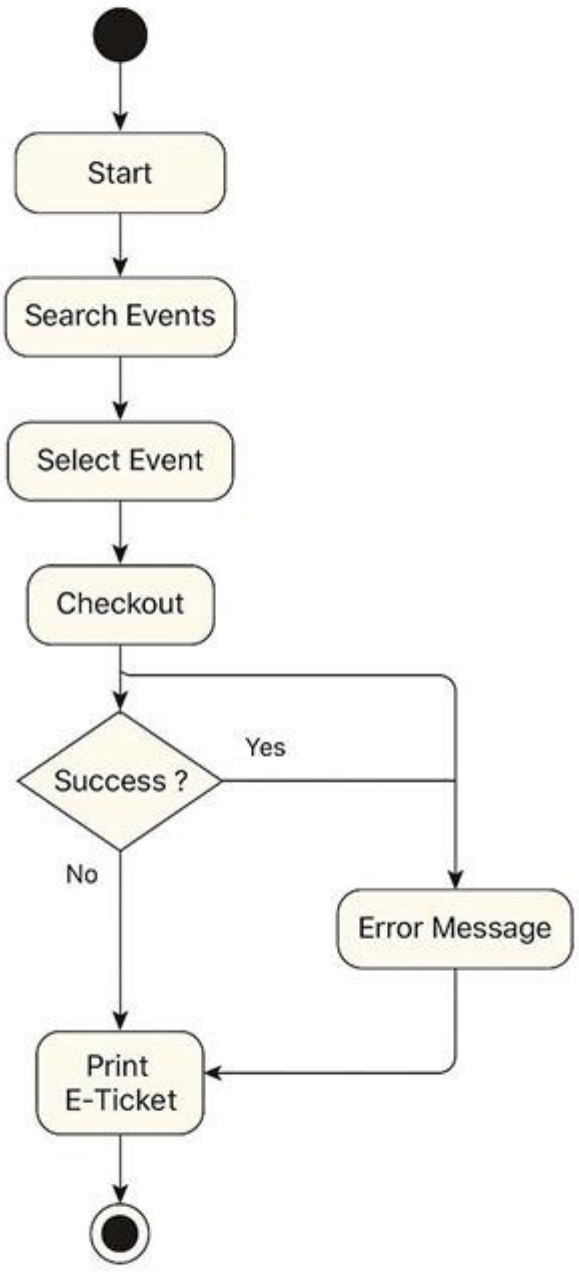
A use case diagram is a visual representation that means the different interactions between the user, or actors, and a system that actually describes the system's functional requirements. It, therefore, indicates all the use cases, or specific functionalities or processes, which the system provides and how various different actors are going to interact with those use cases



]

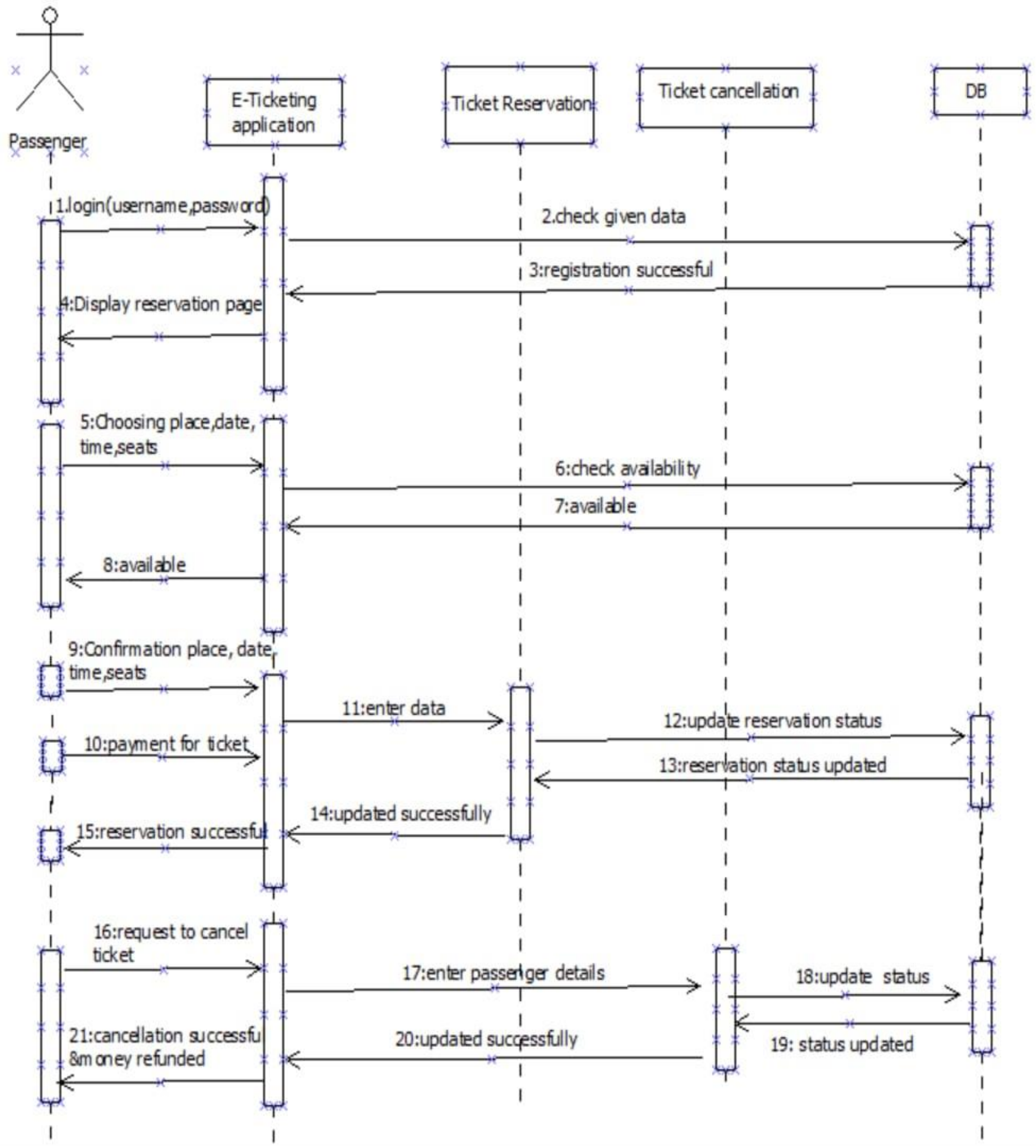
Activity Diagram:

An activity diagram is a UML diagram that depicts the flow of activities within a process, showing actions, decisions, and parallel workflows. It uses rounded rectangles for activities, arrows for transitions, and diamonds for decision points.



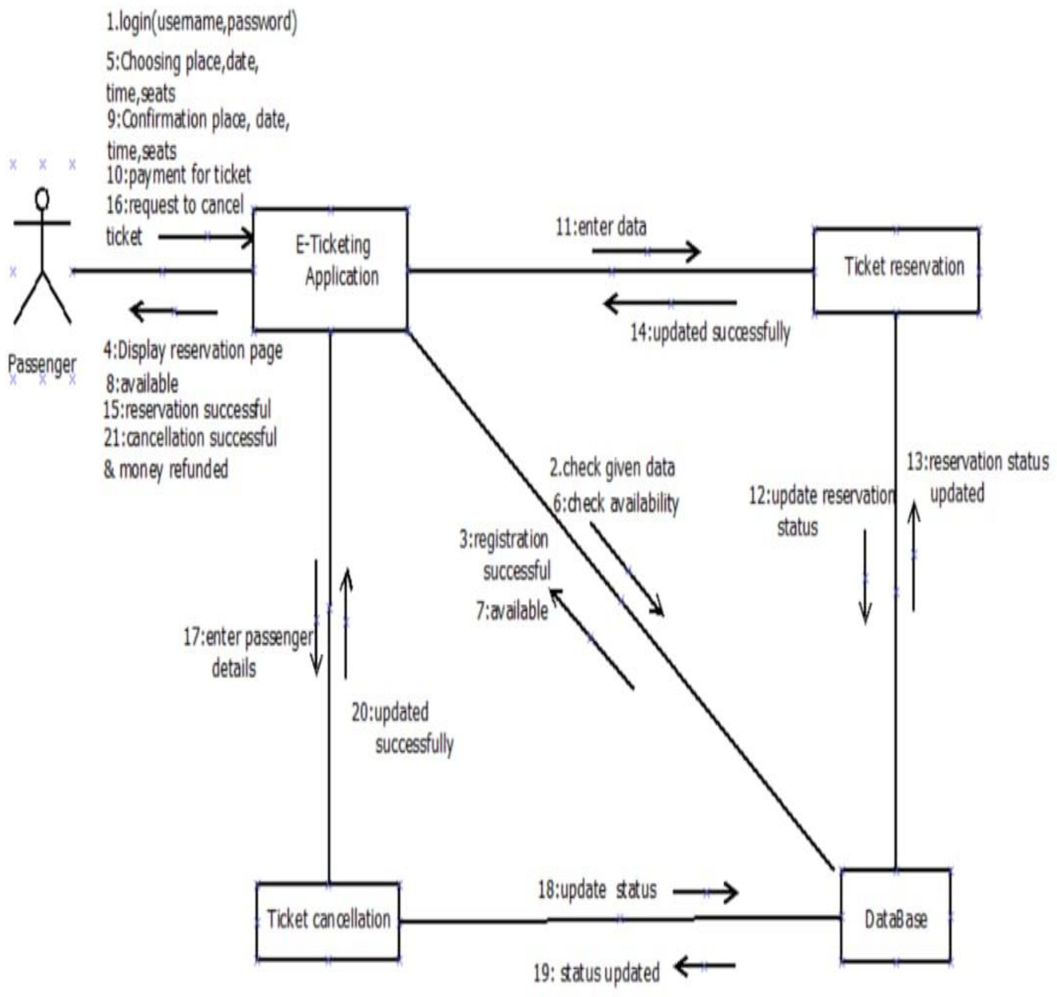
Sequence Diagram:

A sequence diagram is a type of UML diagram that illustrates how objects interact in a specific scenario over time. It shows the sequence of messages exchanged between objects, represented as vertical lines, with horizontal arrows indicating the messages or calls. The diagram captures the order of interactions, highlighting timing and the relationships between different components in a process. Sequence diagrams are useful for modelling dynamic behaviour, clarifying system operations, and understanding the flow of control in complex systems.


Passenger
E-Ticketing application
Ticket Reservation
Ticket cancellation
DB
1.login(username,password)
2.check given data
3:registration successful
4:Display reservation page
5:Choosing place,date, time,seats
6:check availability
7:available
8:available
9:Confirmation place, date, time,seats
10:payment for ticket
11:enter data
12:update reservation status
13:reservation status updated
14:updated successfully
15:reservation successful
16:request to cancel ticket
17:enter passenger details
18:update status
19: status updated
20:updated successfully
21:cancellation successful &money refunded

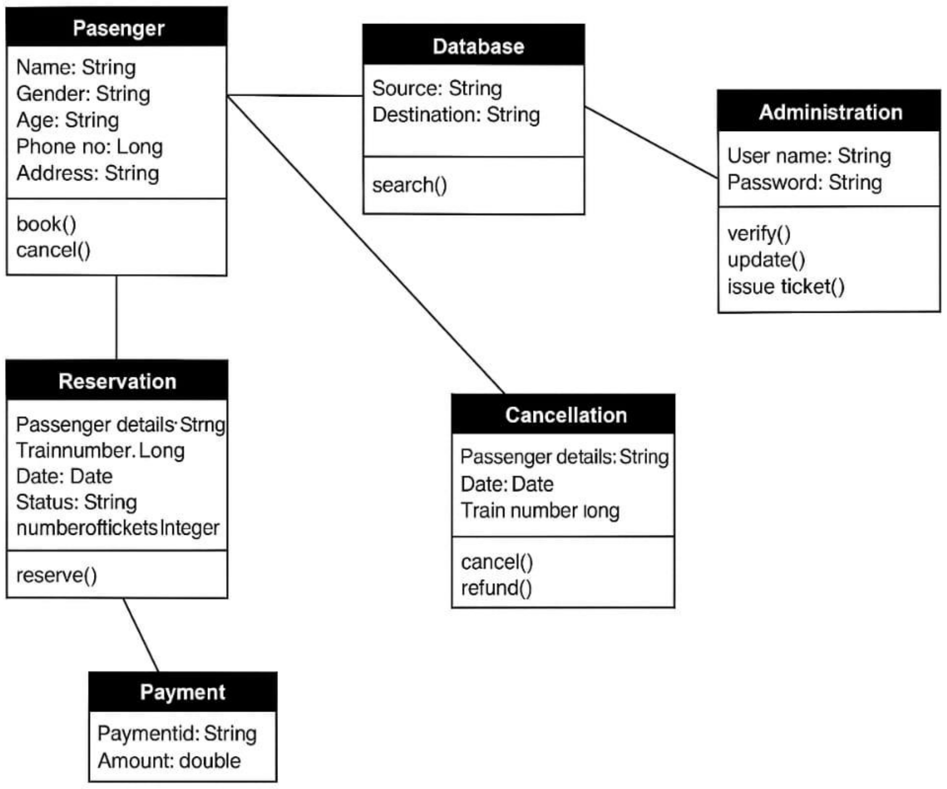
Collaboration Diagram:

A collaboration diagram, or communication diagram, illustrates the interactions between objects in a system, emphasizing their relationships. Objects are represented as circles, while messages are shown as label arrows connecting them. The diagram uses numbered arrows to indicate the sequence of message exchanges. It is useful for visualizing how objects work together to complete tasks within a process.


1.login(username,password)
5:Choosing place,date, time,seats
9:Confirmation place, date, time,seats
10:payment for ticket
16:request to cancel ticket
Passenger
E-Ticketing Application
11:enter data
Ticket reservation
14:updated successfully
4:Display reservation page
8:available
15:reservation successful
21:cancellation successful & money refunded
2.check given data
6:check availability
3:registration successful
7:available
12:update reservation status
13:reservation status updated
17:enter passenger details
20:updated successfully
Ticket cancellation
18:update status
19: status updated
DataBase

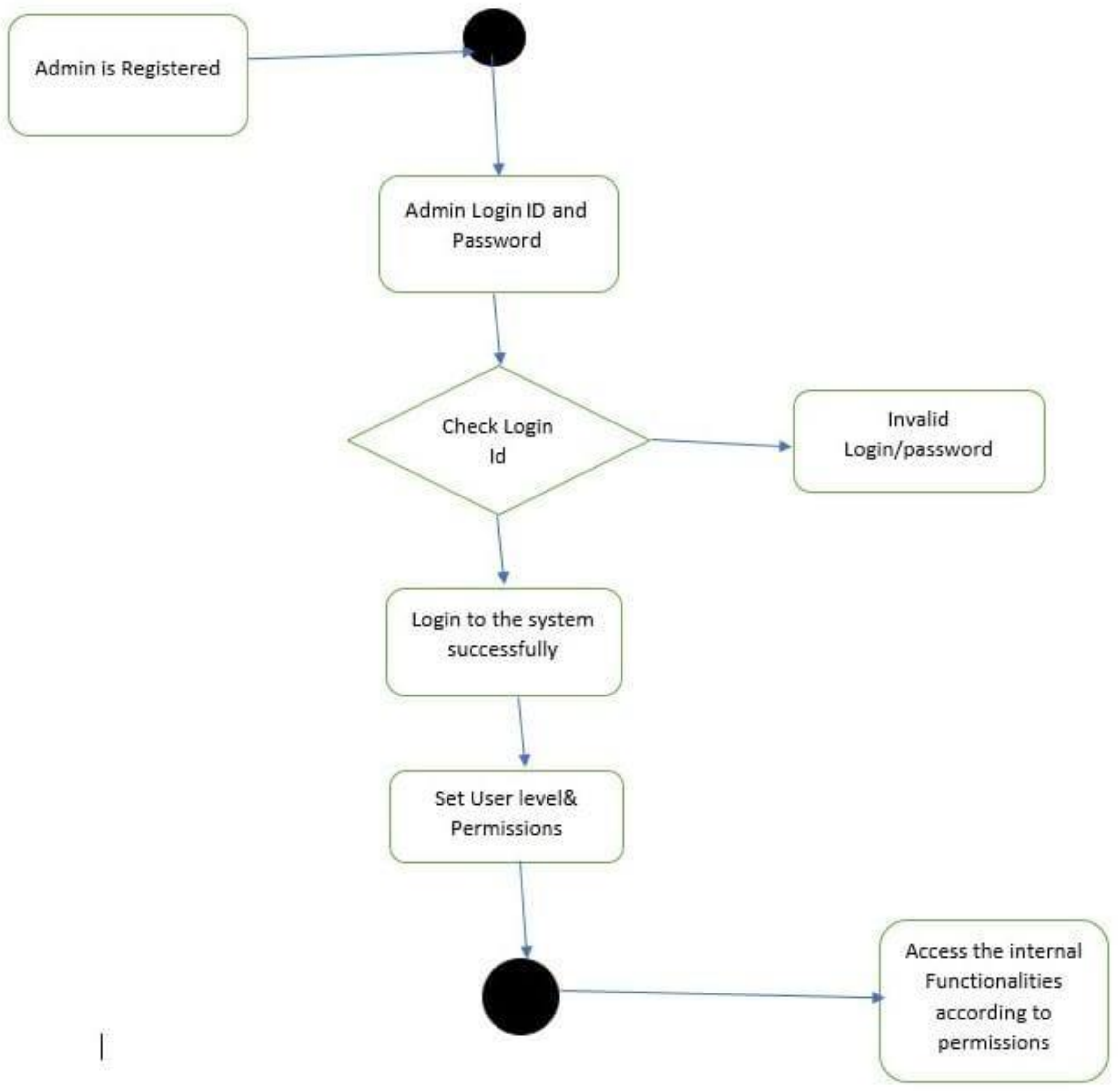
Class Diagram:

A class diagram is a UML diagram that depicts the static structure of a system, showing its classes, attributes, methods, and relationships. Classes are represented as rectangles divided into sections for the name, attributes, and operations. Relationships such as associations and inheritances are illustrated with lines and symbols. Class diagrams are essential for modelling system architecture and guiding development processes.



State-chart Diagram:

A state-chart diagram, or state machine diagram, illustrates the dynamic behaviour of an object by showing its possible states and the transitions between them. States are represented as rounded rectangles, while arrows indicate transitions triggered by events. The diagram can also include entry and exit actions for each state. State-chart diagrams are valuable for modeling object lifecycles and system responses to various events.

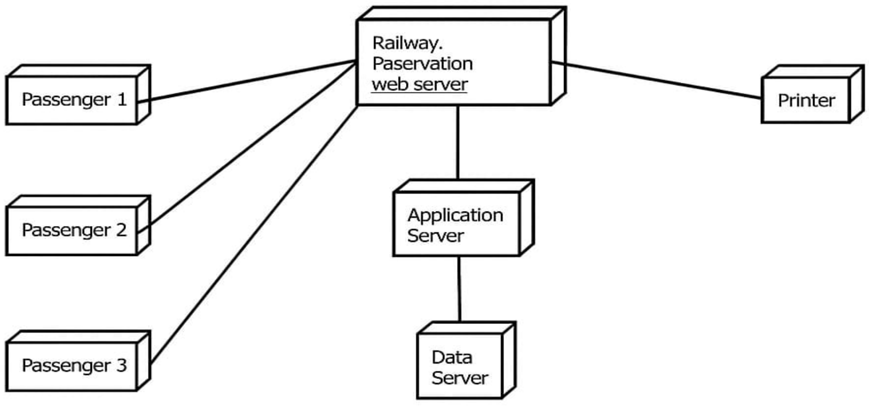

Admin is Registered
Admin Login ID and Password
Check Login Id
Invalid Login/password
Login to the system successfully
Set User level& Permissions
Access the internal Functionalities according to permissions

STATE CHART DIAGRAM FOR E TICKETING

Deployment Diagram:

A deployment diagram visualizes the physical deployment of software artifacts on nodes, such as servers or devices, within a system. Nodes are represented as three-dimensional boxes, while artifacts are shown as rectangles.

.


Railway.
Paservation
web server
Passenger 1
Passenger 2
Passenger 3
Printer
Application Server
Data Server