AIRLINES RESERVATION SYSTEM

PROJECT REPORT

18CSC202J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY

(2018 Regulation)

II Year/ III Semester

Academic Year: 2022 -2023

By

TUSHAR TANISHQ (RA2111003010682)

SACHIN GUPTA (RA2111003010684)

Under the guidance of

Ms. M. Suchithra

Assistant Professor

Department of Computing Technologies

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Tamil Nadu- 603203

NOVEMBER 2022

BONAFIDE

This is to certify that **18CSC202J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY project report** titled "**AIRLINES RESERVATION SYSTEM"** is the bonafide work of **TUSHAR TANISHQ (RA2111003010682) and SACHIN GUPTA (RA2111003010684)** who undertook the task of completing the project within the allotted time.

Signature of the Guide	**Signature of II Year Academic Advisor**
Ms.M. Suchithra	Dr. G. K. Sandhia
Assistant Professor	**Professor and Head**
Department of CTech	Department of CTech
SRM Institute of Science and Technology	SRM Institute of Science and Technology

About the course:-

18CSC202J - Object Oriented Design and Programming are 4 credit courses with **L-T-P-C as 3-0-2-4** (Tutorial modified as Practical from 2018 Curriculum onwards).

Objectives:

The student should be made to:

- Learn the basics of OOP concepts in C++
- Learn the basics of OOP analysis and design skills.
- Be exposed to the UML design diagrams.
- Be familiar with the various testing techniques

Course Learning Rationale (CLR): The purpose of learning this course is to:

1. Utilise class and build domain model for real-time programs
2. Utilise method overloading and operator overloading for real-time application development programs
3. Utilise inline, friend and virtual functions and create application development programs
4. Utilise exceptional handling and collections for real-time object-oriented programming applications
5. Construct UML component diagram and deployment diagram for design of applications
6. Create programs using object-oriented approach and design methodologies for real-time application development

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

1. Identify the class and build domain model
2. Construct programs using method overloading and operator overloading
3. Create programs using inline, friend and virtual functions, construct programs using standard templates
4. Construct programs using exceptional handling and collections
5. Create UML component diagram and deployment diagram
6. Create programs using object oriented approach and design methodologies

Table 1: Rubrics for Laboratory Exercises

(Internal Mark Splitup:- As per Curriculum)

CLAP-1	5=(2(E-lab Completion) + 2(Simple Exercises)(from CodeZinger, and any other coding platform) + 1(HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-2	7.5=(2.0(E-lab Completion)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-3	7.5=(2.0(E-lab Completion(80 Pgms)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	**2 Mark -** E-lab Completion **80 Program** Completion from 10 Session (Each session min 8 program) **2 Mark -** Code to UML conversion GCR Exercises **3.5 Mark - HackerRank** Coding challenge completion
CLAP-4	5= 3 (Model Practical) + 2(Oral Viva)	• **3 Mark** – Model Test • **2 Mark** – Oral Viva
Total	25	

COURSE ASSESSMENT PLAN FOR OODP LAB

S.No	List of Experiments	Course Learning Outcomes (CLO)	Blooms Level	PI	No of Programs in each session
1.	Implementation of I/O Operations in C++	CLO-1	Understand	2.8.1	10
2.	Implementation of Classes and Objects in C++	CLO-1	Apply	2.6.1	10
3,	To develop a problem statement. 1. From the problem statement, Identify Use Cases and develop the Use Case model. 2. From the problem statement, Identify the conceptual classes and develop a domain model with a UML Class diagram.	CLO-1	Analysis	4.6.1	Mini Project Given
4.	Implementation of Constructor Overloading and Method Overloading in C++	CLO-2	Apply	2.6.1	10
5.	Implementation of Operator Overloading in C++	CLO-2	Apply	2.6.1	10
6.	Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams and Collaboration diagrams	CLO-2	Analysis	4.6.1	Mini Project Given
7.	Implementation of Inheritance concepts in C++	CLO-3	Apply	2.6.1	10
8.	Implementation of Virtual function & interface concepts in C++	CLO-3	Apply	2.6.1	10
9.	Using the identified scenarios in your project, draw relevant state charts and activity diagrams.	CLO-3	Analysis	4.6.1	Mini Project Given
10.	Implementation of Templates in C++	CLO-3	Apply	2.6.1	10
11.	Implementation of Exception of Handling in C++	CLO-4	Apply	2.6.1	10
12.	Identify the User Interface, Domain objects, and Technical Services. Draw the partial layered, logical architecture diagram with UML package diagram notation such as Component Diagram, Deployment Diagram.	CLO-5	Analysis	4.6.1	Mini Project Given
13.	Implementation of STL Containers in C++	CLO-6	Apply	2.6.1	10
14.	Implementation of STL associative containers and algorithms in C++	CLO-6	Apply	2.6.1	10
15.	Implementation of Streams and File Handling in C++	CLO-6	Apply	2.6.1	10

LIST OF EXPERIMENTS FOR UML DESIGN AND MODELLING:

To develop a mini-project by following the exercises listed below.

1. To develop a problem statement.
2. Identify Use Cases and develop the Use Case model.
3. Identify the conceptual classes and develop a domain model with UML Class diagrams.
4. Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams.
5. Draw relevant statecharts and activity diagrams.
6. Identify the User Interface, Domain objects, and Technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.

Suggested Software Tools for UML:

StarUML, Rational Suite, Argo UML (or) equivalent, Eclipse IDE and Junit

ABSTRACT

Airline reservation systems (ARS) are systems that allow an airline to sell their inventory (seats). It is a complete flight booking quotation system which automates flight booking processes to help book flights online for particular seats available from various flights and increase revenue. It contains information on schedules and fares and contains a database of reservations (or passenger name records) and of tickets issued. ARSs are part of **passenger service systems (PSS)**, which are applications supporting the direct contact with the passenger.

ARS eventually evolved into the **computer reservations system (CRS)**. A computer reservation system is used for the reservations of a particular airline and interfaces with a **global distribution system (GDS)** which supports travel agencies and other distribution channels in making reservations for most major airlines in a single system.

Airline reservation systems incorporate airline schedules, fare tariffs, passenger reservations and ticket records. An airline's direct distribution works within their own reservation system, as well as pushing out information to the GDS. The second type of direct distribution channel are consumers who use the internet or mobile applications to make their own reservations. Travel agencies and other indirect distribution channels access the same GDS as those accessed by the airline reservation systems, and all messaging is transmitted by a standardised messaging system that functions on two types of messaging that transmit on a **high level network (HLN)**. These messaging types are called Type A for real time interactive communication and Type B for informational and booking type of messages. Message construction standards set by IATA and ICAO, are global, and apply to more than air transportation. Since airline reservation systems are business critical applications, and they are functionally quite complex, the operation of an in-house airline reservation system is relatively expensive.

MODULE DESCRIPTION

The **Unified Modelling Language (UML)** prescribes a standard set of diagrams and notations for modelling object oriented systems, and describes the underlying semantics of what these diagrams and symbols mean. Whereas there has been to this point many notations and methods used for object-oriented design, now there is a single notation for modellers to learn. UML can be used to model different kinds of systems: software systems, hardware systems and real-world organisations.

UML offers nine diagrams in which to model systems:

- **Use Case diagram** for modelling the business processes
- **Class diagram** for modelling the static structure of classes in the system
- **Sequence diagram** for modelling message passing between objects
- **Communication diagram** for modelling object interactions
- **State Chart diagram** for modelling the behaviour of objects in the system
- **Activity diagram** for modelling the behaviour of Use Cases, objects, or operations
- **Package diagram** for modelling the static structure of objects in the system
- **Component diagram** for modelling components
- **Deployment diagram** for modelling distribution of the system.

The following sections present a more detailed look at Modelling with UML. A very simple Airline Reservation System is used to illustrate UML Modelling techniques and diagrams.

The following topics are covered:

- Organising your system with packages.
- Modelling with Use Cases.
- Modelling with Sequence and Collaboration diagrams.
- Analysing and designing with the Class diagram.
- Modelling behaviour with State and Activity diagrams.
- Modelling software components, distribution, and implementation.

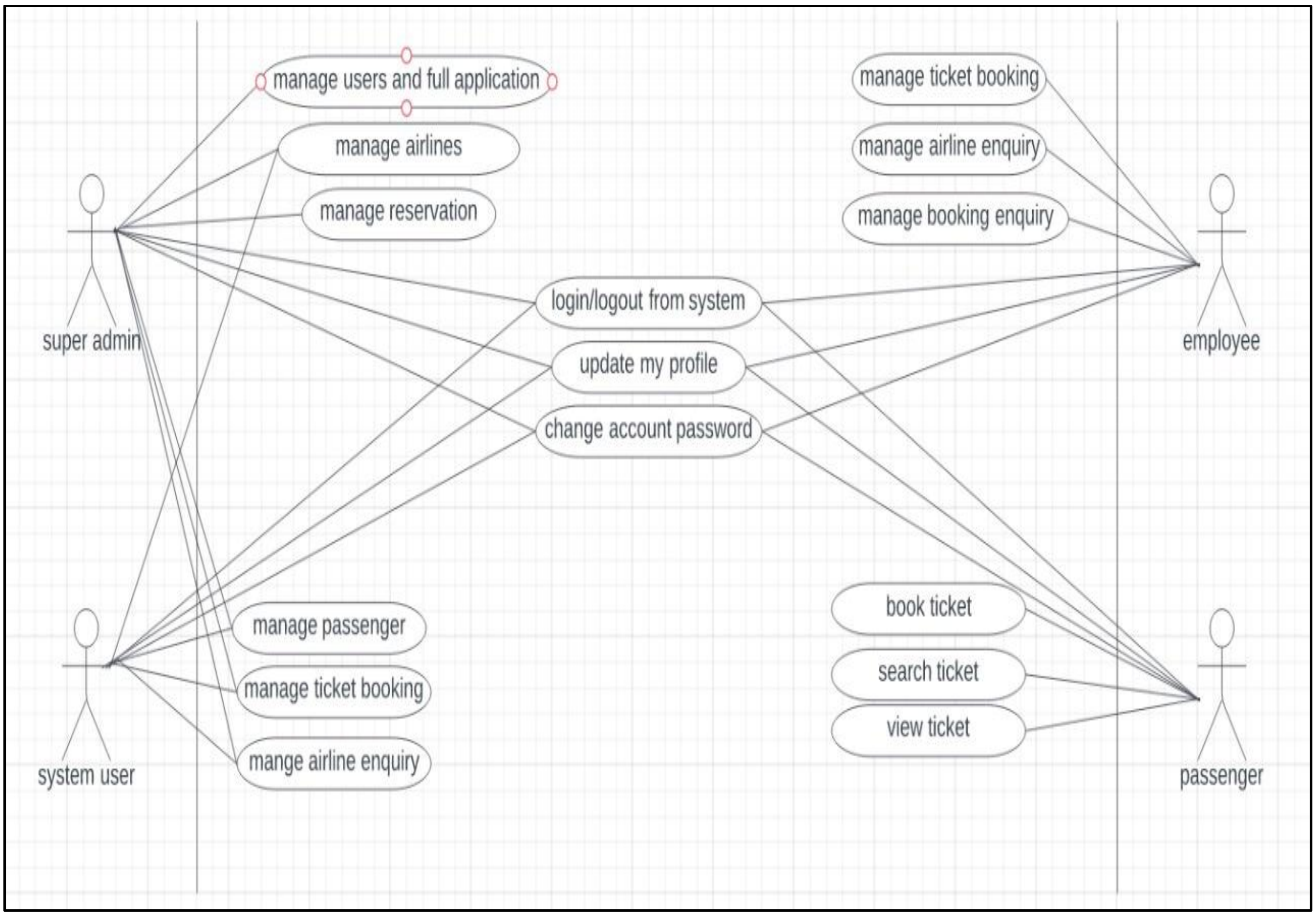
USE CASE DIAGRAM WITH EXPLANATION

The Use Case Diagram is a graphic depiction of the interactions among the elements of the Airlines Reservation System. It represents the methodology used in system analysis to identify, clarify, and organise system requirements of the Airlines Reservation System. The main actors of Airlines Reservation System in this Use Case Diagram are: Super Admin, System User, Employee, Passenger, who perform the different type of use cases such as Manage Airlines, Manage Reservation, Manage Passenger, Manage Ticket Booking, Manage Employee, Manage Airline Enquiry, Manage Users and Full Airlines Reservation System Operations. Major elements of the UML use case diagram of the Airlines Reservation System are shown in the picture below.

The relationships between and among the actors and the use cases of Airlines Reservation System:

- **Super Admin Entity:** Use cases of Super Admin are Manage Airlines, Manage Reservation, Manage Passenger, Manage Ticket Booking, Manage Employee, Manage Airline Enquiry, Manage Users and Full Airlines Reservation System Operations.
- **System User Entity:** Use cases of System User are Manage Airline, Manage Passenger, Manage Ticket Booking.
- **Employee Entity:** Use cases of Employee are Manage Ticket Booking, Manage Booking Enquiry, Manage Airline Enquiry, Manage Employee.
- **Passenger Entity:** Use cases of Passenger are Book ticket, search ticket, View Ticket, Login, Register.

Use Case Diagram of Airlines Reservation System:



CLASS DIAGRAM WITH EXPLANATION

Airlines Reservation System Class Diagram describes the structure of a Airlines Reservation System classes, their attributes, operations (or methods), and the relationships among objects. The main classes of the Airlines Reservation System are Airlines, Reservation, Passenger, Ticket Booking, Employee, Airline Enquiry.

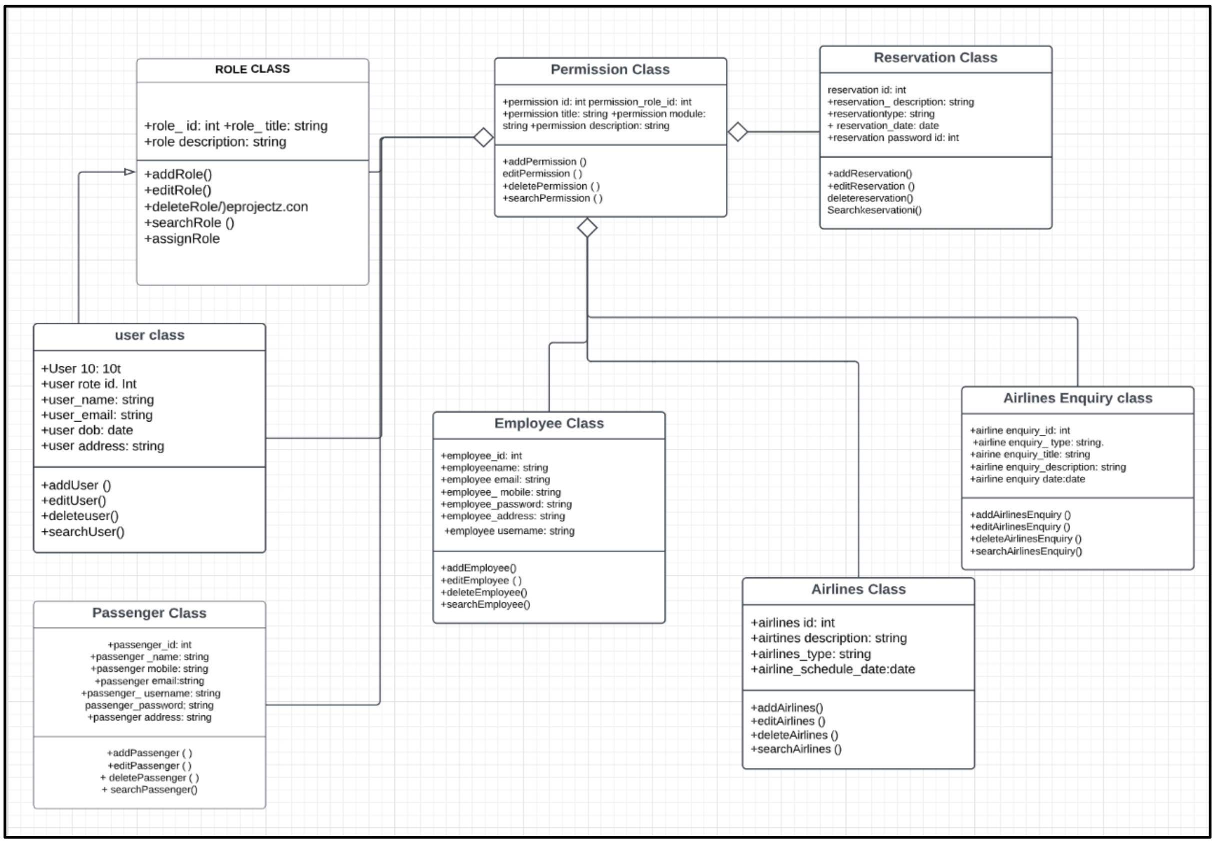
Classes of Airlines Reservation System Class Diagram:

- **Airlines Class:** Manage all the operations of Airlines.
- **Reservation Class:** Manage all the operations of Reservation.
- **Passenger Class:** Manage all the operations of Passenger.
- **Ticket Booking Class:** Manage all the operations of Ticket Booking.
- **Employee Class:** Manage all the operations of the Employee.
- **Airline Enquiry Class:** Manage all the operations of Airline Enquiry.

Classes and their methods of Airlines Reservation System Class Diagram:

- **Airlines Methods:** addAirlines), editAirlines(), delete Airlines(), updateAirlines), saveAirlines(), searchAirlines()
- **Reservation Methods:** addReservation(), editReservation(), deleteReservation(), updateReservation(), saveReservation(), searchReservation()
- **Passenger Methods:** addPassenger(), editPassenger(), deletePassenger(), updatePassenger(), savePassenger(), searchPassenger()
- **Ticket Booking Methods:** addTicket Booking(), editTicketBooking(), deleteTickeBooking(), updateTicketBooking(), saveTicketBooking(), searchTicketBooking()
- **Employee Methods:** addEmployee(), editEmployee(), deleteEmployee(), updateEmployee(), saveEmployee(), searchEmployee()
- **Airline Enquiry Methods:** addAirline Enquiry(), editAirline Enquiry(), deleteAirlineEnquiry(), updateAirlineEnquiry(), saveAirlineEnquiry(), searchAirlineEnquiry()

Class Diagram of Airlines Reservation System:



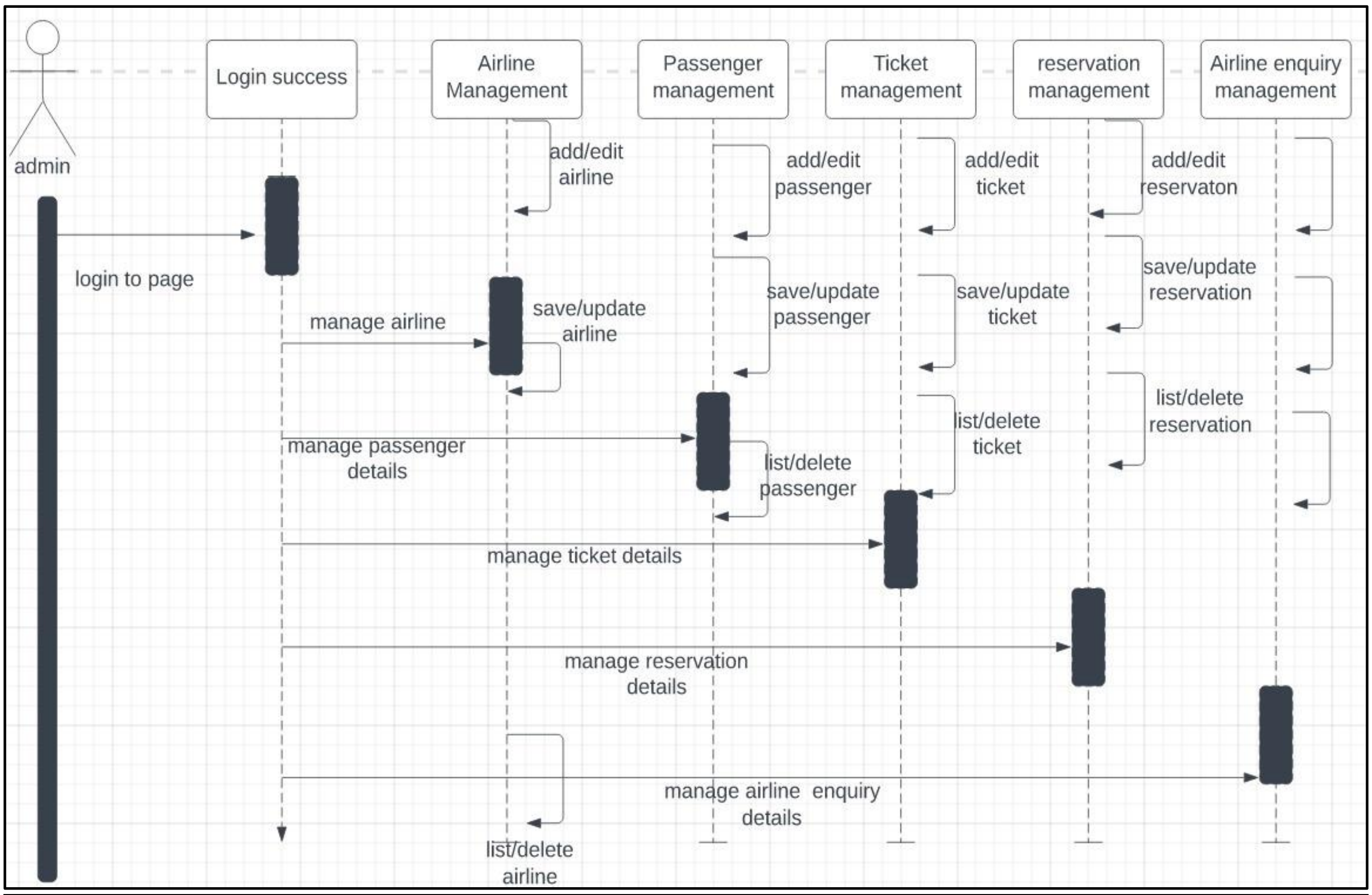
SEQUENCE DIAGRAM WITH EXPLANATION

The sequence diagram of the Airline Booking System shows the interaction between the objects of Booking Enquiry, Ticket Booking, Airline Enquiry, Airlines Booking, Passenger.

The instance of class objects involved in this UML Sequence Diagram of Airline Booking System are as follows:

- Ticket Booking Object
- Airlines Object
- Passenger Object
- Reservation Object
- Employee Object

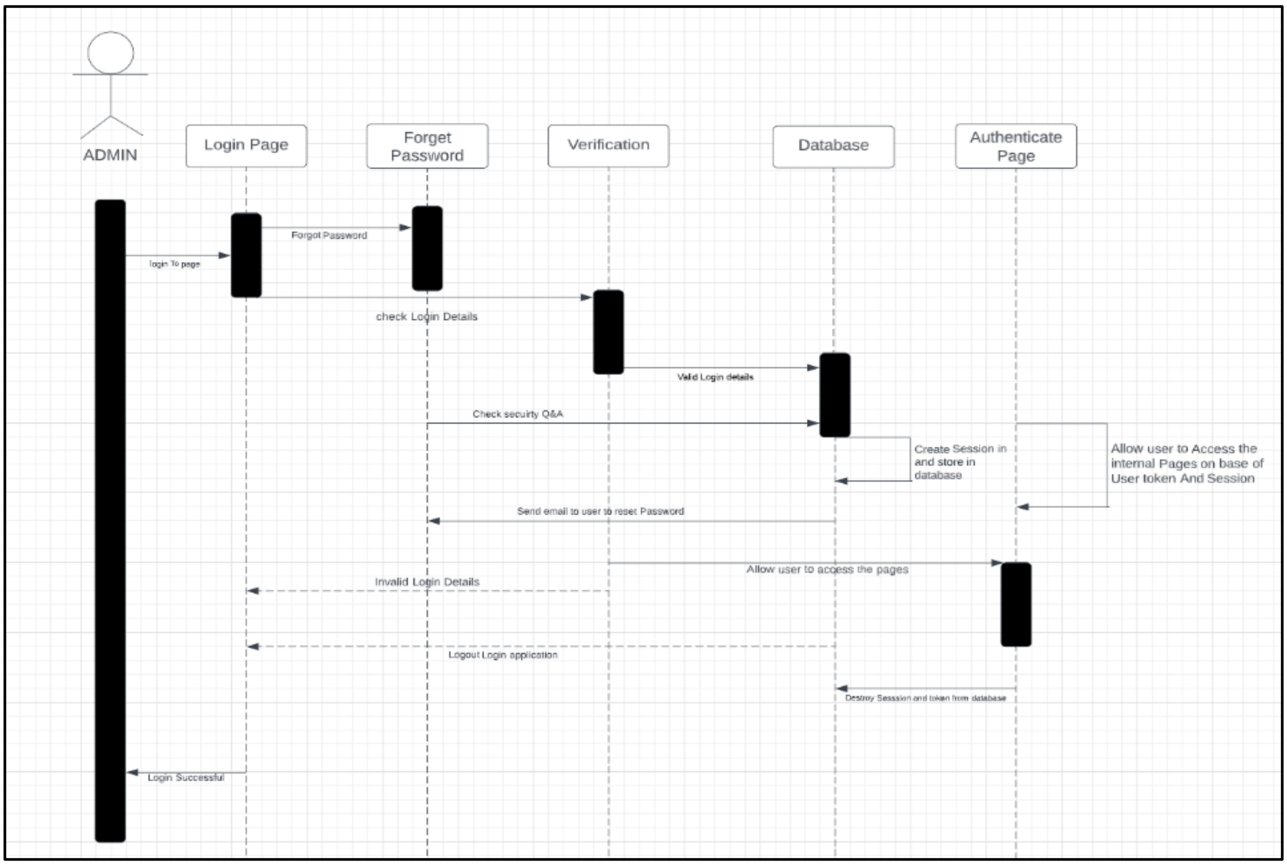
Sequence diagram of the Airline Reservation System:



Login Sequence Diagram Of Airline Booking System:

The Login Sequence Diagram of Airline Booking System is where admin will be able to login in their account using their credentials. After login, users can manage all the operations on Airline Enquiry, Booking Enquiry. Ticket Booking Passenger, Airlines Booking All the pages such as Ticket Booking, Passenger, Airlines Booking are secure and users can access these pages after login. The diagram below helps demonstrate how the login page works in an Airline Booking System. The various objects in the Passenger. Airline Enquiry Booking Enquiry, Ticket Booking, and Airlines Booking page Interact over the course of the sequence, and users will not be able to access this page without verifying their identity.

Login Sequence Diagram of Airline Booking System:


ADMIN
Login Page
Forget Password
Verification
Database
Authenticate Page
Forgot Password
login To page
check Login Details
Valid Login details
Check security Q&A
Create Session in and store in database
Allow user to Access the internal Pages on base of User token And Session
Send email to user to reset Password
Allow user to access the pages
Invalid Login Details
Logout Login application
Destroy Sesssion and token from database
Login Successful

COMMUNICATION DIAGRAM WITH EXPLANATION

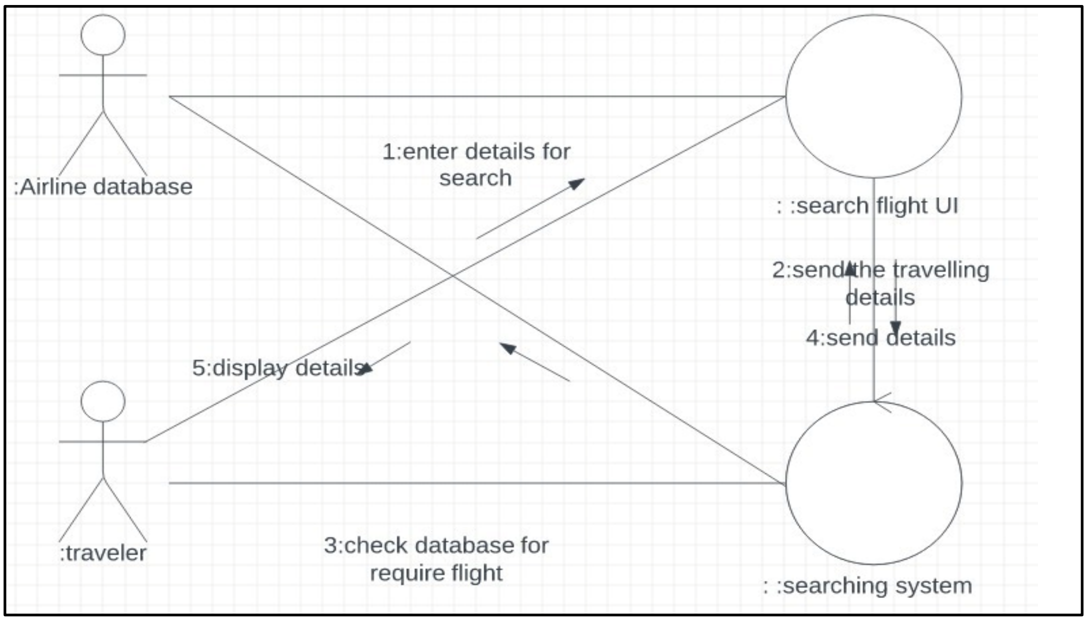
Communication diagrams, like the sequence diagrams, are a kind of interaction diagram, shows how objects interact. A communication diagram is an extension of an object diagram that shows the objects along with the messages that travel from one to another. In addition to the associations among objects, a communication diagram shows the messages the objects send each other. They convey the same information as sequence diagrams, but they focus on object roles instead of the times that messages are sent. In a sequence diagram, object roles are the vertices and messages are the connecting links.

The objects that interact in Communication Diagram of Airline Booking System are as follows:

- Ticket Booking Object
- Airlines Object
- Passenger Object
- Reservation Object
- Employee Object

The object-role rectangles are labelled with either class or object names (or both). Class names are preceded by colons (:). Each message in a collaboration diagram has a sequence number. The top-level message is numbered 1. Messages at the same level (sent during the same call) have the same decimal prefix but suffixes of 1, 2, etc. according to when they occur.

Communication Diagram of Airline Booking System:



STATE CHART DIAGRAM WITH EXPLANATION

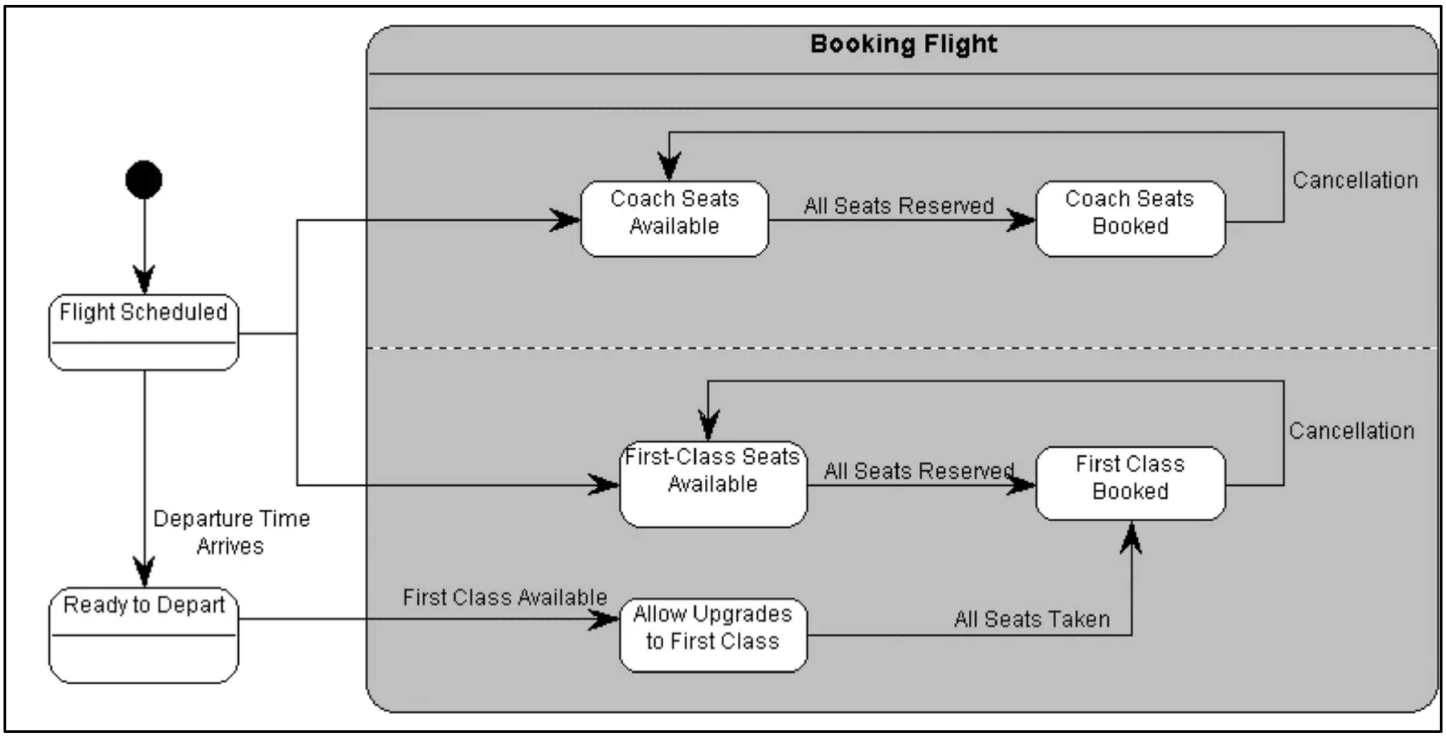
Statechart diagrams are used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

They describe the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of a Statechart diagram is to model the lifetime of an object from creation to termination.

Following are the main purposes of using Statechart diagrams:

- To model the dynamic aspect of a system.
- To model the lifetime of a reactive system.
- To describe different states of an object during its lifetime.
- Define a state machine to model the states of an object.

State Chart Diagram of Airline Booking System:



ACTIVITY DIAGRAM WITH EXPLANATION

The Activity UML diagram of Airlines Reservation System which shows the flows between the activity of Airlines, Passenger, Reservation, Airline Enquiry, Employee.

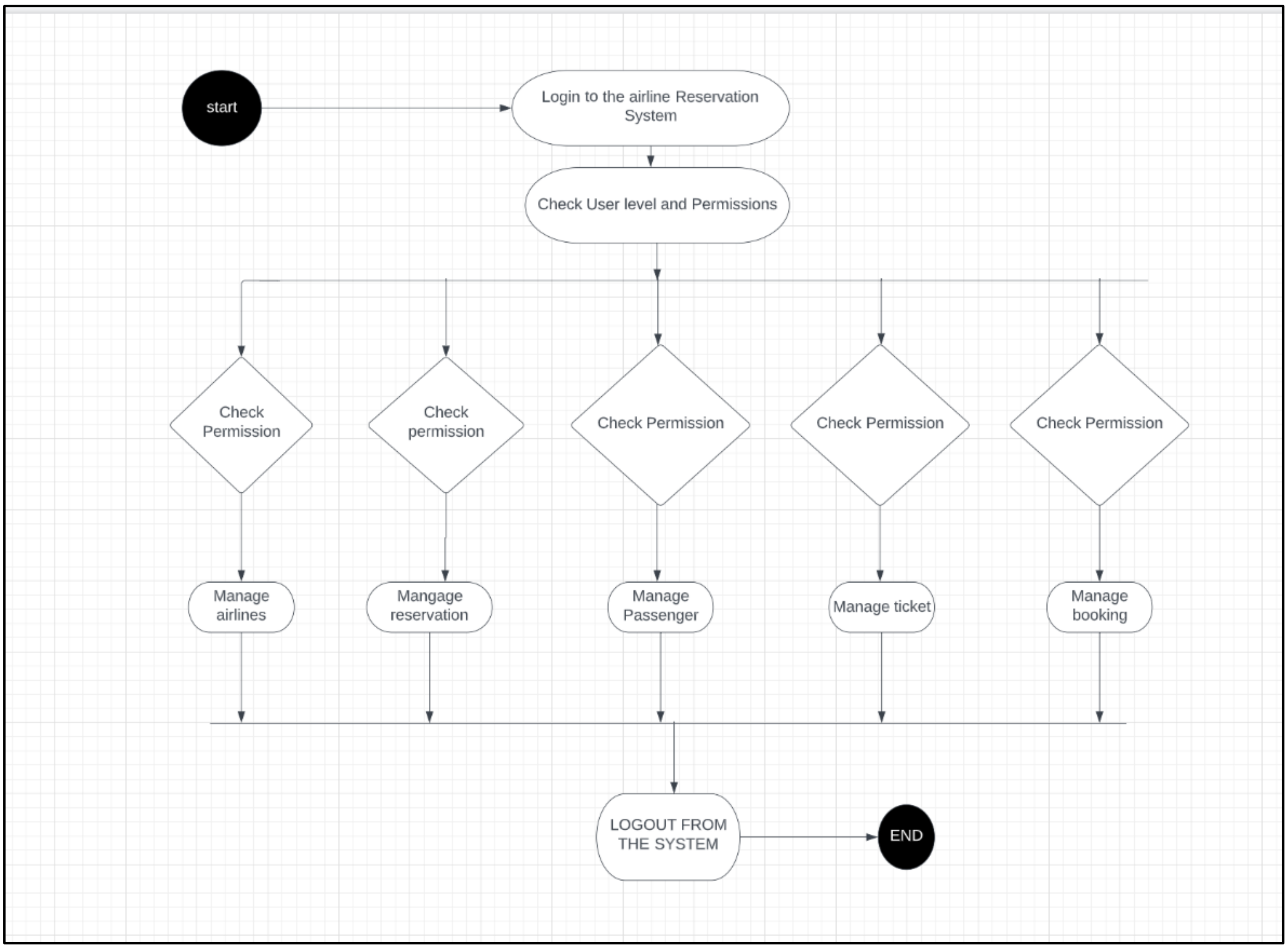
The main activity involved in this UML Activity Diagram of Airlines Reservation System are as follows:

- Airlines Activity
- Passenger Activity
- Reservation Activity
- Airline Enquiry Activity
- Employee Activity

Features Of The Activity UML Diagram Of Airlines Reservation System:

- Admin user can search Airlines, view descriptions of selected Airlines, add Airlines, update Airlines and delete Airlines.
- It shows the activity flow of editing, adding and updating of Passenger.
- User will be able to search and generate report of Reservation, Airline Enquiry, Employee
- All objects such as Airline, Passenger, Employee are interlinked
- It shows the full description and flow of Airlines, Airline Enquiry, Employee, Reservation, Passenger.

The Activity Diagram of Airlines Reservation System:



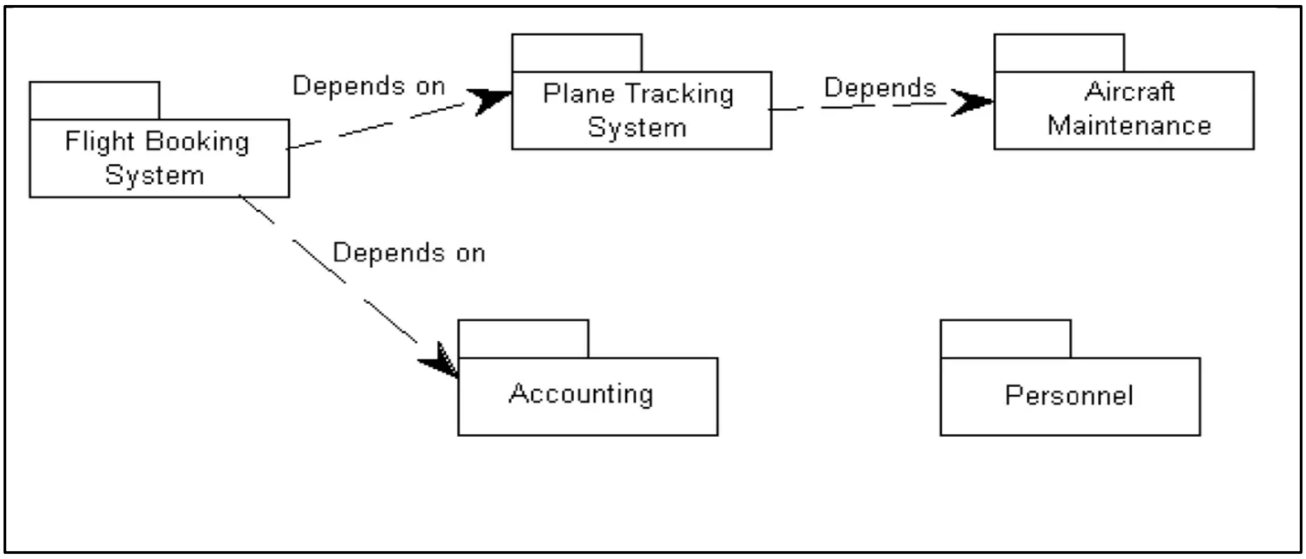
PACKAGE DIAGRAM WITH EXPLANATION

Package diagrams are structural diagrams used to show the organisation and arrangement of various model elements in the form of packages. A package is a grouping of related UML elements, such as diagrams, documents, classes, or even other packages. Each element is nested within the package, which is depicted as a file folder within the diagram, then arranged hierarchically within the diagram. Package diagrams are most commonly used to provide a visual organisation of the layered architecture within any UML classifier, such as a software system.

Benefits of a package diagram are as follows:

- A well-designed package diagram provides numerous benefits to those looking to create a visualisation of their UML system or project.
- They provide a clear view of the hierarchical structure of the various UML elements within a given system.
- These diagrams can simplify complex class diagrams into well-ordered visuals.
- They offer valuable high-level visibility into large-scale projects and systems.
- Package diagrams can be used to visually clarify a wide variety of projects and systems.
- These visuals can be easily updated assystems and projects evolve.

The Package Diagram of Airlines Reservation System:



COMPONENT DIAGRAM WITH EXPLANATION

This is a Component diagram of Airline Booking System which shows components, provided and required interfaces, ports, and relationships between the Booking Enquiry, Passenger Reservation, Airlines Booking.

Airline Enquiry and Ticket Booking This type of diagram is used in Component Based Development (CBD) to describe systems with Service-Oriented Architecture (SOA) Airline Booking System UML component diagram, describing the organisation and wiring of the physical components in a system.

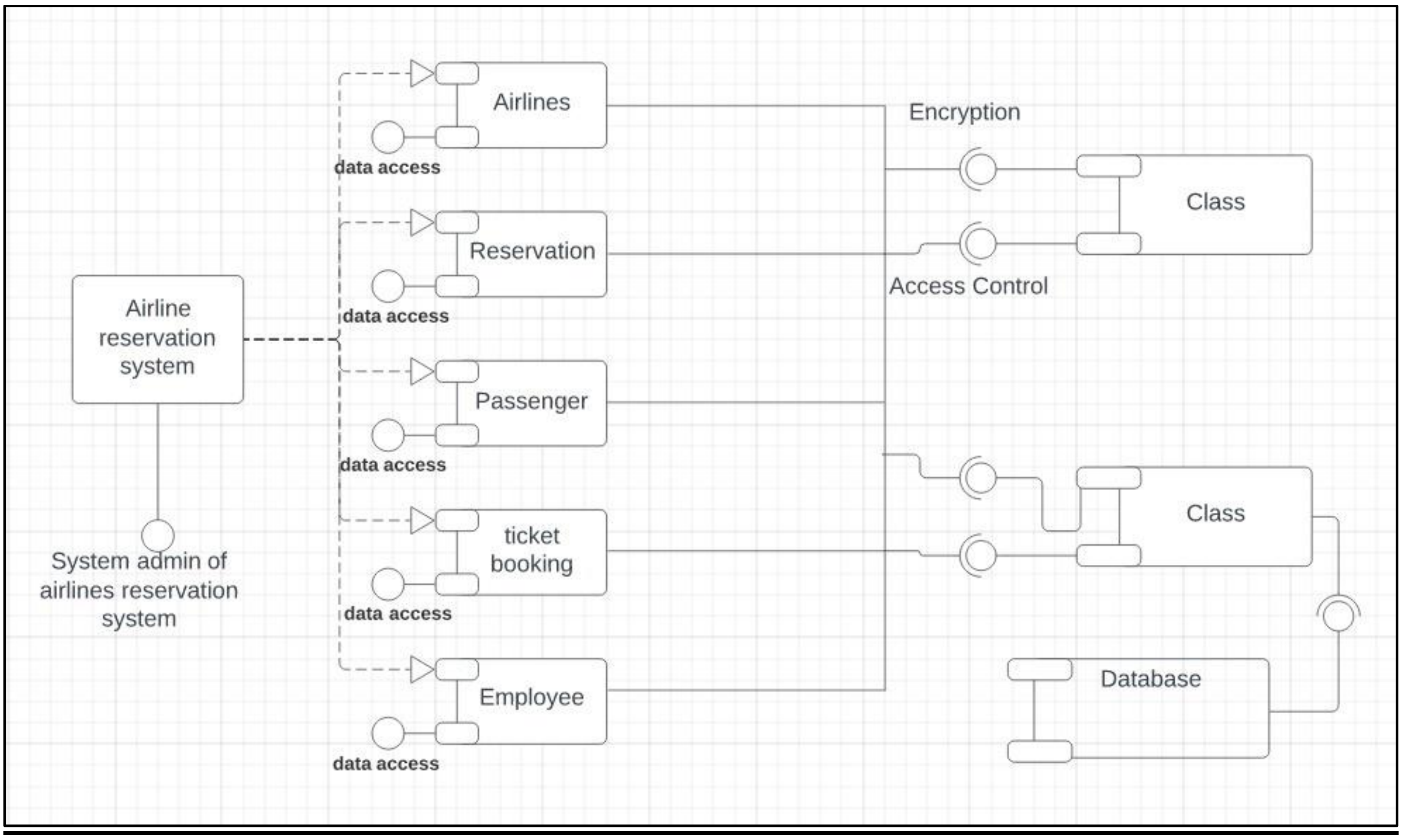
Components of UML Component Diagram of Airline Booking System:

- Booking Enquiry Component
- Passenger Reservation Component
- Airlines Booking Component
- Airline Enquiry Component
- Ticket Booking Component

Features of Airline Booking System Component Diagram:

- You can show the models the components of Airline Booking System
- Model the database schema of Airline Booking System
- Model the executables of an application of Airline Booking System
- Model the system's source code of Airline Booking System

The Component diagram of Airline Booking System:



DEPLOYMENT DIAGRAM WITH EXPLANATION

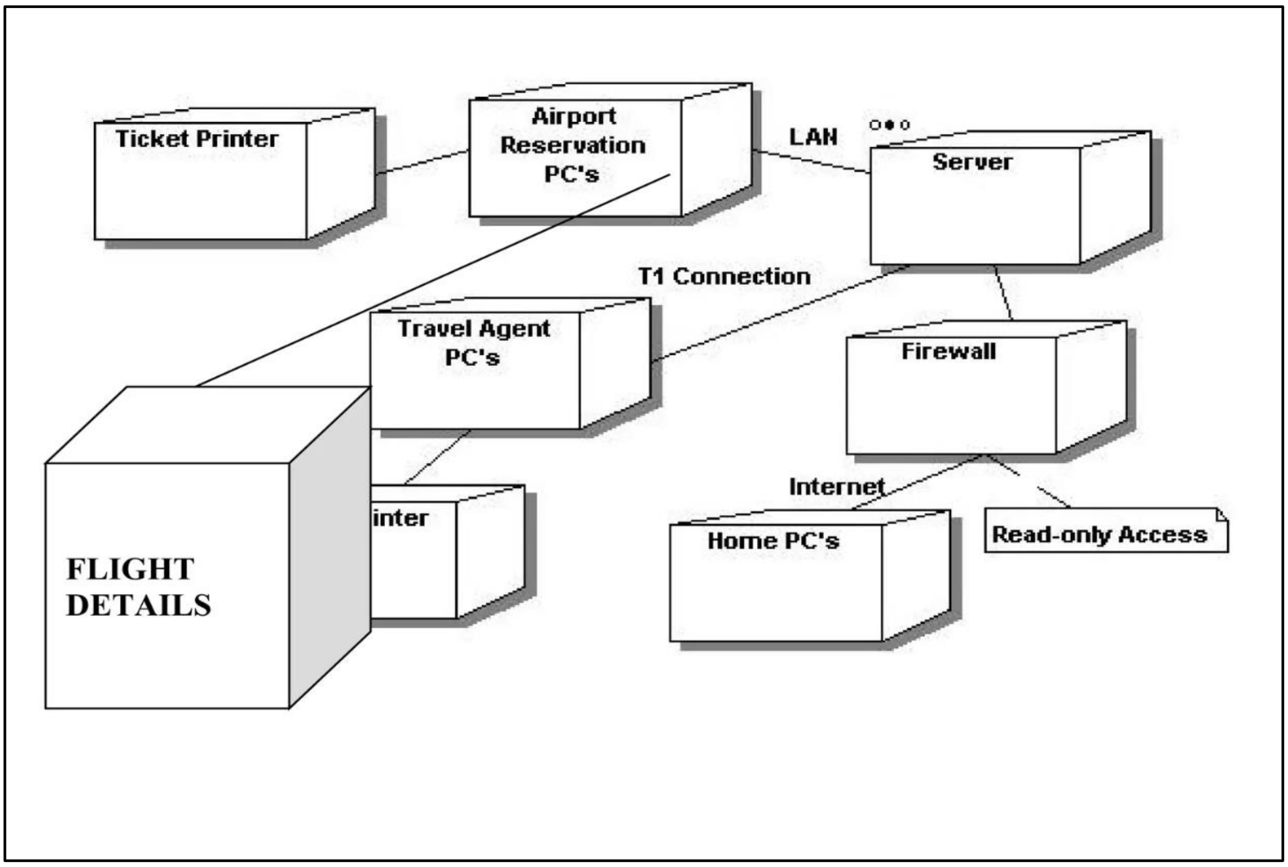
Deployment diagrams are used to model the configuration of run-time processing elements and the software components, processes, and objects that live on them. In the deployment diagram, you start by modelling the physical nodes and the communication associations that exist between them.

For each node, you can indicate what component instances live or run on the node. You can also model the objects that are contained within the component. Deployment diagrams are used to model only components that exist as run-time entities; they are not used to model compile-time only or link-time only components. You can also model components that migrate from node to node or objects that migrate from component to component using a dependency relationship with this becomes a stereotype.

Components used in our deployment diagram are as follows:

- Ticket Printer
- Airport Reservation PC's
- LAN
- Server
- Travel Agent PC's
- Firewall
- Printer
- Home PC's
- Read-only Access

The Deployment diagram of Airline Booking System:



CONCLUSION

UML diagrams can be used as a way to visualise a project before it takes place or as documentation for a project afterward. But the overall goal of UML diagrams is to allow teams to visualise how a project is or will be working, and they can be used in any field, not just software engineering. The diagrams will allow teams to visualise together how a system or process will work or did work. It can provide new ideas for how the team needs to collaborate to achieve the goal of the workflow process.

With the appropriate UML diagram tools, the development phase of designing/refining the application is shifted to the design and analysis phase. This helps in quick testing of the architecture system before beginning the coding. Many UML diagram tools help in generating skeleton code which proves to be object-oriented and efficient.

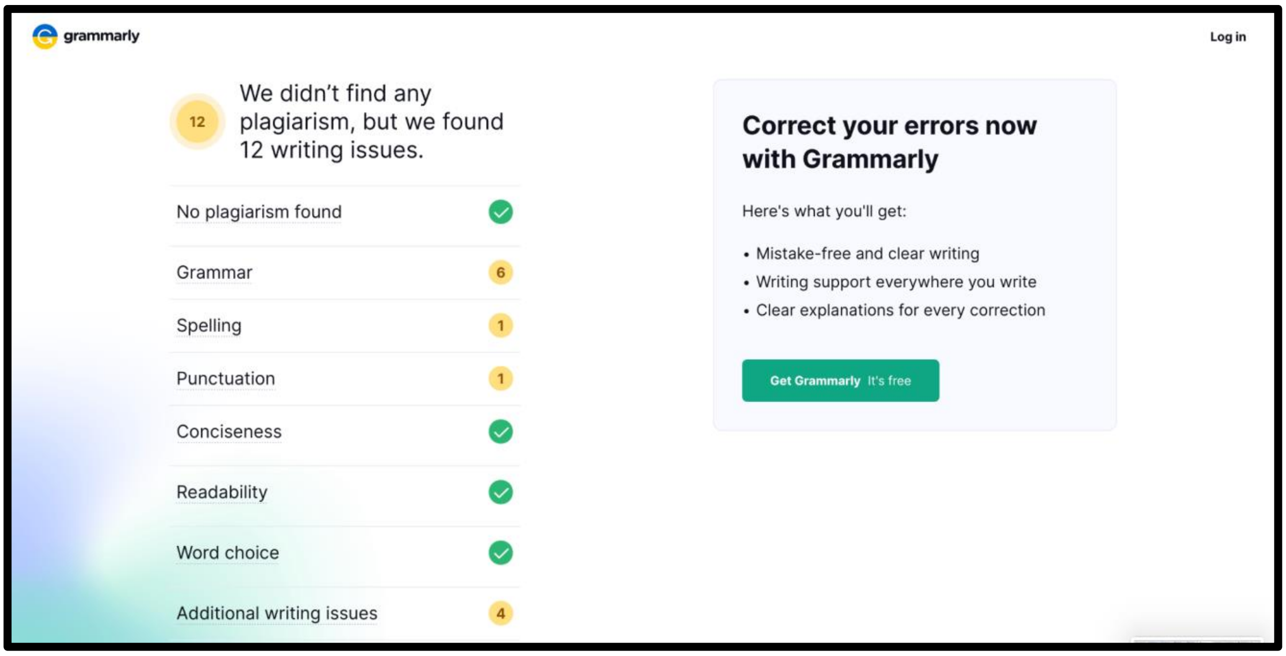
This visual language helps communicate the software architecture in detail to all users. It has a wide reach which is important for security assessing performance, providing essential guidelines, and tracking assignments.

By completing all the given Diagrams, the development of the Airline Reservation System as a complete software would be much easier to build and deploy.

REFERENCES

- https://www.wikipedia.org/
- https://lucid.co/
- Object-Oriented Analysis and Design with Applications by Grady Booch, Robert A. Maksimchuk, Michael W. Engle.

PLAGIARISM CHECK



12 We didn't find any plagiarism, but we found 12 writing issues.

No plagiarism found	✓
Grammar	6
Spelling	1
Punctuation	1
Conciseness	✓
Readability	✓
Word choice	✓
Additional writing issues	4

Correct your errors now with Grammarly

Here's what you'll get:

- Mistake-free and clear writing
- Writing support everywhere you write
- Clear explanations for every correction