JCEI'S JAIHIND COLLEGE ENGINEERING, KURAN

A.Y. 2024-25

MINI PROJECT ON

"AIRLINE SCHEDULING AND CARGO SCHEDULE''

UNDER

"ARTIFICIAL INTELLIGENCE [310253]."

TE-AI&DS ENGINEERING

DEPARTMENT OF AI & DS ENGINEERING

SEMESTER- V

SUBMITTED BY

Mr. Thorve Avishkar Shrikrushna

EXAM SEAT NO. : T1908402068

Department of Artificial Intelligence and Data Science Engineering

(2024-25)

Certificate

This is to certify that,

Mr. Thorve Avishkar Shrikrushna

EXAM SEAT NO. : T1908402068

have successfully completed the Mini project entitled **"AIRLINE SCHEDULING AND CARGO SCHEDULE"** under guidance of **Prof.Raut.S.P.** in partial fulfillment of the requirements for the Third Year of Engineering in Artificial intelligence and data science under the Savitribai Phule Pune University during the academic year 2024-2025.

Prof. Raut. S.P.
Subject teacher.

Prof.S.K.Said.
Head Of Department
AI&DS engg.

Dr.D.J.Garkal.
Principle
JCOE,kuran.

ACKNOWLEDGEMENT

With deep sense of gratitude we would like to thank all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work.

It is our proud privilege to express a deep sense of gratitude to **Dr.D.J.Garkal** Principal of Jaihind College Of Engineering, Kuran, for his comments and kind permission to complete this project. We remain indebted to **Prof.S.K.Said,** H.O.D. Artificial Intelligence And Data Science Engineering Department for his timely suggestion and valuable guidance.

The special gratitude goes **to Prof.Raut.S.P.** excellent and precious guidance in completion of this work .We thanks to all the colleagues for their appreciable help for our working project. With various industry owners or lab technicians to help, it has been our endeavor throughout our work to cover the entire project work.

We are also thankful to our parents who provided their wishful support for our project completion successfully .And lastly we thank our all friends and the people who are directly or indirectly related to our project work.

Yours faithfully,

Thorve Avishkar Shrikrushna

INDEX

ABSTRACT

The airline industry plays a crucial role in global transportation, with millions of passengers and vast quantities of cargo being transported daily. Efficient scheduling of flights and cargo operationsis paramount for airlines to meet customer demands while minimizing operational costs. This abstract provides an overview of the key concepts and challenges in optimizing airline schedulingand cargo schedules.

The efficient scheduling of airline flights and cargo operations is crucial in the aviation industry. Airline scheduling involves coordinating various elements, such as flight routes, crew assignments,and maintenance requirements, to meet passenger demands while adhering to regulatory constraints. Cargo scheduling focuses on optimizing the allocation of cargo space within aircraft, ground handling facilities, and transportation networks to ensure timely delivery. Combining these two schedules is essential for enhancing overall operational efficiency. Challenges in scheduling include managing disruptions and evolving customer expectations. The integration of real-time decision support systems, artificial intelligence, and machine learning will be key to addressing these challenges and optimizing airline and cargo schedules in the future.

1. INTRODUCTION

The airline industry is a critical component of global transportation, connecting people and goods across the world. Efficient airline scheduling and cargo schedules are fundamental to its smooth operation. Airline scheduling involves the intricate coordination of flights, crews, and resources, ensuring that passengers reach their destinations safely and on time. Cargo scheduling is equally important, enabling the timely and reliable transportation of goods, ranging from perishable commodities to industrial products. This introduction sets the stage for understanding the significance of these schedules, as well as the complex challenges and inter dependencies involved in optimizing them. In this context, the integration of passenger and cargo schedules emerges as a pivotal strategy for improving resource utilization, reducing costs, and enhancing customer satisfaction. To address these challenges and harness emerging opportunities, the aviation industry continues to explore innovative technologies and data-driven solutions. This paper delves into the multifaceted world of airline scheduling and cargo schedules, shedding light on the evolving landscape of aviation operations.

Objectives:

1. **Maximize Revenue**: Airlines aim to create flight schedules that align with peak travel times and destinations to maximize passenger loads and fare yields.
2. **Efficient Fleet Utilization**: Optimizing the use of the airline's fleet by minimizing idle time and ensuring aircraft are used as close to capacity as possible.
3. **Minimize Costs**: Reducing operating costs by selecting the most efficient routes, minimizing fuel consumption, optimizing crew schedules, and reducing turnaround times.
4. **Passenger Convenience**: Providing attractive flight times, connections, and frequencies to satisfy customer demand while maintaining competitive service offerings.
5. **Slot Management**: Ensuring that the airline can secure takeoff an d landing slots at congested airports, especially during peak hours.

Limitations:

1. **Airport Constraints**: Congested airports may have limited availability of slots, gates, or facilities, which restrict the number of flights that can be scheduled.
2. **Regulatory Restrictions**: International regulations, bilateral agreements, and air traffic rights

can limit scheduling, especially for international flights.

3. **Crew Scheduling Rules**: Labor agreements, safety regulations, and rest requirements for pilots and cabin crew impact when and how long a crew can work, affecting flight scheduling.
4. **Weather Conditions**: Weather-related delays, particularly in regions prone to storms or snow, can lead to unpredictable scheduling disruptions.
5. **Operational Constraints**: Technical maintenance requirements for aircraft, airport curfews, and runway availability can constrain the number and timing of flights.

2. SYSTEM REQUIREMENT

It requires Developing and implementing software solutions for airline scheduling and cargo schedules requires robust systems that can handle the complexity and real-time demands of the aviation industry.

2.1 Hardware requirements

- High-performance servers with multi-core processors.
- Sufficient RAM and fast storage (e.g., SSDs).
- Redundant hardware for fail-over.

2.2 Software requirements

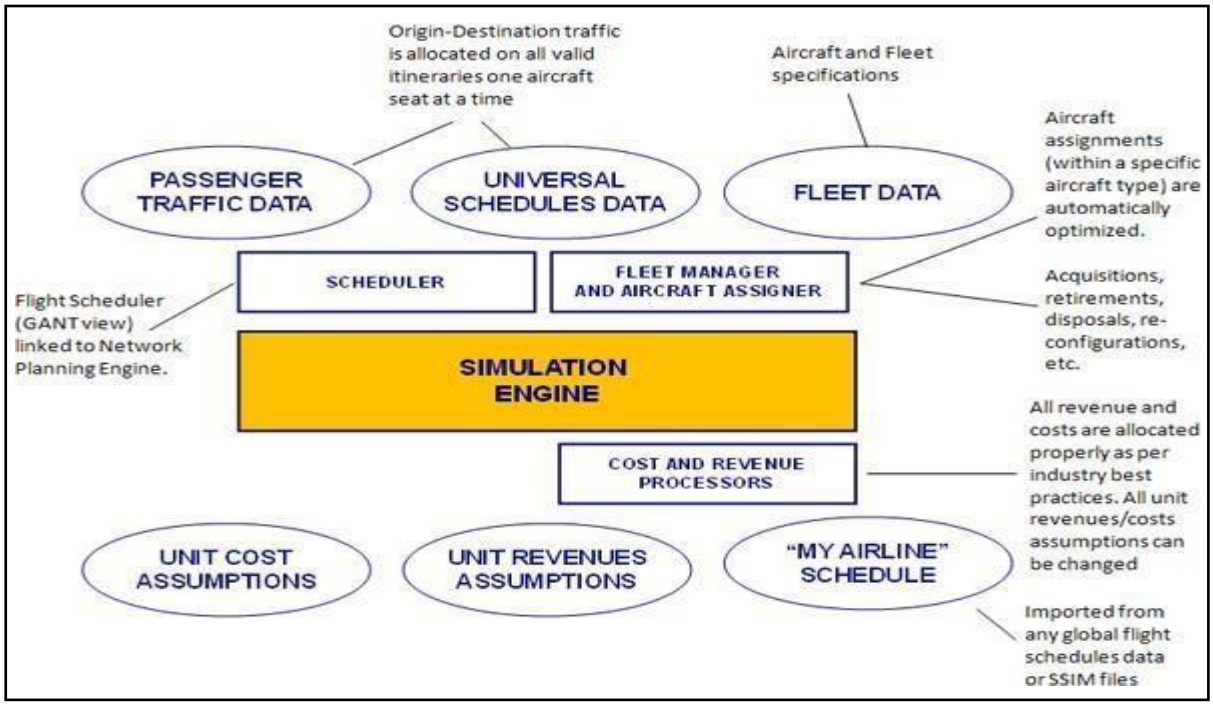
- Programming languages (Java, Python, C++, or C#).
- Integrated Development Environments (IDEs).
- Version control systems (e.g., Git).

3. DATABASE DESIGN

3.1 Conceptual Diagram:

3.1.1 ER Diagram

Designing an effective system for airline scheduling and cargo schedules involves careful consideration of numerous components and features to ensure operational efficiency, safety, and reliability. Here are key aspects of the system design in detail:



- **Modular Architecture:** The system should be designed with a modular architecture, allowing for the separation of key components such as flight scheduling, crew management, cargo allocation, and data storage. This modular approach enables easier maintenance, scalability, and the ability to upgrade individual components without affecting the entire system.
- **Real-Time Data Integration:** Integration with real-time data sources is essential. This includes interfaces to airline reservation systems, weather data providers, air traffic control systems, and cargo tracking services. These integration provide up-to-the-minute information that is vital for making informed scheduling decisions.
- **Optimization Algorithms:** The core of the system should employ advanced optimization algorithms. These algorithms factor in various parameters such as aircraft availability, crew

3.2 Implementation:

Implementing a system for airline scheduling and cargo schedules is a multifaceted process that demands meticulous planning and execution to ensure the seamless operation of the aviation industry.

- **Software Development:** Software development is at the heart of the implementation process. This stage involves the creation or configuration of scheduling software that adheres to the system design and requirements. The software should integrate modular components, including real-time data sources, optimization algorithms, and user interfaces, all tailored to meet the aviation industry's specific needs.

- **Main Source Code:**

```
class Flight:
  def__init_(self, flight_number, origin, destination, departure_time, arrival_time,
    cargo_capacity):self.flight_number = flight_number
    self.origin = origin
    self.destination =
    destination
    self.departure_time =
    departure_time
    self.arrival_time =
    arrival_time
    self.cargo_capacity = cargo_capacity  # in
    kilogramsself.cargo_assigned = 0
    self.cargo_list = []

  def assign_cargo(self, cargo):
    if self.cargo_assigned + cargo.weight <=
      self.cargo_capacity:
      self.cargo_list.append(cargo)
      self.cargo_assigned += cargo.weight
      print(f"Cargo {cargo.cargo_id} assigned to flight
    {self.flight_number}.")else:
      print(f"Not enough capacity for cargo {cargo.cargo_id} on flight {self.flight_number}.")

  def__str_(self):
    return (f"Flight {self.flight_number} from {self.origin} to
        {self.destination}.\n"f"Departure: {self.departure_time},
        Arrival: {self.arrival_time}\n"
        f"Cargo Capacity: {self.cargo_capacity} kg, Cargo Assigned: {self.cargo_assigned} kg")
```

```
class Cargo:
   def__init_(self, cargo_id, weight,
      destination):self.cargo_id = cargo_id
      self.weight = weight # in
      kilogramsself.destination =
      destination

   def__str_(self):
      return f"Cargo {self.cargo_id}: Weight {self.weight} kg, Destination: {self.destination}"

   def__init_(self):
      self.flights = []

   def add_flight(self,
      flight):
   self.flights.append(
      flight)
      print(f"Flight {flight.flight_number} added to schedule.")
    def assign_cargo_to_flight(self, cargo):
      # Find a flight going to the cargo's
      destinationfor flight in self.flights:
         if flight.destination ==
            cargo.destination:
           flight.assign_cargo(cargo)
           return
      print(f"No available flight to {cargo.destination} for cargo {cargo.cargo_id}.")

   def
      display_schedule(se
      lf):for flight in
      self.flights:
          print(flight)
          print("Cargo on this
          flight:")for cargo in
          flight.cargo_list:
             print(f"  {cargo}")
          print("-" * 40)
-# Function to get flight details
fromthe userdef create_flight():
   flight_number = input("Enter flight
   number: ")origin = input("Enter flight
   origin: ") destination = input("Enter
   flight destination: ")
   departure_time = input("Enter departure time (e.g.,
   08:00): ")arrival_time = input("Enter arrival time
   (e.g., 11:00): ") cargo_capacity = int(input("Enter
   cargo capacity (in kg): "))
```

```
    return Flight(flight_number, origin, destination, departure_time, arrival_time, cargo_capacity)

# Function to get cargo details from
the userdef create_cargo():
    cargo_id = input("Enter cargo ID: ")
    weight = int(input("Enter cargo weight
    (in kg): "))destination = input("Enter
    cargo destination: ") return
    Cargo(cargo_id, weight, destination)

# Main interactive
programdef main():
    schedule = Schedule()

    while True:
        print("\nAirline Scheduling
```

❖ Front End (Javascript):

```
import React, { useState, useEffect } from 'react';
import FlightList from './components/FlightList';
import CargoList from './components/CargoList';
import ScheduleForm from './components/ScheduleForm';

function App() {
    const [flights, setFlights] = useState([]);
    const [cargo, setCargo] = useState([]);

    useEffect(() => {
        // Fetch initial flight and cargo data (API integration)
        fetchFlights();
        fetchCargo();
    }, []);

    const fetchFlights = async () => {
        // Fetch flights from API
        const response = await fetch('http://localhost:5000/api/flights');
        const data = await response.json();
        setFlights(data);
    };

    const fetchCargo = async () => {
        // Fetch cargo from API
        const response = await fetch('http://localhost:5000/api/cargo');
        const data = await response.json();
        setCargo(data);
    };
```

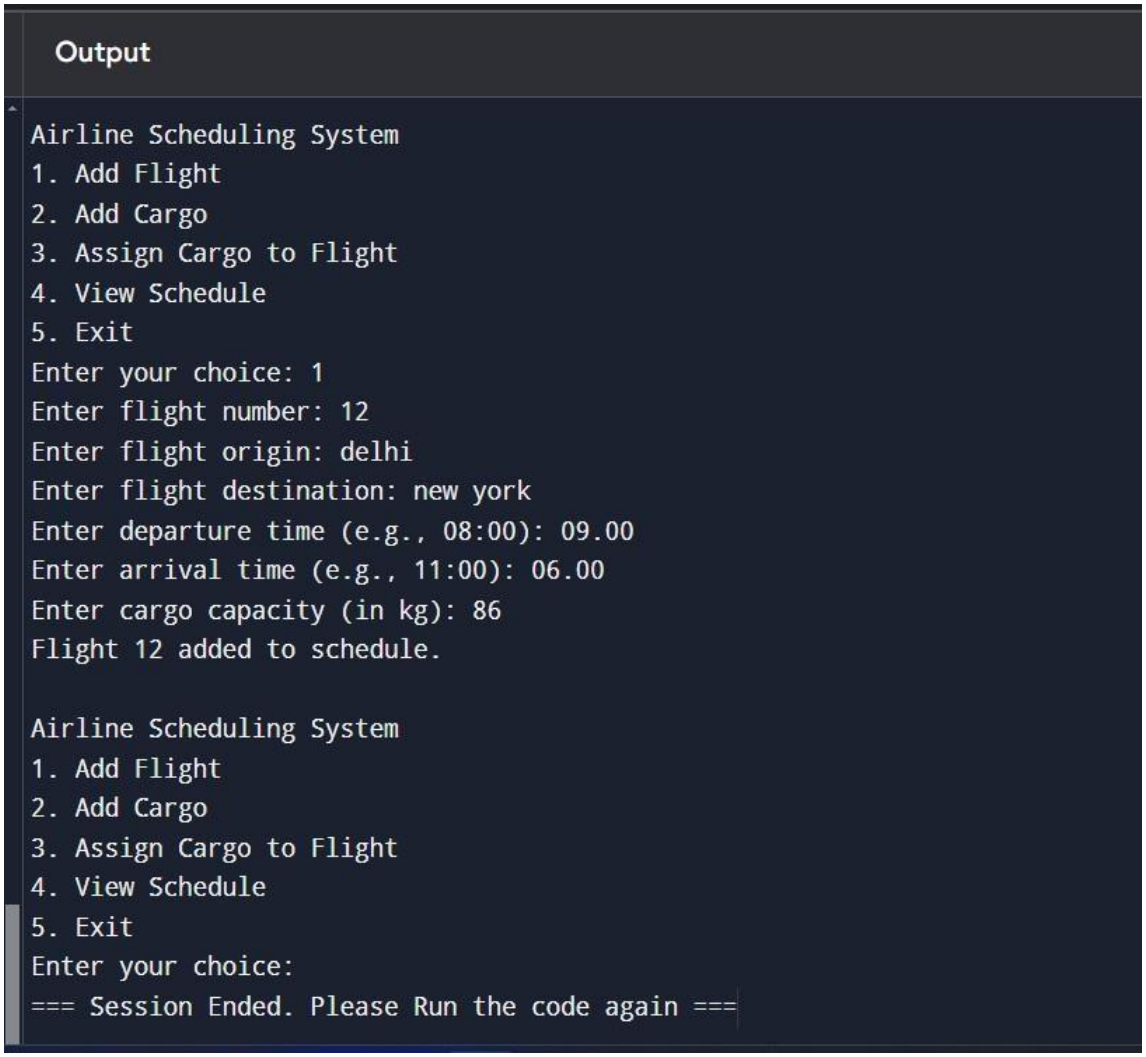
```
  return (
    <div className="App">
      <h1>Airline Scheduling and Cargo Management</h1>
      <ScheduleForm fetchFlights={fetchFlights} fetchCargo={fetchCargo} />
      <FlightList flights={flights} />
      <CargoList cargo={cargo} />
```

❖ Back End (Javascript):

```
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');
const mongoose = require('mongoose');

const app = express();

// Middleware
app.use(cors());
app.use(bodyParser.json());

// Connect to MongoDB (example)
mongoose.connect('mongodb://localhost:27017/airline', { useNewUrlParser: true, useUnifiedTopology: true });

// Flight Schema
const flightSchema = new mongoose.Schema({
  flightNumber: String,
  departure: String,
  arrival: String,
  time: String,
});
const Flight = mongoose.model('Flight', flightSchema);
// Cargo Schema
const cargoSchema = new mongoose.Schema({
  cargoType: String,
  weight: Number,
  flightNumber: String,
});
const Cargo = mongoose.model('Cargo', cargoSchema);
// Routes
// Get Flights
app.get('/api/flights', async (req, res) => {
  const flights = await Flight.find();
  res.json(flights);
});
// Get Cargo
app.get('/api/cargo', async (req, res) => {
  const cargo = await Cargo.find();  res.json(cargo);
```

```
});
// Post Schedule (Flight and Cargo)
app.post('/api/schedule', async (req, res) => {
    const { flightNumber, departure, arrival, cargoType, weight } = req.body;
    const flight = new Flight({ flightNumber, departure, arrival, time: new Date().toISOString() });
    await flight.save();
    if (cargoType && weight) {
        const cargo = new Cargo({ cargoType, weight, flightNumber });
        await cargo.save();
    }
    res.send('Schedule Created');
});
// Start Server
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`Server running on port ${PORT}`))
```

❖ **Output:**


Output
Airline Scheduling System
1. Add Flight
2. Add Cargo
3. Assign Cargo to Flight
4. View Schedule
5. Exit
Enter your choice: 1
Enter flight number: 12
Enter flight origin: delhi
Enter flight destination: new york
Enter departure time (e.g., 08:00): 09.00
Enter arrival time (e.g., 11:00): 06.00
Enter cargo capacity (in kg): 86
Flight 12 added to schedule.
Airline Scheduling System
1. Add Flight
2. Add Cargo
3. Assign Cargo to Flight
4. View Schedule
5. Exit
Enter your choice:
=== Session Ended. Please Run the code again ===

4. INTERFACES

Airline scheduling and cargo scheduling are interconnected processes that require coordination between different operational departments within an airline. The interfaces between passenger airline scheduling and cargo schedules ensure that both passenger and cargo operations optimize the use of available resources like aircraft, crew, and airport slots. Here are some key interfaces between the two:

1. Aircraft Utilization Interface

- Passenger and cargo schedules share the same aircraft, especially in mixed-use flights (e.g., **belly cargo** on passenger planes).
- Cargo operations need to coordinate with passenger scheduling to ensure that there's enough **belly capacity** on the flight, especially during high-demand seasons.
- **Freighter schedules** (dedicated cargo planes) must fit within broader airline scheduling constraints such as **aircraft maintenance**, crew availability, and airport slot restrictions.

2. Crew Scheduling Interface

- Both passenger and cargo flights require optimized crew assignments.
- In mixed-fleet operations (passenger and freighter aircraft), the scheduling of crew for both services has to be synchronized to avoid crew shortages and ensure adherence to regulatory rest times.

3. Airport Slot Management

- Airlines must negotiate for **airport slots** where both passenger and cargo schedules arealigned, especially at busy airports.
- Cargo operations often need late-night or off-peak slots to minimize conflicts with passenger traffic, but sometimes both types of flights overlap in terms of required slots.

4. Fleet Assignment and Routing Interface

- Cargo and passenger flights are integrated into the overall **fleet assignment** plan to maximize aircraft usage.
- In some cases, the need to prioritize high-value cargo shipments (such as medical supplies) may lead to changes in a passenger airline's schedule to accommodate certain routes or capacities.

5. Network Design and Optimization

- Airlines optimize their networks to balance both passenger demand and cargo demand. This means developing schedules where certain aircraft can serve dual purposes on the same route.
- **Hub-and-spoke models** for passengers can be adapted for cargo distribution, allowing efficient transfer of goods at central hub airports.

6. Revenue Management

- Airlines use **dynamic pricing** for both seats and cargo space, so real-time data on cargo load factors must be fed into the scheduling system.
- This interface ensures that cargo capacities are effectively sold without impacting the availability of seats for passengers.

7. Regulatory and Safety Requirements

- Passenger and cargo scheduling must comply with **aviation safety regulations** (e.g., weight balance on mixed flights, hazardous material transport).
- Cargo schedules have specific regulatory interfaces for handling dangerous goods, live animals, or perishables, which influence the flight planning and timing.

8. IT Systems and Scheduling Tools

- Airlines use integrated scheduling systems like **Sabre**, **Amadeus**, or customized tools for fleet management and crew scheduling, which help align both cargo and passenger operations.

5. CONCLUSION

In conclusion, airline scheduling and cargo schedules play a pivotal role in the aviation industry, ensuring the efficient and safe movement of passengers and goods across the globe. These complex processes are intertwined, and their effective coordination is essential for optimizing resource utilization, reducing costs, and enhancing customer satisfaction. Airline scheduling involves the intricate allocation of aircraft, crew, and resources to meet passenger demands while adhering to stringent safety and regulatory requirements. Optimizationtechniques, such as mathematical programming and advanced algorithms, enable airlines to createschedules that maximize operational efficiency and revenue while minimizing delays and disruptions.

6. FUTURE SCOPE

1. Airline scheduling and cargo schedules is characterized by the adoption of advanced technologies like AI and block chain, a focus on sustainability, dynamic scheduling, and increased personalization of passenger experiences.
2. Emerging markets, regulatory compliance, and the potential integration of space travel and drones are also key areas of development in the aviation industry's scheduling processes.
3. Blockchain for Data Security: Explore the use of blockchain technology to enhance data security and integrity.
4. Blockchain can help ensure the immutability of medical records, secure data sharing, and maintain a transparent and auditable trail of healthcare transactions.

7. REFERENCES

i. "Airline Operations and Scheduling" by Massoud Bazargan – Focuses on flight, crew, and fleet scheduling.
ii. "The Global Airline Industry" by Peter Belobaba et al. – Covers airline economics and scheduling, including cargo.
iii. Barnhart, C., & Smith, B. (1995). "Quantitative Models of Airline Schedule Planning: Fleet Assignment, Aircraft Routing, and Crew Scheduling."