BASAVARAJESWARI GROUP OF INSTITUTIONS

**BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT**

Autonomous institute under VISVESVARAYA TECHNOLOGICAL UNIVERSITYJNANA SANGAMA,

BELAGAVI 590018

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)

"JnanaGangotri"Campus,No.873/2,Ballari-HospetRoad,Allipur,
Ballar1-583 104 (Karnataka)(India)

Ph: 08392 – 237100 / 237190, Fax: 08392 –237197

**INTERNSHIP**

**Report On**

# Flight Booking simulation

Submitted in partial fulfillment of the requirements for the award of degree of

## Bachelor of Engineering

**In**

## ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

**Submitted by**

| | |
|---|---|
| Ajith B | 3BR23AI005 |
| Ayush Shetty | 3BR23AI016 |
| Ayushman Hansraj | 3BR23AI017 |
| Charan Raj S | 3BR23AI033 |
| Darshan Bandi | 3BR23AI038 |

**External Guide**
**Ms. Neha**

**BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT**

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)

"Jnana Gangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballar1-583 104 (Karnataka) (India)

Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

**2025-2026**

**BASAVARAJESWARI GROUP OF INSTITUTIONS**

# BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

**Autonomous institute under VISVESVARAYA TECHNOLOGICAL UNIVERSITYJNANA SANGAMA,**

**BELAGAVI 590018**

NACC Accredited Institution*

**(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)**

**"JnanaGangotri"Campus,No.873/2,Ballari-HospetRoad,Allipur, Ballar1-583 104 (Karnataka)(India)**

**Ph: 08392 – 237100 / 237190, Fax: 08392 –237197**

## DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

# CERTIFICATE

This is to certify that the Internship entitled "FLIGHT BOOKING SIMULATION " hasbeen successfully completed by **Ajith B, Ayush Shetty, Ayushmaan Hansraj singh, Charan Raj, Darshan Bandi** bearing USN **3BR23AI005, 3BR23AI016, 3BR23AI017, 3BR23AI033, 3BR23AI038** a bonafide student of Ballari Institute of Technology and Management, Ballari. For the partial fulfillment of the requirements for the **Bachelor's Degree in Artificial Intelligence and Machine Learning** of the VISVESVARAYA TECHNOLOGICAL UNIVERSITY, Belagavi during the academic year 2025-2026.

**Signature of HOD**

**Dr. B.M. Vidyavathi**
**Prof. and HOD(AIML)**

# DECLARATION

I, **Ajith B, Ayush Shetty, Ayushmaan Hansraj singh, Charan Raj, Darshan Bandi** sTH year student of Computer Science and Engineering, Ballari Institute of Technology, Ballari, declare that Internship entitled **FLIGHT BOOKING SIMULATION** is a part of Internship Training successfully carried out by **EZ TECHNOLOGIES & TRAININGS PVT.LTD, Hyderabad** at "**BITM, BALLARI".** This report is submitted in partial fulfillment of the requirements for the award of the degree, Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi.

**Date :28-9-2025** **Signature of the Student**

**Place :Ballari**

# ACKNOWLEDGEMENT

The satisfactions that a company the successful completion of my internship on " **Python Internship program** " would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is my privilege to express my gratitude and respect to all those who inspired me in the completion of my internship.

I am grateful to our respective coordinator for his noble gesture, support co-ordination and valuable suggestions given to me inthe completion of Internship.

I also thank **Dr.B.M.Vidyavathi ,** H.O.D. Department **Artificial Intelligence and**

**Machine Learning** forextending all his valuable support and encouragement.

# Table of Contents

CHAPTER – 1

# Introduction

This project presents a comprehensive analysis of a Python-based Flight Booking Simulation implemented using Flask and JSON data handling. The application simulates a real-world food ordering platform, providing functionalities such as menu display, cart management, checkout, and order storage. By leveraging Flask for routing and Python for backend logic, the system ensures smooth user interaction and dynamic content rendering.

The application architecture follows object-oriented programming principles, separating key functionalities into modules such as add flight, add passengers, and booking a seat. JSON files are used for lightweight and flexible data storage, ensuring that menu updates and order histories can be maintained without complex databases.

This design approach not only promotes scalability and maintainability but also bridges both terminal-based and web-based versions of the project. The terminal version demonstrates pure Python logic, while the web version enhances the user experience with a simple yet effective frontend interface.

**CHAPTER – 2**

# Objective

The primary objective of this project is to develop and evaluate a **Python-based Flight Booking Simulation** that demonstrates strong backend logic and practical implementation of core programming concepts. The system aims to provide essential functionalities such as Add Flight, Remove Flight, Book Seat, Cancel Booking, Passanger List

This assessment focuses on:

- Examining the **code organization** and object-oriented design principles.
- Identifying **key features and capabilities** like dynamic JSON data handling, session-based cart storage, and checkout flow.
- Evaluating the **user experience** across both terminal-based and web-based versions.
- Assessing the **technical implementation decisions** including Flask routing, JSON integration, and modular design.

The analysis ultimately highlights the project's **strengths, potential improvements, and effectiveness** in simulating a real-world Booking platform.

## CHAPTER – 3

# Methodology

The methodology of this project focuses on designing and implementing a **Flight Booking Simulation** using **Python and Flask**, supported by **object-oriented programming principles** and **JSON-based data handling**
Key steps include:

- **System Design:** Defining core modules such as Add Flight, Remove Flight, Book Seat,Cancel Booking, Passanger List
- **Backend Development:** Implementing Python functions and OOP classes to handle cart management, billing logic, and data flow.
- **Data Handling:** Using JSON files for storing menu items and order history, ensuring flexibility and easy modification without changing code.
- **Web Integration:** Applying Flask to connect backend logic with frontend templates, enabling smooth navigation between pages.
- **Testing & Validation:** Running both terminal-based and web-based versions to ensure correctness, smooth flow, and error handling.

This methodology ensures that the system is **structured, modular, and efficient**, closely replicating real-world Booking platforms.

# CHAPTER – 4

## Code

```
import json
import os
from datetime import datetime

# File names
FLIGHTS_FILE = "flights.json"
PASSENGERS_FILE = "passengers.json"

# ---------------- Utility Functions ----------------

def load_data(file):
    if not os.path.exists(file):
        return {}
    try:
        with open(file, "r") as f:
            data = json.load(f)
            if isinstance(data, dict):
                return data
            return {}
    except json.JSONDecodeError:
        return {}

def save_data(file, data):
    with open(file, "w") as f:
        json.dump(data, f, indent=4)

# ---------------- Flight Management ----------------

def add_flight():
    flights = load_data(FLIGHTS_FILE)

    flight_no = input("Enter Flight Number: ").strip().upper()
    if flight_no in flights:
        print("Flight already exists!")
        return

    origin = input("Enter Origin: ").strip()
    destination = input("Enter Destination: ").strip()
    capacity = int(input("Enter Seat Capacity: "))

    flights[flight_no] = {
        "flight_no": flight_no,
        "origin": origin,
        "destination": destination,
        "capacity": capacity,
        "booked_seats": []
    }

    save_data(FLIGHTS_FILE, flights)
    print(f"☑ Flight {flight_no} added successfully!")
```

```
def remove_flight():
    flights = load_data(FLIGHTS_FILE)
    flight_no = input("Enter Flight Number to remove: ").strip().upper()
    if flight_no not in flights:
        print("Flight not found!")
        return

    del flights[flight_no]
    save_data(FLIGHTS_FILE, flights)
    print(f"❌ Flight {flight_no} removed successfully!")

def list_flights():
    flights = load_data(FLIGHTS_FILE)
    if not flights:
        print("No flights available!")
        return
    print("\nAvailable Flights:")
    print("=" * 40)
    for f in flights.values():
        print(f"{f['flight_no']} | {f['origin']} -> {f['destination']} | Seats: {f['capacity']} | Booked:
{len(f['booked_seats'])}")
    print("=" * 40)

# ---------------- Booking Management ----------------

def book_seat():
    flights = load_data(FLIGHTS_FILE)
    passengers = load_data(PASSENGERS_FILE)

    flight_no = input("Enter Flight Number: ").strip().upper()
    if flight_no not in flights:
        print("Flight not found!")
        return

    flight = flights[flight_no]
    if len(flight["booked_seats"]) >= flight["capacity"]:
        print("No seats available on this flight!")
        return

    # Step 1: Passenger details
    name = input("Enter Passenger Name: ").strip()
    contact = input("Enter Contact Info: ").strip()

    # Step 2: Show seat summary
    total_seats = flight["capacity"]
    booked = sorted(flight["booked_seats"])
    available = [s for s in range(1, total_seats + 1) if s not in booked]

    print("\nSeat Summary for Flight", flight_no)
    print("=" * 40)
    print(f"Total Seats   : {total_seats}")
    print(f"Booked Seats  : {booked if booked else 'None'}")
    print(f"Available     : {available if available else 'None'}")
    print("=" * 40
    # Step 3: Choose seat
```

```
    seat_no = input("Enter Seat Number (leave blank for auto): ").strip()
    if seat_no:
        seat_no = int(seat_no)
        if seat_no < 1 or seat_no > flight["capacity"]:
            print(f"Invalid seat number! Please enter between 1 and {flight['capacity']}")
            return
        if seat_no in flight["booked_seats"]:
            print("Seat already booked! Choose from available seats.")
            return
    else:
        if not available:
            print("No seats available on this flight!")
            return
        seat_no = min(available)

    # Step 4: Save booking
    booking_id = uuid.uuid4().hex[:8].upper()
    passenger_id = uuid.uuid4().hex[:8]
    passengers[booking_id] = {
        "passenger_id": passenger_id,
        "name": name,
        "contact": contact,
        "flight_no": flight_no,
        "seat_no": seat_no,
        "booking_id": booking_id,
        "status": "CONFIRMED",
        "booked_at": datetime.now().strftime("%Y-%m-%d %H:%M")
    }

    flight["booked_seats"].append(seat_no)
    save_data(FLIGHTS_FILE, flights)
    save_data(PASSENGERS_FILE, passengers)
    print(f"☑ Seat {seat_no} booked successfully! Booking ID: {booking_id}")

def cancel_booking():
    passengers = load_data(PASSENGERS_FILE)
    flights = load_data(FLIGHTS_FILE)

    booking_id = input("Enter Booking ID: ").strip().upper()
    if booking_id not in passengers:
        print("Booking not found!")
        return

    booking = passengers[booking_id]
    flight_no = booking["flight_no"]
    seat_no = booking["seat_no"]

    # Mark passenger as canceled
    booking["status"] = "CANCELED"
    passengers[booking_id] = booking

    # Free the seat
    if flight_no in flights and seat_no in flights[flight_no]["booked_seats"]:
        flights[flight_no]["booked_seats"].remove(seat_no)
```

```
    save_data(FLIGHTS_FILE, flights)
    save_data(PASSENGERS_FILE, passengers)
    print(f"❌ Booking {booking_id} canceled successfully!")

def passenger_list():
    passengers = load_data(PASSENGERS_FILE)
    if not passengers:
        print("No passengers found!")
        return

    print("\nPassenger List")
    print("=" * 60)
    for p in passengers.values():
        print(f"{p['name']} | Flight: {p['flight_no']} | Seat: {p['seat_no']} | Status: {p['status']} | BookingID:
{p['booking_id']}")
    print("=" * 60)

# ---------- Main Menu ----------

def main():
    while True:
        print("\nFlight Booking System")
        print("=" * 40)
        print("1. Add Flight")
        print("2. Remove Flight")
        print("3. List Flights")
        print("4. Book Seat")
        print("5. Cancel Booking")
        print("6. Passenger List")
        print("7. Exit")
        choice = input("Enter choice: ")

        if choice == "1":
            add_flight()
        elif choice == "2":
            remove_flight()
        elif choice == "3":
            list_flights()
        elif choice == "4":
            list_flights()
            book_seat()
        elif choice == "5":
            cancel_booking()
        elif choice == "6":
            passenger_list()
        elif choice == "7":
            print("Goodbye!")
            break
        else:
            print("Invalid choice! Try again.")

if __name__ == "__main__":
    main()
```

# CHAPTER - 5

## OUTPUT


## Available Flights

**AIR101**
KPL → BLY

Departure: 16/8/2025, 4:11:00 pm
Arrival: 17/8/2025, 4:11:00 pm

**Available: 100/100**
Booked: 0

View Details | Delete

**FL 12**
kpl → blr

Departure: 18/8/2025, 7:40:00 pm
Arrival: 19/8/2025, 7:40:00 pm

**Available: 109/110**
Booked: 1

View Details | Delete

## CHAPTER - 6

# Conclusion (Final Slide Content)

- The Python-based Flight Booking Simulation provides a simple yet powerful solution that effectively balances **functionality, usability, and technical quality**. The project successfully implements all the essential features of a real Booking platform, including Add Flight, Remove Flight, Book Seat,Cancel Booking, Passanger List

**Key strengths** of the system include:

- **Core logic clarity**: Built using Python with an emphasis on object-oriented principles, making the code reusable and easy to maintain.
- **Data persistence**: Orders and restaurants are managed using JSON files, ensuring reliability even without external databases.
- **Smooth workflow**: From selecting restaurants to generating bills, the user experience is seamless.
- **Flexibility**: The system works both in a **terminal environment** (logic-focused) and a **Flask-based web version** (frontend-enabled), showing adaptability across platforms.

**Areas for future enhancement** may include:

- Integration with real databases (e.g., MySQL, MongoDB).
- Adding user authentication and role-based access.
- Implementing payment gateway simulation.
- Enhanced UI/UX design for the web version.

Overall, the project demonstrates effective **Python programming practices**, a strong focus on **core logic**, and lays a solid foundation for further development into a more advanced, industry-ready Flight Booking application.

**CHAPTER – 7**

# References (Final Slide Content)

**Flight Booking Simulation System – Python-based application implemented using Flask (for web version) and Object-Oriented Python (for terminal version). The project uses JSON files for storing** Add Flight, Remove Flight, Book Seat,Cancel Booking, Passanger List


- **Python Official Documentation**
- **Flask Documentation**
- **W3Schools – Python**
- **GeeksforGeeks – Python Tutorials**