Test Plan Documentation for ContosoAir

1. Introduction

The ContosoAir aims to deliver a full-featured web platform that allows users to search for flights, view details, make bookings, and manage their itineraries. This test plan describes the testing approach, strategy, and scope to ensure the system meets defined quality standards and functions as expected.

2. Test Plan Objective

The primary objectives of this test plan are to validate that the application:

- Complies with functional, performance, usability, and security requirements.
- Is free of major defects that could hinder user experience.
- Performs reliably across supported web browsers and devices.
- Integrates properly with external systems (e.g., flight data APIs).

3. Scope of Testing

In-Scope

1. **Flight Search**
    a. Validate accurate search results based on user input.
    b. Test filtering and sorting by price, stops, and airlines.
    c. Confirm sorting mechanisms like time and price work as expected.
2. **Flight Booking**
    a. Test end-to-end user flow from flight selection to booking.
    b. Validate authentication and access control.
    c. Check payment gateway integration, including validation, successful transactions, and error handling.
3. **Flight Display**
    a. Ensure correct display of timing, layovers, baggage information, and pricing.
4. **Itinerary Overview**
    a. Verify users can view and update their itineraries.
    b. Confirm updates reflect real-time changes (e.g., schedule updates).
5. **Real-Time Flight Availability**
    a. Check current flight availability based on user queries.

b. Test integration with external data providers for real-time responses.

4. Test Objectives

The testing process will aim to verify:

- **Functional Testing**: Validate that all features work as intended.
- **Usability Testing**: Ensure the interface is user-friendly.
- **Performance Testing**: Measure system behavior under various loads.
- **Security Testing**: Safeguard sensitive user and payment data.
- **Compatibility Testing**: Ensure cross-browser and cross-platform support.

5. Test Approach

1. Test Types

- **Manual Testing**: For exploratory and functional testing.
- **Automated Testing**: Used for regression and repetitive test cases.
- **Performance Testing**: To assess system behavior under load.

2. Testing Levels

- **Unit Testing**: Conducted by developers for isolated components.
- **Integration Testing**: Ensure components like booking, payment, and flight search work together.
- **System Testing**: Full-system validation under various scenarios.
- **User Acceptance Testing (UAT)**: Performed by stakeholders to confirm readiness.

3. Testing Tools

- **Test Management**: Azure DevOps Test Plans
- **Automation**: Selenium, Postman (API), JMeter (load testing)
- **Bug Tracking**: Azure DevOps

6. Test Deliverables

1. **Test Cases**: Covering flight search, booking, itinerary, profile, and package features.
2. **Execution Reports**: Detailing test results (pass/fail).
3. **Defect Logs**: Including issue severity, steps to reproduce, and status.
4. **Summary Report**: Overview of test progress, findings, and final status.

7. Test Schedule

Phase	Start Date	End Date
Test Planning	15-05-2025	16-05-2025
Interface Creation	17-05-2025	18-05-2025
Test Case Creation	19-05-2025	20-05-2025
Test Implementation	21-05-2025	22-05-2025
Test Case Verification	23-05-2025	24-05-2025
Test Completion	24-05-2025	24-05-2025

8. Resource Planning

- **QA Testers:** 4 team members for manual and functional testing
- **Automation Engineers:** 4 members focused on regression and performance automation
- **Developers:** Available for bug resolution
- **Project Manager**: Oversees progress and delivery timelines

G. Test Environment

1. **Hardware**

- Backend Server: Windows Server
- Client Systems: Windows and macOS for compatibility testing

2. **Software**

- **Management:** Azure DevOps
- **Browsers:** Chrome, Firefox, Safari, Edge
- **Backend Services:** APIs for search and payment
- **Automation Tools**: Selenium, Postman, JMeter, LoadRunner

3. Database

- **Azure Cosmo'sDB** with test data for flights and users

10. Risk and Mitigation

Risk	Likelihood	Impact	Mitigation Strategy
Flight data API delay	High	High	Coordinate with API providers; use mock data.
Browser inconsistencies	Medium	Medium	Conduct early-stage compatibility testing.
Payment integration errors	Low	High	Perform testing in sandbox environments.
Incomplete test coverage	Low	High	Develop comprehensive test cases covering all major flows.

11. Time Allocation **Total Time:**

Task	Estimated Time (hrs)	Assigned To
UAT Test Cases	6	*Raihan Mohamed*
Project Setup C Environment	4	*Sobiya Kumar*
Automation Test Cases	10	*Shivani Gudiboyana* *Vaishnavi Chennapragada* *Sobiya Kumar* *Raihan Mohamed*
API Testing	5	*Vaishnavi Chennapragada* *Shivani Gudiboyana*
Performance Testing	5	*Raihan Mohamed*
Project Review C Presentation	5	*Vaishnavi Chennapragada*

12. Approval

Role	Name

Project Manager	*Suresh Nanjan*
QA Tester 1	*Raihan Mohamed*
QA Tester 2	*Vaishnavi Chennapragada*
QA Tester 3	*Shivani Gudiboyana*
QA Tester 4	*Sobiya Kumar*