Test Plan for Airline Application

1. Introduction

This test plan outlines all the functionalities of an airline application including the booking, flight management, user management, payment gateway etc. This plan will ensure that the application meets all the required functional, performance and useability standards allowing the users a seamless experience meeting all the business requirements.

2. Objectives

- Verify the application meets all functional and non-functional requirements.
- Verify the correctness, performance, usability, and security of the application.
- Verify the application meets all the performance standards.
- Ensure integration with third-party services (e.g., payment gateways, APIs).
- Identify and fix critical bugs before release.

3. Scope

- Flight search and filter functionality
- Flight booking and seat selection
- User registration and login
- Payment gateway integration
- E-ticket generation and email notification
- Online check-in
- Admin Panel for flight schedule management

4. Test Strategy

- A combination of both Manual and Automation testing will be employed
- **Manual Testing:** Will be used for exploratory testing, usability testing, and functional testing of complex scenarios.
- **Automation Testing:** Will be used for regression testing, performance testing, and repetitive functional tests. Tools like Selenium will be considered.
- **Types of Testing:**
  Functional Testing: Verify that all application functional functionality works as expected.
  Usability Testing: Evaluate the application ease of use and user experience.
  Performance Testing: Assess the application's response, load times, and stability under normal and peak conditions.

Security Testing: Identify potential security vulnerabilities, especially in payment processing and data handling.

Regression Testing: Ensure that new changes do not negatively impact existing functionality.

Integration Testing: Verify the correct interaction between different modules, especially with the payment gateway and the application backend systems.

Accessibility Testing: Verify the app adheres to accessibility guidelines.

5. **Test Environment**
    - **Browsers:** Chrome, Safari, Firefox, Edge (Latest Versions)
    - **OS:** Windows, iOS, MacOS, Android
    - **Mobile Device:** Android and iOS
    - Test Data for various user scenarios
6. **Test Scenario**
    - User and Admin Login
    - Admin flight schedule management
    - User searches the flight
    - User books the seat
    - Payment with online banking
    - Payment with UPIs
    - Payment with Debit/Credit Card
7. **Sample Test case**

    Admin Login

    1. Open Admin portal login page
    2. Enter the correct user login credentials
    3. Click login

    Admin flight schedule management

    1. Open Admin portal login page
    2. Enter the user login credentials
    3. Click login
    4. Click on schedule manager
    5. Enter the details (Flight number, Departure, Arrival)
    6. Click on Update

8. **Roles and responsibilities**

Role	Responsibility
QA Lead	Test planning and reporting
QA Engineers	Test case writing and execution
Developers	Bug fixing and unit testing
Product Owner	Requirement clarification

9. Test Deliverables

- Test cases and Test scripts
- Test Plan Document
- Bug report
- Test summary report
- Automation test results

10. Test Schedule

Phase	Timeline
Requirement Analysis	2 Days
Test Case Design	2 Week
Test Execution	1 Week
Bug Fix Verification	1 Week

11. Risk and Mitigation

Risk	Mitigation
Issue with Environment Setup	A proper testing environment must be setup and backup plan should be ready
Unavailability of Data	Desired data should be ready
Tight Deadline	Testcases prioritising the main functionality should be executed
Resource unavailability	A strategy should be ready to mitigate the risk of resource unavailability

12. Exit Criteria

- 95%+ testcase execution
- All the major functionalities should be covered
- All critical defects must be resolved