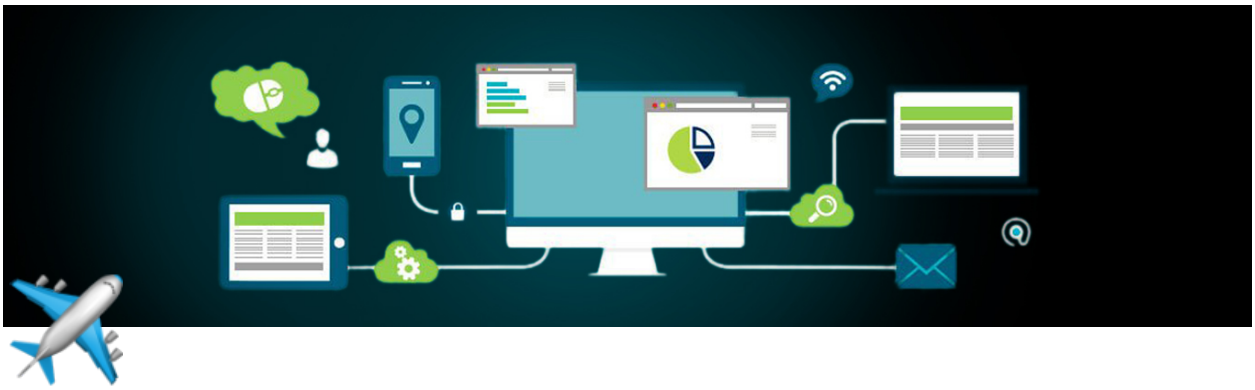

SE Assignment #1

Airline Booking System

Requirement Specifications Document

1. Introduction

Our project “Airline Booking Management” aims to provide a funcitonal platform for managing airline bookings efficiently. It will facilitate users to search & book flights, manage booking reservations, and handle flight information all with a user-friendly platform. The system will ensure re-usability, maintainability, reliability, and security as outlined in the project requirements.

2. Functional Requirements

▼ **User Registration and Authentication:**

- Users should be able to register/login and create accounts.
- Different levels of access should be implemented for admin, staff, and user.

- User info (username, password) stored in user database.

▼ **Flight Search and Booking:**

- Passengers should be able to search for available flights based on criteria such as date, origin, destination, and departure time.
- Passengers should be able to book flights by providing necessary details such as passenger names, contact information, and payment details.
- The system should handle multiple passenger counts, seat allocation and availability.
- The system should confirm booking requests and issue printable tickets upon successful payment.
- User should be able to view today's flights schedule with appropriate flight status.
- Usage of appropriate booking and flight databases to retrieve info.

▼ **Payment Processing:**

- The system should securely process payments for flight bookings using various payment methods.

▼ **Booking Reservation Management:**

- Passengers should have the ability to view and manage their reservations based on booking reference.
- Options to modify passenger info details or cancel bookings.

▼ **Flight Management:**

- Admin should be able to manage flight schedules, including viewing, adding, modifying, and deleting flights from the database.
- Filtering and search options should be provided for flight viewing.

▼ **System Logs:**

- Admin should have the ability to view system logs, including user activities, booking history, and payment transaction logs.

3. Nonunctional Requirements

▼ **Re-usability:**

- The system architecture should follow an object-oriented approach, promoting re-usability of components and modules.
- Design patterns such as “Singleton” classes, where a single instance of a class exists and there is global access to it, and it reduces the need for multiple instances of an object.
- Code should be organized into reusable components, functions, libraries and modules.

▼ **Maintainability:**

- Code should be well-structured and documented (proper indents, spaces, appropriate variable and file names, no code redundancy) to facilitate easy maintenance.
- The system should be modular, allowing for updates and enhancements without risk of disruption of other components.
- Documentation & User-manual should be provided for user self-maintenance.

▼ **Reliability:**

- The system should guarantee an operational availability of atleast 99%, ensuring uninterrupted service for users.
- Failover mechanisms and backup servers should be in place incase of system failures.

▼ **Security:**

- User access should be securely managed through authentication mechanisms such as multi-factor authentication (MFA).
- All sensitive data, including user information and payment details, should be encrypted with https protocol.

▼ **Portability:**

- The system should be designed to be portable across different platforms.
- It should support deployment on various operating systems such as Windows, Linux, and macOS.
- The system should be compatible with different web browsers, ensuring accessibility for users regardless of their preferred browser.

▼ **Scalability:**

- The system should be designed to scale both vertically and horizontally to accommodate varying levels of usage and data volume.
- Scalability should be achieved without compromising system performance, responsiveness, and user experience.
- The system should incorporate automated scaling mechanisms to dynamically adjust resource allocation based on workload and traffic patterns.

▼ **User Experience:**

- User-friendly interface specifically for all users, presented with a home page and dashboard from which they can easily navigate to different features.