# **Software Requirements Specification (SRS) for an Online Flight Reservation System**

Student Name: Precious Mwale

Student ID: SE2023008

College: Lusaka Institute of Technology (LIT)

Date: June 20, 2025

## **1. Introduction: What We're Building Here**

So, this whole document is basically my blueprint for an online system where people can book flights. I'm calling it the "Zambian Air Connect" (ZAC) system. The main idea is to make a website that's super easy for anyone to use – they can search for flights, pick their seats, and pay for tickets right there. On top of that, it also needs tools for the airline staff to keep track of flights and bookings on their end. It's all about making flight reservations smooth and digital, moving away from all the paper and phone calls.

## **2. General Description of the System**

### **2.1 The System at a Glance**

The ZAC is going to be a web application. Think of it as a central online place for buying plane tickets. It should cut down on manual errors and make sure flight info is always correct and current. For now, it's a standalone thing, but it's built to eventually link up with any existing airline systems if they have them, to get real-time flight updates.

### **2.2 Who Are the Users?**

There are two main groups who will interact with the ZAC:

* **Travellers (Passengers):** These are the everyday people looking to fly. They'll use the public part of the website to search, compare prices, make bookings, and view their trips. They're assumed to be comfortable with using websites generally.
* **Airline Operatives (Admin Staff):** These are the people who work for the airline. They'll need a secure, restricted part of the system to manage flight schedules, handle customer queries, and make changes to bookings. Their access will depend on what their job is.

### **2.3 Where It Will Run**

The ZAC will run in standard web browsers like Chrome, Firefox, Edge, and Safari. It's really important that it looks good and works perfectly on any device, whether it’s a big computer screen, a tablet, or a small mobile phone. It'll be hosted on a reliable cloud platform, so it’s always accessible.

### **2.4 Constraints – What We Must Do**

* **Data Protection:** This is huge. Since we're dealing with people's personal info and their money, everything has to be locked down tight. All data, whether it's being sent or just sitting there, needs to be encrypted. We need strong safeguards against any kind of data breach.
* **Rules and Regulations:** We absolutely have to follow all the aviation laws in Zambia and international privacy rules about customer data. No shortcuts here.
* **External Links:** The system has to connect securely with different online payment providers.

### **2.5 Assumptions – What We're Counting On**

I'm assuming that most users will have a stable internet connection. Also, that the payment gateways we choose will have good, reliable APIs for us to integrate with. And, of course, that we’ll have a solid, scalable database system ready to hold all the flight and user data.

## **3. Specific Functions of the System**

### **3.1 User Accounts & Logins**

* **F1.1:** New passengers must be able to sign up for an account. This means providing a unique email, setting a strong password, and giving their basic details (like name and phone).
* **F1.2:** Registered users need to be able to log in securely using their email and password.
* **F1.3:** Once logged in, users should easily find and update their profile details.
* **F1.4:** If someone forgets their password, there should be a secure way for them to reset it via their registered email.

### **3.2 Finding Flights**

* **F2.1:** Users must be able to search for flights. They'll put in where they're flying from, where they're going, the dates they want to travel (for one-way or return), and how many people are going (adults, children, infants).
* **F2.2:** The search results should clearly show all the important flight information: airline, flight number, exact departure and arrival times, how long the flight takes, how many stops there are, and the price.
* **F2.3:** The system needs filtering options so users can sort by things like price, specific airlines, number of stops, or what time of day the flights are.

### **3.3 The Booking Process**

* **F3.1:** Users need to be able to pick the flight(s) they want from the search results to start booking.
* **F3.2:** The system will guide the user through entering all the necessary details for each person flying (full legal name, date of birth, and any passport info needed for international trips).
* **F3.3:** Before they pay, the user must see a clear summary of everything – the flights they picked, all the passenger details, and the total cost – for them to review and confirm.
* **F3.4:** After a successful booking, the system will give them a unique booking reference number.

### **3.4 Paying for Flights**

* **F4.1:** The system will securely connect with at least one big online payment gateway to handle credit/debit card payments (e.g., Visa, MasterCard). We should also look into local mobile money payment options for Zambia.
* **F4.2:** As soon as the payment goes through, the booking status needs to immediately change to "Confirmed."
* **F4.3:** An electronic ticket (e-ticket) and a detailed booking confirmation email will be automatically sent to the user’s registered email after successful payment.

### **3.5 Admin Tasks for Airline Staff**

* **F5.1:** Authorized administrative staff must be able to manage flight schedules. This means adding new flights, changing details for existing ones (like times or capacity), and cancelling flights.
* **F5.2:** Admins also need to be able to search for, view, change, or cancel any passenger booking.
* **F5.3:** The system should provide various reports for admins, like how full flights are, total revenue summaries, and passenger lists for specific flights.

## **4. How It Connects with the Outside World (Interfaces)**

### **4.1 What the User Sees (User Interface)**

The ZAC will have a web-based interface that's easy to use and looks good. It needs to adjust perfectly to different screen sizes (desktop, tablet, mobile) and be straightforward to navigate.

### **4.2 How It Talks to Other Software (Software Interfaces)**

* **Payment Gateway APIs:** Essential for secure online payment processing.
* **Email Service API:** For sending automatic emails like booking confirmations and password resets.
* **Future Airline Core System API (potential):** A possibility to integrate with the airline’s internal systems for real-time flight data.

### **4.3 Communication Security**

All data exchanged between the user's browser and our server, especially sensitive information, must be secured using industry-standard HTTPS encryption.

## **5. The "Behind the Scenes" Requirements (Non-Functional)**

### **5.1 Performance – How Quick It Is**

* **NFR.P1 (Search Time):** Flight search results should show up within 2.5 seconds for 90% of requests, even during busy times.
* **NFR.P2 (User Load):** The system should easily handle up to 80 concurrent users performing searches and bookings without slowing down significantly.

### **5.2 Security – Keeping It Safe**

* **NFR.S1 (Data Encryption):** All sensitive data (PII, payment info) must be encrypted both when it’s being sent and when it’s stored in the database.
* **NFR.S2 (Access Control):** Access to admin functions will be strictly controlled based on what each staff member is allowed to do.
* **NFR.S3 (Authentication):** User logins will use strong password rules, account lockouts, and protection against people trying to guess passwords.
* **NFR.S4 (Session Security):** User sessions will be managed securely to prevent unauthorized access.

### **5.3 Reliability – Always Working**

* **NFR.R1 (Uptime):** The system should be available 99.7% of the time, not counting planned maintenance. It needs to be very dependable.
* **NFR.R2 (Data Safety):** We need strong systems in place (like transactions and regular backups) to make sure booking and payment data is always consistent and never lost.

### **5.4 Usability – How Easy It Is**

* **NFR.U1 (Booking Flow):** A new user, even with just basic computer skills, should be able to complete a flight booking (from search to confirmation) in 5 steps or less.
* **NFR.U2 (Error Messages):** When a user makes a mistake or something goes wrong, the system should give clear, simple messages that help them fix the problem.

### **5.5 Scalability – Ready for Growth**

The ZAC system needs to be built so it can easily handle more users, more flights, and more data in the future without needing a total rebuild. It should be able to grow as the airline grows.

### **5.6 Maintainability – Easy to Manage**

* **NFR.M1 (Code Quality):** The code should be neat, well-commented, and organized in modules. This makes it easy for new developers to understand it, fix bugs, and add new features later on.
* **NFR.M2 (Deployment):** Updates and new features should be able to be put live quickly and smoothly, causing minimal downtime for users.