Functional Requirements Specification

# 1. Introduction

**1.1 Purpose**

The purpose of this document is to define the functional and non-functional requirements for the development of a command-line-based Seat Booking System for Apache Airlines. The system is intended to support operations on their newly acquired Burak757 aircraft by enabling passengers to book, cancel, and view seat statuses through an interactive menu.

# 2. Functional Requirements

**2.1 Requirement 1: Seat Availability Check**

Description: The system shall allow the user to input a seat(e.g., “12C”) and display weather it is free (F), Reserved(R), or unavailable duo to being an aisle(X)or storage(S).

**2.2 Requirement 2: Book a Seat**

Description:

 The system shall allow a user to reserve a seat by entering a valid, available seat code. If the seat is marked as “F”, it will be updated to “R”. The system must reject bookings on already reserved seat “R”, aisle(X), or storage (S) seats.

**2.3 Requirement 3: Free a Seat**

Description:

The system shall allow users to free a reserved seat. If a seat marked as “R” is selected, it will be returned to the “F” (free) state. Seats not currently booked will not be affected.

**2.4 Requirement 4: Show Booking Status**

Description:

The system shall visually display a seat layout grid that represents each seat and its current status (R, F, X, or S). This will assist users in viewing which seats are booked or available.

**2.5 Requirement 5: Prevent Invalid Bookings**

Description:

The system shall ensure that no seat marked as “X” (aisle), “S” (storage), or already reserved (R) can be booked. The system shall show appropriate feedback messages when such bookings are attempted.

# 3. Non-functional Requirements

**3.1 Performance**

-The system shall process user inputs and update the seat map in real time, with a response time of less than 1 second for all operations.

-The system must handle all 80 rows (480 seats) efficiently without noticeable delays.

**3.2 Usability**

-The system must have a simple, intuitive command-line interface (CLI) with clearly labelled menu options. Seat codes and instructions must be easy to understand.

-Error messages must be user-friendly and help users correct their input (e.g., invalid seat codes or unavailable options).

-The seat map shall be displayed in a readable table-like format that reflects the real seating layout of the aircraft.

**3.3 Security**

- Input validation shall be enforced to prevent invalid seat entries or unauthorized seat status changes.

-Booking and freeing of seats shall only be possible through the defined menu options.

-Although no authentication is required in Part A, the system must be structured in a way that can support secure features in future versions (e.g., storing customer details in Part B).

**3.4 Security**

-

# 4. Assumptions and Constraints

4.1 Assumptions

-The seat layout will remain fixed based on the provided Burak757 diagram.

-The initial version will run locally on a command-line terminal.

-Users are expected to understand the basic interface without the need for a help system.

4.2 Constraints

-The application must be console-based (Hard to show the seats in table).

-The application must follow the seat rules (e.g., no booking on X or S).

-Future versions may require database support for customer data (to be implemented in Part B).

-Input errors (e.g., selecting invalid seats or entering wrong menu choices) must be handled gracefully.