EECS 3550 – Software Engineering

Spring 2021 – Team Project – "Air3550" – Problem Statement

Now that the 737Max fiasco has been resolved, and the COVID-19 vaccine has become available, we are going to fly (pun intended) in the face of conventional wisdom, and get into the airline market.

Surely, after working from home for the last year and meeting via Zoom/WebEx/Teams, all of those business travelers can't wait to get back to standing in lines at the airport, boarding long aluminum tubes full of sneezing, coughing strangers, and staying in hotels last inhabited by…who knows who, just so they can meet with a client, and do all of the above in reverse to get home!

So, we're going to capitalize on this inevitable surge in airline travel by building a reservation system for a new airline we're going to launch.

There is too much red tape to get approved for international travel, and flying over the ocean requires extra certification (you have to be able to prove that your flights can safely get all the way there on one engine, and we don't really trust our maintenance folks THAT much). So, we're only looking at servicing the 48 contiguous US States.

We won't do seat assignments (a la Southwest), so you don't need to worry about how the seats are configured in various aircraft (three on each side of the aisle, etc). Otherwise, here's how the system will work:

1) There will be a list of (real-world) cities with (real) airports served by our airline. You will need to decide where your planes will fly, but you must include a reasonable set of cities.

You may want to consult various airlines to see where they fly. I might suggest you start with the "big locations" (search Wikipedia for a list of the busiest airports in the US).

There are real-world issues to deal with. I don't believe any domestic flights of less than 1000 miles go to/from JFK (it's primarily international and/or long-distance), so you can "collapse" the New York locations into a single airport and dump all of the traffic to LaGuardia. Same with Chicago (Midway/O'Hare)

You don't have to service every US airport, but you must service at least 10, and because I plan to use your service, you must include Nashville and Cleveland in your list.

2) You need to know the (straight-line) distances between airports. Customers will earn points (and flights will be priced) based on how far they're flying.

3) You won't have direct flights *from* everywhere *to* everywhere else. This isn't a fully-connected graph. For example, a flight from Nashville to Cleveland may be direct, but a flight from Nashville to Seattle may have to connect in Chicago. No flight may have more than three legs (2 connections), and no connection (layover) may be less than 40 minutes (if the arriving plane is delayed, there won't be time to make the connection, and then we have to scramble to get the traveler on a different flight).

4) You need to have a variety of planes – choose at least three from the 737, 747, 757, 767, and 777 models (after all, we need to help Boeing get back on its feet). Each will have a different capacity, so there are only so many people you can book on a given flight. You'll need to know what kind of plane a given flight will use. It makes sense that the longer flights (LA – New York?) use something like a 747, while "regional" flights (perhaps Cleveland – Nashville) uses the smaller, less-costly-to-operate 737.

We do NOT oversell. If a plane holds 135 passengers, some airlines will sell up to 140 tickets, banking on 5 of the passengers to not show up – this maximizes their revenue. If they only sold 135, and 5 don't show up, they fly with empty seats. If MORE than 135 happen to show up, they apologize to the one-to-five people affected, bump them from the flight they paid for, get them on the next available flight, and given them a $100 (or more) travel voucher for their trouble. We don't want that hassle, so we will never over-book a flight.

5) Passengers can book a one-way flight, or a round-trip. They have to know when they book the round-trip when they plan to return (date/time). It's up to you to pick some schedules. You may want to look at the Southwest website, pick a few airports, and duplicate their schedules. If you would like to compute your own, consider TravelMath.com. Assume 30 minutes per flight (15 to get from the gate to the end of the runway, and then 15 more to land and taxi to the gate at the destination), plus 500 miles per hour in-flight.

6) Tickets may be booked for up to 6 months in advance (no further out than that).

7) Pricing for tickets is $50 (for fixed costs), plus twelve cents per mile between airports (longer flights take more fuel, more hours of flight attendants' time, put more wear on the plane, which means more maintenance, etc.), and a mandatory $8 per segment federal fee to pay for the TSA agents (if a flight lands between its origin and destination, every time it takes off, that's $8). Flights leaving before 8 AM, or arriving after 7 PM receive a 10% off-peak discount. Flights leaving or arriving between midnight and 5 AM receive a 20% red-eye discount.

8) A marketing manager will decide which planes to use for each flight

9) A load engineer will decide which routes to offer – you may manually populate your user store with one or more marketing managers and load engineers. Obviously, customers can create only customer user types. There will need to be some way for a load engineer to manage the schedule flights between cities (i.e., add/delete/edit the list of available origin/destination/times). You may assume that the same flights operate 7 days a week.

10) Customers will book their own flights (we don't have reservation agents).

11) Flights may be cancelled up to an hour before departure time. The customer receives a credit (which the airline keeps) that they can use for a later flight with us.

12) Each user will have a unique 6-digit (can't start with zero), randomly-assigned customer number (which will act as their user ID), tied to their name and address. Customers will earn points (10 points per dollar spent) on purchases. We keep the balance of each customer's account online. Customers will log in with their ID numbers and a password, which they will pick, and which they will be able to change. We will store their password as a SHA-512 hash of the real password.

13) If a customer cancels their flight, we credit them with the cost of the flight, but they only earn points when they take the flight (or pay for it and don't show up).

14) Customers may purchase tickets using their accumulated points. A point corresponds to one cent, so a $100 ticket requires 10,000 points. If a customer books a flight with points and cancels the flight, we credit the points back to their account (not the dollar equivalent), so you will need to know whether a flight was booked with points or dollars. Flights may not have mixed-payment methods (you can't use $50 and 5,000 points to book at $100 flight) – they're all-or-nothing (money / points).

15) A customer can create their account, which will include contact information (name, address, phone, age, etc.), billing information (credit card number) that they provide. They will be able to see their account history (flights booked, taken, canceled, points used / available).

16) Customers can print their boarding pass 24 hours prior to flight time. The boarding pass will include their flight number, first and last names, and where they are traveling from and to, as well as the departure and arrival times, and their account number. You don't have to actually produce a piece of paper (this is a GUI; just demonstrate the screen; we'll assume implementing "print this" can be done).

17) For financial transactions, you just have to create a record of the customer's first and last name, the credit card number, and the amount to charge (no communication has to be generated; you won't actually be simulating the payment – just display and record it).

18) Each flight will have a number, an origin city (and code), a destination city (and code), and an associated price and number of points that purchasing that flight will earn.

19) When the customer wants to book a flight, they will log in with their account number, and then provide the travel date(s) (one for a one-way, and both for a round-trip), and origin an destination cities, and your program will propose flights, which they can then select and book. There is no option for "reserve now; pay later" – if they reserve it, they pay for it now.

20) A flight manager will need to be able to print flight manifests – lists of everyone on each flight when it takes off. You won't have to actually generate a piece of paper for the manifest, but you will have to create some representation of the manifest (a grid on the GUI, CSV text file, etc.).

21) The accountants will want to know how many flights we had, how full each one was (percentage of capacity), and what the income was not only per flight, but for the company as a whole. Such summary reports will need to be available to an accounting manager (yes, you can manually create the accounting manager ID), and you only have to generate the GUI of the report, or a CSV file.