Project Proposal - Phase II

Group 3: Abhi Bhagat, Nate Campbell, Rami Dari, Megan LaRoy, Christina Lin, Tom Van den bulck, Ciara Wheeler
Date due: March 6, 2024

Problem Statement

To allow for users trying to book cheap and efficient flights for their travel, Flight Sky Scanner requires a database which can track different airplanes, airlines, airports, and route schedules from different airports. Flight Sky Scanner does not have a database right now and there is no efficient way for users to find the cheapest available flights for their travel.

To help Flight Sky Scanner, we will design a fully functioning website that interfaces with a database and is hosted on a web server. We will develop a user page, where the user can enter where they would like to travel to and from, and the dates of their travel and the interface will populate with flight routes, their costs, and the availability of each. We will also develop a page that just has upcoming flights listed, not specified to the search of the user. This can be for browsing cheap flights to unspecified destinations. Then there will be a page for booking the flight, which will ask for all of the user's information required and payment.

System Requirements

1. Information provided by the system should be up-to-date
2. The system should update when the information is uploaded or selected in the web-based application
3. The system should load the web page within 5 seconds of the user entering a search (assuming user has proper connection)

Functional Requirements

1. The system should allow the user to navigate and search for flights
2. The system should allow the user to book a ticket on the website for the flights
3. The system should allow the user to see available flights
4. The system should only show flight information that has available bookings
5. The system should allow administrator to view statistics on common searches

Conceptual Database Design
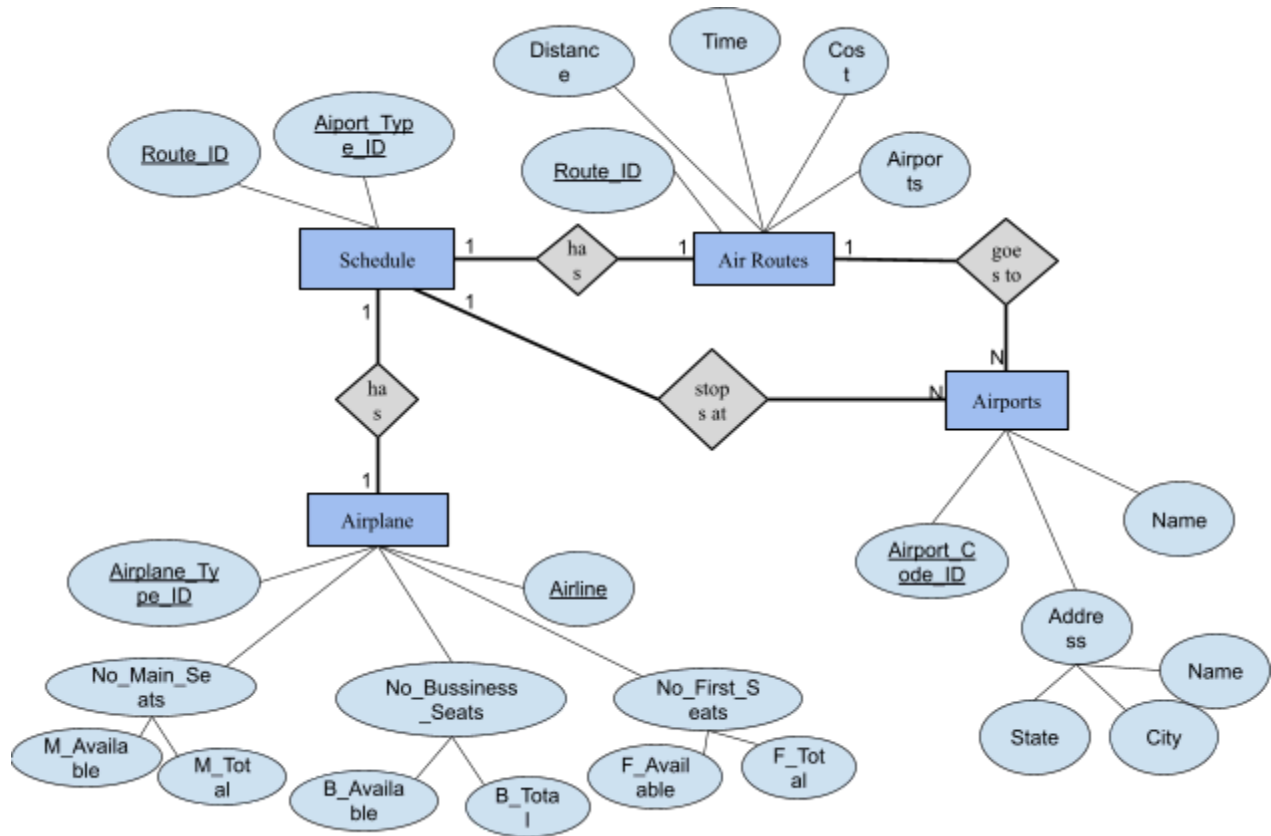
Logical Database Design

- Underline indicates primary key
- Italics indicates foreign key
- * marks a value which must be unique

Airports

Attribute	Type	Description	Functionally Depends On
*Airport_code_Id *****	Int	Unique ID number for each airport	
Name	Varchar(100)	Name of the airport	
Address	Varchar(100)	Address for the airport	
State	Varchar(25)	Name of the state	
City	Varchar(50)	Name of the city	

Airplanes

Attribute	Type	Description	Functionally Depends On
*Airplane_type_Id *****	Int	Unique Id for the type of plane	
Airline	Varchar(50)	Name of the airline the plane belongs to	
no_main_seats	Int	Number seats on the plane	
M_available	Int	Number of seats available	
M_total	Int	Total number of seats in the main class	
no_bussiness_seats	Int	Number of seats on the business cabin	
B_available	Int	Total number of available seats	

B_total	Int	Total number of seats in the business class	no_business_seats
no_first_seats	Int	Number of seats in first class	
F_available	Int	Number of seats available for first class	no_first_class
F_total	Int	Total number of seats in first class	no_first_class

Air routes

Attribute	**Type**	**Description**	**Functionally Depends On**
Route ID*	Int	Unique Id given to the route.	
Distance	Int	The amount of miles flown for the route.	Airports
Time	Date	The time and date for the air route.	
Cost of Flight	Double	The cost for the flight.	
Airports	Varchar(50)	The name of the location the plane is flying to.	

Schedule

Attribute	**Type**	**Description**	**Functionally Depends On**
Route ID*	Int	Unique id given to the route. (Foreign key)	
Airplane Type ID*	Int	Unique id given to the plane type. (Foreign key)	

Airport Code ID*	Int	Unique Id given to the airport. (Foreign key)	

Application Programs Design

Ticket Purchasing

Display Available Airplane Module

Description:

This module is called upon by the Purchase Ticket module. When the user accesses the website to find the flight that they would like to purchase, this module will present the available air routes and their respective schedule according to the users' desired requirements. If the user desired requirements do not fit an available route a message will be prompted to inform them that there are no available routes with the given specifications. If there are available routes with the given specifications the air route will be displayed and the user may purchase the ticket if they decide to do so.

Input:

User specified air route

Output:

Tickets available for specified air route or a message that no air route was found

Requirements:

User must use valid airports and provide a minimum of 2 airports (1 departure and 1 arrival)

Interactions:

This module is only called by the Purchase Ticket Module.

Conflicts/Side-effects:

There can be multiple air routes that will satisfy the user requirements for the ticket that they would like to find. The user may sort the available air routes based on the filter preference.

Purchase Ticket Module

Description:

This module allows a user to purchase a ticket that satisfies their desired travel requirements if there is a route available and if there are seats available.

Input:

User travel requirements/desires

Output:

Airplane ticket or a message displaying that the requirements cannot be fulfilled.

Requirements:

The user must enter a valid airport for departure and arrival. The user must provide contact information such as first name, last name,middle name(initial), email address, phone number, and a birth date.
The user must enter payment information such as payment method(credit, or debit), billing address, and contact information.

Interactions:

This module will only be called the Display Available Trains Module. This module is mainly used for user interaction.

Conflicts/Side-effects:

This module uses and references several tables to find available route tickets for purchase. Double booking may become an issue if the attributes are not updated correctly.

Flight Progress Module

Description:

This module displays a flight in progress. It will include the departure time, arrival time, and the estimated duration of the flight as well as the progress of how far into the flight it has flown.

Input:

User specified air route or route ID

Output:

Displays the specific flight in progress requested by the user. Displays the time of departure and arrival and the progress of the flight.

Requirements:
User must use valid airports and provide a minimum of 2 airports (1 departure and 1 arrival)

Interactions:
This module is used for user interaction. This module is not called upon by other modules.

Conflicts/Side-effects:
There can be multiple air routes that will satisfy the user requirements for the ticket that they would like to find. The user may sort the available air routes based on the filter preference. Or if the user has the route ID they may use it to search for the specific flight.

Airplane Operation
Start Route Module
Description:
This module notifies tables and stops the sale of tickets for a given flight once a flight has taken off.

Input:
Administrators will be able to activate this module.
Third party websites will automatically activate this module based on the time of the flight.
Output:
A Start route notification is sent to the system to notify the databases when the flight has taken off.

Requirements:
No additional requirements needed.
Route must have a start time to automatically activate this module.

Interactions:

      This module will be used with the routes to signal that the route has started and to update the table. Once a route has started ticket sales will stop and prevent the user from buying any tickets. The user will be able to see the flight in progress in the display flight module.

Conflicts/Side-effects:

      Referenced tables should be updated accordingly to avoid displaying misleading information about a flight route.

End Route Module

Description:

      This module notifies tables that the flight has ended.

Input:

      Administrators will be able to activate this module.

      Third party websites will automatically activate this module based on the time of the flight.

Output:

      An End route notification is sent to the system to notify the databases when the flight has landed.

Requirements:

      No additional requirements needed.

      Route must have a start time to automatically activate this module.

Interactions:

      This module will be used with the routes to signal that the route has ended and to update the table. A route will not have ticket sales available and prevent the user from buying any tickets. The user will be able to see the flight in progress in the flight in progress module.

Conflicts/Side-effects:

      Referenced tables should be updated accordingly to avoid displaying misleading information about a flight route.

Cancel Station Module

Description:

This module allows administrators and third party sites to cancel a flight before it has taken off.

Input:

Administrators will be able to activate this module.

Third party websites can access this module if they decide to cancel a flight.

Output:

A Canceled route notification is sent to the system to notify the databases when the flight has been canceled.

Requirements:

The canceled route must not be in progress when canceled.

Interactions:

Canceled routes will update respective tables in a specific allotted time. Ticket sales will be halted to prevent the user from buying any additional tickets.

Conflicts/Side-effects:

Referenced tables should be updated accordingly to avoid displaying misleading information about a flight route.

Administration Operation

View Active Routes Module

Description:

This module displays detailed information about specific routes, including the stops, distance, time, and cost.

Input:

Requests from the user or the administrator to view active routes.

Output:

list of active routes with details such as distance, time, cost, and availability.

Requirements:
> *the latest information from the Air Routes, Airplane, and Schedule tables in the database (to ensure the data that is shown is up to date).*

Interactions:
> *Air Routes and Schedule tables, to retrieve current routes and their schedules.*

Conflicts/Side-effects: none

Create Schedule Module

Description:
> *Will allow administrators to schedule flights by assigning airplanes to routes and defining departure and arrival times.*

Input:
> *Airplane type, route, and timing details.*

Output: A newly created schedule entry in the database.

Requirements:
> *The system should validate the availability of the airplane and the route before creating a schedule. It must also make sure there are no time conflicts on the assigned airplane or route.*

Interactions:
> *Airplane, Air Routes, and Schedule tables (ensures that the airplane is available and the route is not already overbooked or scheduled at the proposed time)*

Conflicts/Side-effects:
> *Make sure to check for overlaps, potential: overbooking or double booking*

Modify Schedule Module

Description:
> *Allows administrators to change existing flight schedules.*

Input:
> *Existing schedule details and the new changes to apply.*

Output:

> *An updated schedule within the database.*

Requirements: check for conflicts with other existing schedules and make sure that the updated information is within operational parameters.

Interactions: Works with the Schedule table, checks against the Air Routes and Airplane tables for conflicts.

Conflicts/Side-effects: passengers who already booked flights must be looked after, in case of disruption

View Route Module

Description: This module displays detailed information about specific routes, including the stops, distance, time, and cost.

Input: A request to view details of a specific route.

Output: Detailed information about the route requested.

Requirements: system should retrieve the most recent data from the Air Routes and Airports tables and display it fast.

Interactions: Pulls data from the Air Routes table, which includes a relationship to the Airports table.

Conflicts/Side-effects: None (if everything is up to date)

Create Route Module

Description: Allows administrators to define new air routes, including the origin, destination, and stops in between.

Input: Data including the airports (origin, stops, destination), distance, estimated time, and cost parameters.

*Output: A new route entry in the Air Routes table.*

Requirements: Must ensure that the input airports exist in the Airports table and that the route does not duplicate existing routes.

Interactions: Interacts with the Airports table to validate the existence of the airports and with the Air Routes table to create a new route.

Conflicts/Side-effects: creating routes that are too similar to existing ones or that don't follow aviation regulations or logistical constraints.

User Interface Design

- Home page will display the search functionality that prompts user to enter the relevant information
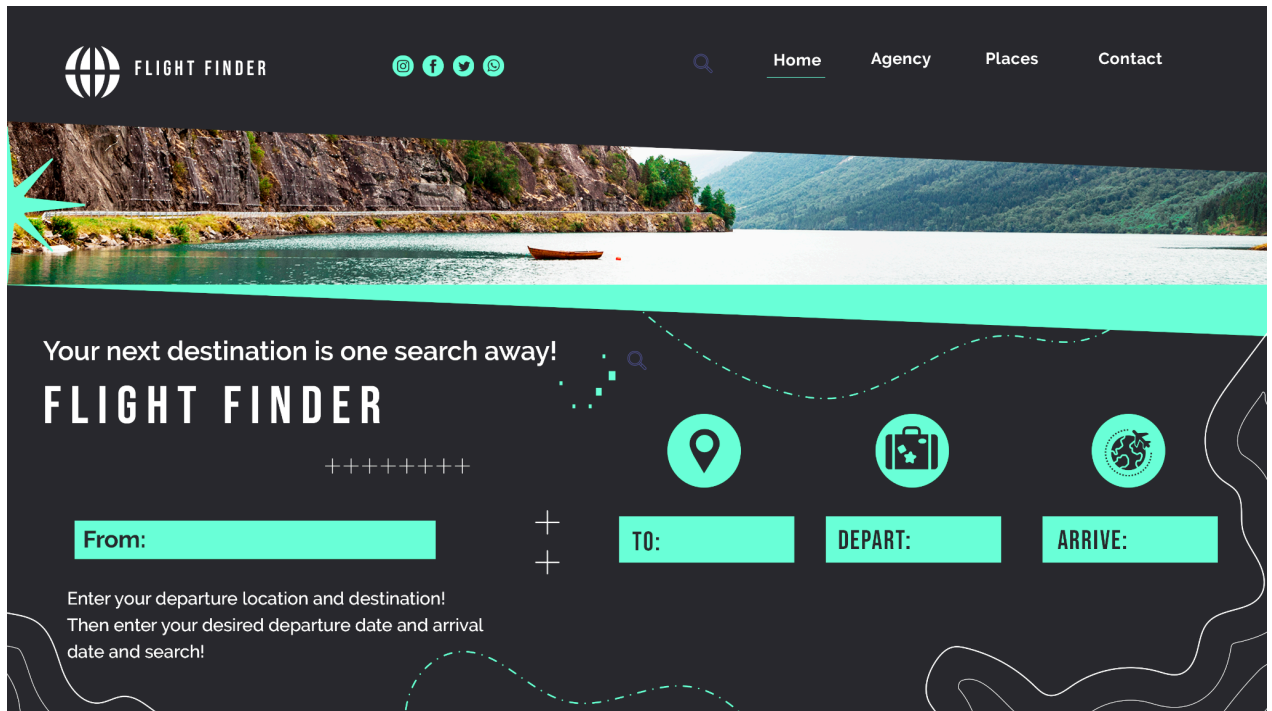
- Search page will display populated flights in accordance with the user entered data
- After selecting the flight the user desires, the user will be redirected to the airline website with necessary flight information

Implementation and Testing Plans

Phase 1:
- Setting up database: Ciara and Nate
- Web Front End Design: Christina and Tom
- Web Back End: Abhi and Megan

Phase 2:
- Link Website to database: Abhi and Megan
- Display Data to website: Christina and Tom
- Queries from website to data: Ciara and Nate

Finish:
- Utilize APIs to add real world data to the database instead of test values: Rami

Testing Plans

- Testing user input functions correctly and displays proper results
- Add ability to redirect user to respective pages that display flight information
- Ensure database is linked correctly to website

Users should be able to type in destinations and enter in departure information. From there, the website should populate correct flight information that follows the criteria. After the flights have been populated, the user should be able to select a flight they are interested in and be redirected to the page displaying booking information.