CS340 Project Step 5 Draft Version

Group 90 - Generic Team Name

Orcas Air

Matthew Wilkinson

Joseph Dehoney

Index URL: http://flip3.engr.oregonstate.edu:8251

a) Project Outline and Database Outline - Updated Version:

Overview:

- Orcas Air is a small airline that transports passengers and freight to various destinations on the West Coast. The airline started as a single airplane but has quickly expanded to a fleet of 10 various models of airplanes. The airline currently operates on a paper ledger to handle flight scheduling and would like to advance to an electronic database to keep track of their planes, pilots, and flight schedules. The airline normally operates all 10 airplanes in their fleet on a given day, with each plane having a pilot and co-pilot for each flight, and each plane only doing a maximum of two flights per day. It currently only services 6 different airports across the West Coast of the United States.

Database Outline

- **Customers**
  - CustomerID: int, auto_increment, unique, not NULL, PK
  - FirstName: varchar, not NULL
  - LastName: varchar, not NULL
  - Addressine1: varchar, not NULL
  - Addressine2: varchar, not NULL
  - City: varchar, not NULL
  - State: varchar, not NULL
  - ZIP_Code: varchar, not NULL
  - PhoneNumber: varchar, not NULL

  Relationship: a M:M relationship between *Customers* and *Flights* is implemented by way of intersection table *Flights_has_Customers* with *CustomerID* as the FK inside of *Flights_has_Customers*

- **Airplanes**
  - TailNumber: varchar, unique, not NULL, PK
  - Make: varchar, not NULL
  - Model: varchar, not NULL
  - Range: int, not NULL
  - FuelCapacity: int, not NULL

  - LastMaintenancePerformed: date, not NULL
  - MaintenanceFrequency: int, not NULL
  - NextMaintenanceDue: date, not NULL, Generated

  Relationship: a 1:M relationship between *Airplanes* and *Flights* is implemented with *Aircraft* a FK inside of *Flights.*

- **Pilots**
  - PilotID: int, auto_increment, unique, not NULL, PK
  - FirstName: varchar, not NULL
  - LastName: varchar, not NULL
  - PhoneNumber: varchar, not NULL

  Relationship: a 1:M relationship between *Pilots* and *Flights. Pilot* is implemented with *PilotID* as a FK inside of *Flights.*
  Relationship: a 1:M relationship between *Pilots* and *Flights. CoPilot* is implemented with *PilotID* as a FK inside of *Flights*.

- **Airports**
  - AirportID: varchar(3), unique, not NULL, PK
  - Name: varchar, not NULL
  - City: varchar, not NULL
  - State: varchar, not NULL

  Relationship: a 1:M relationship between *Airports* and *Flights* is implemented with *Origin and Destination* as a FK inside of *Flights.*

- **Flights**
  - FlightID: int, auto_increment, unique, not NULL, PK
  - Origin: varchar, not NULL, FK
  - Destination: varchar, not NULL, FK
  - Departure: datetime, not NULL
  - Arrival: datetime, not NULL
  - FlightDuration: int, not NULL
  - Pilot: int, not NULL, FK
  - CoPilot: int, not NULL, FK
  - Aircraft: varchar, not NULL, FK

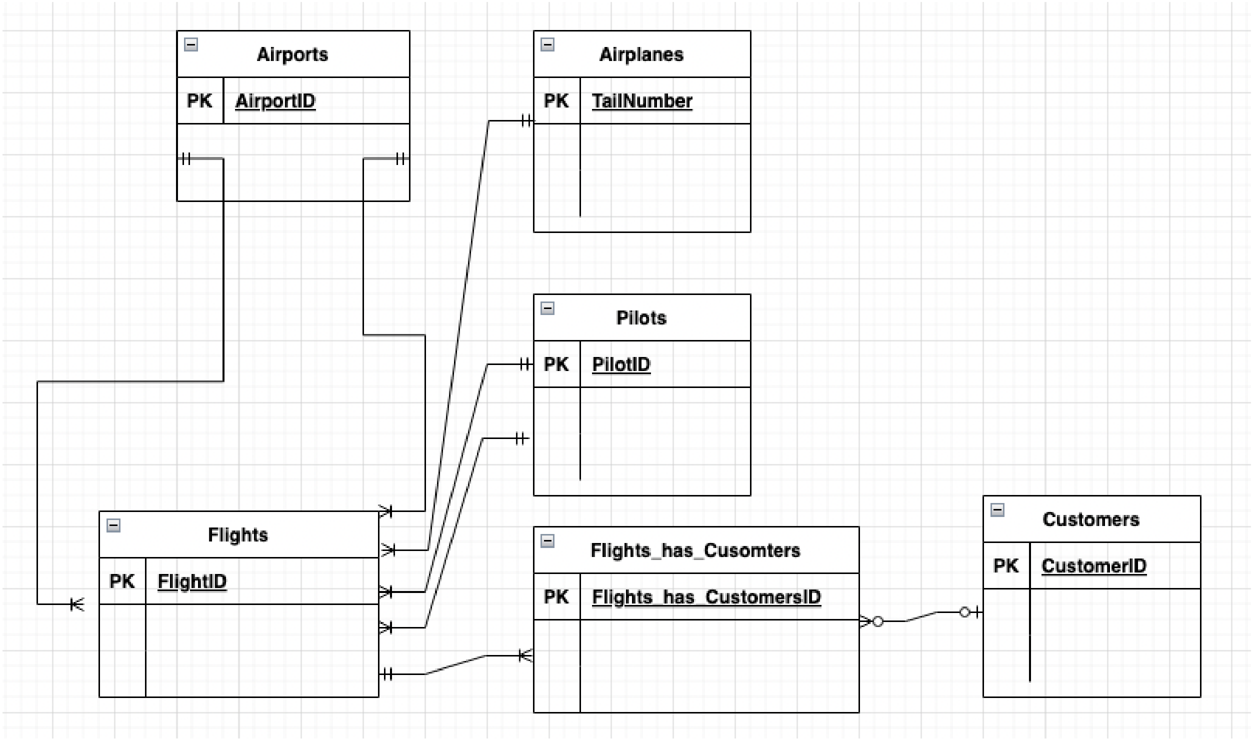
  Relationship: a M:1 relationship between *Airports* and *Flights* is implemented with *AirportID* as a FK inside of *Flights* as *Origin* and *Destination*.
  Relationship: a M:1 relationship between *Pilots* and *Flights.Pilot* is implemented with *PilotID* as a FK inside of *Flights.*
  Relationship: a M:1 relationship between *Pilots* and *Flights.CoPilot* is implemented with *PilotID* as a FK inside of *Flights.*
  Relationship: a M:1 relationship between *Airplanes* and *Flights* is implemented with *Aircraft* as a FK inside of *Flights.*
  Relationship: a M:M relationship between *Flights* and *Customers* is implemented by way of intersection table *Flights_has_Customers* with *FlightID* as the FK inside of *Flights_has_Customers.*

- **Flights_has_Customers**
  - Flights_has_CustomerID: int, auto_increment, unique, not NULL, PK
  - FlightID: int, not NULL, FK

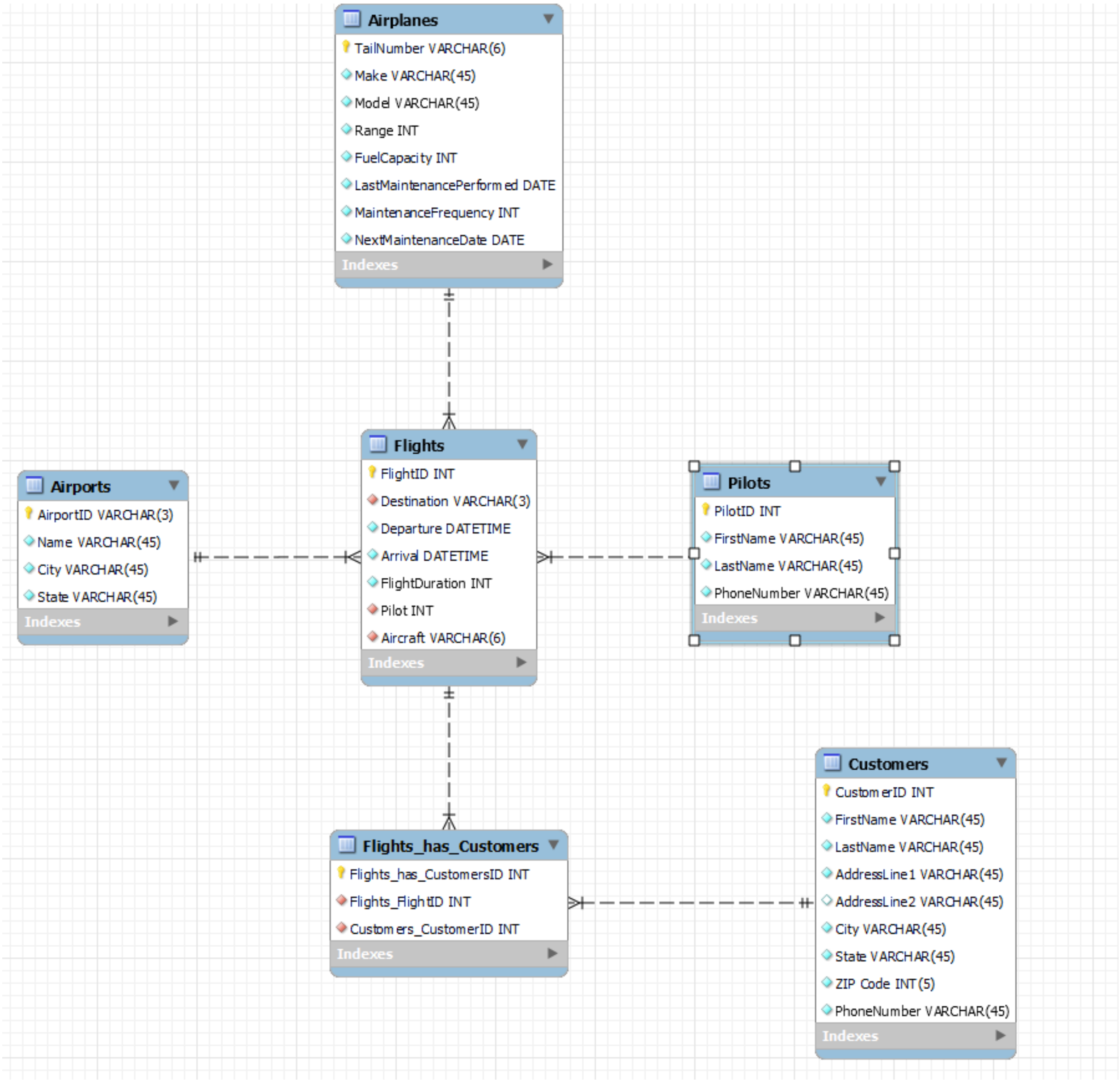
- ○ CustomerID: int, , FK

Relationship: This is an intersection table used to facilitate the M:M relationships between *Flights* and *Customers*

c) Entity-Relationship Diagram:



d) Schema:


Airplanes
TailNumber VARCHAR(6)
Make VARCHAR(45)
Model VARCHAR(45)
Range INT
FuelCapacity INT
LastMaintenancePerformed DATE
MaintenanceFrequency INT
NextMaintenanceDate DATE
Indexes
Flights
FlightID INT
Destination VARCHAR(3)
Departure DATETIME
Arrival DATETIME
FlightDuration INT
Pilot INT
Aircraft VARCHAR(6)
Indexes
Airports
AirportID VARCHAR(3)
Name VARCHAR(45)
City VARCHAR(45)
State VARCHAR(45)
Indexes
Pilots
PilotID INT
FirstName VARCHAR(45)
LastName VARCHAR(45)
PhoneNumber VARCHAR(45)
Indexes
Customers
CustomerID INT
FirstName VARCHAR(45)
LastName VARCHAR(45)
AddressLine1 VARCHAR(45)
AddressLine2 VARCHAR(45)
City VARCHAR(45)
State VARCHAR(45)
ZIP Code INT(5)
PhoneNumber VARCHAR(45)
Indexes
Flights_has_Customers
Flights_has_CustomersID INT
Flights_FlightID INT
Customers_CustomerID INT
Indexes

e) All the example data from each table in your schema should be pasted into your report. Clearly indicate from which table the data is being displayed.

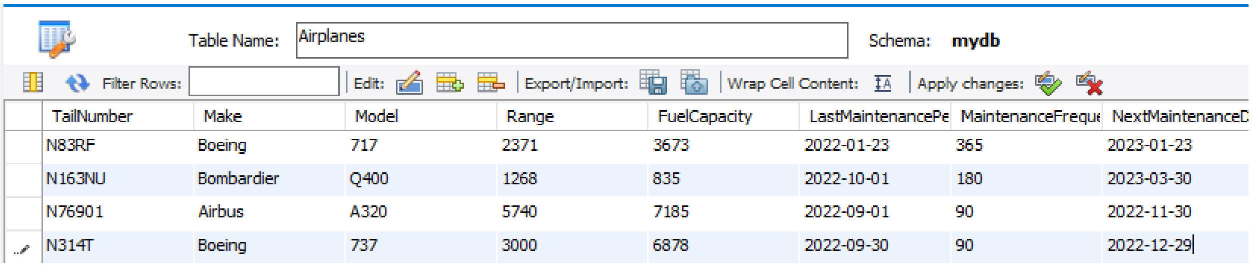


Table Name: Airplanes Schema: mydb

TailNumber	Make	Model	Range	FuelCapacity	LastMaintenancePe	MaintenanceFreque	NextMaintenanceD
N83RF	Boeing	717	2371	3673	2022-01-23	365	2023-01-23
N163NU	Bombardier	Q400	1268	835	2022-10-01	180	2023-03-30
N76901	Airbus	A320	5740	7185	2022-09-01	90	2022-11-30
N314T	Boeing	737	3000	6878	2022-09-30	90	2022-12-29

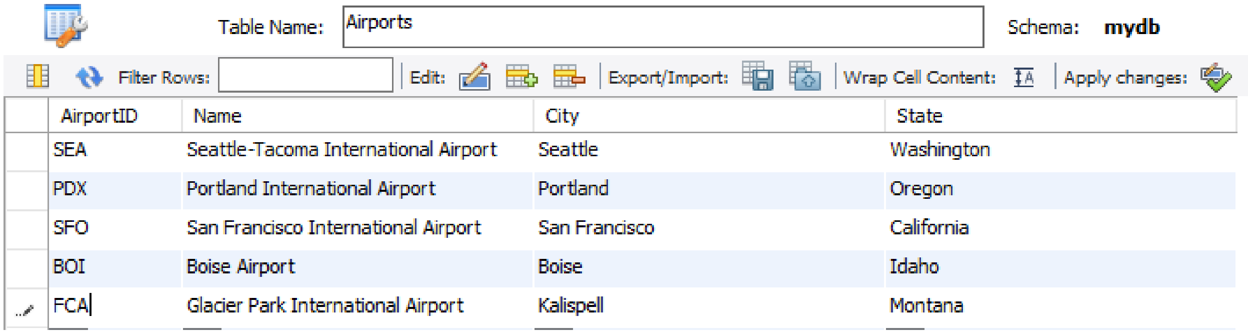


Table Name: Airports Schema: mydb

AirportID	Name	City	State
SEA	Seattle-Tacoma International Airport	Seattle	Washington
PDX	Portland International Airport	Portland	Oregon
SFO	San Francisco International Airport	San Francisco	California
BOI	Boise Airport	Boise	Idaho
FCA	Glacier Park International Airport	Kalispell	Montana

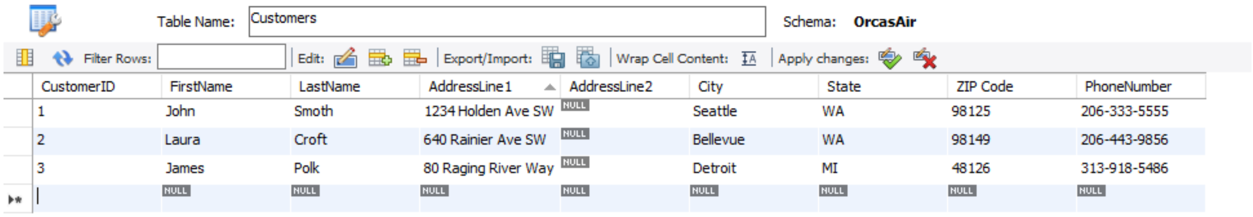


Table Name: Customers Schema: OrcasAir

CustomerID	FirstName	LastName	AddressLine1	AddressLine2	City	State	ZIP Code	PhoneNumber
1	John	Smoth	1234 Holden Ave SW	NULL	Seattle	WA	98125	206-333-5555
2	Laura	Croft	640 Rainier Ave SW	NULL	Bellevue	WA	98149	206-443-9856
3	James	Polk	80 Raging River Way	NULL	Detroit	MI	48126	313-918-5486
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

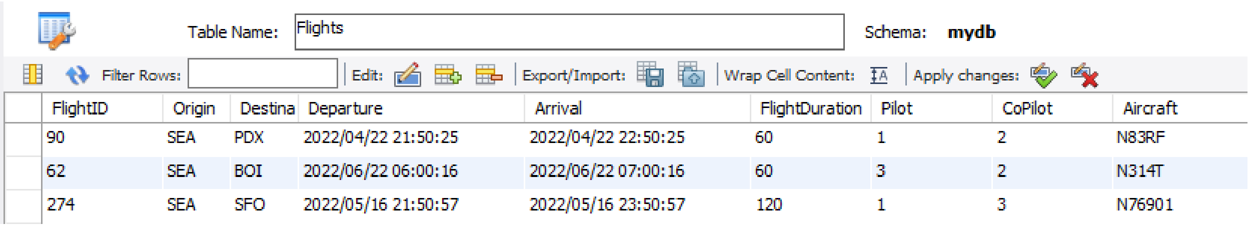


Table Name: Flights Schema: mydb

FlightID	Origin	Destina	Departure	Arrival	FlightDuration	Pilot	CoPilot	Aircraft
90	SEA	PDX	2022/04/22 21:50:25	2022/04/22 22:50:25	60	1	2	N83RF
62	SEA	BOI	2022/06/22 06:00:16	2022/06/22 07:00:16	60	3	2	N314T
274	SEA	SFO	2022/05/16 21:50:57	2022/05/16 23:50:57	120	1	3	N76901

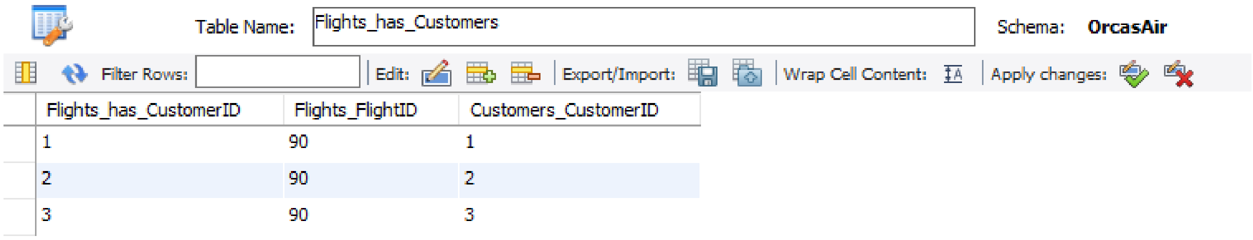


Table Name: Flights_has_Customers Schema: OrcasAir

Flights_has_CustomerID	Flights_FlightID	Customers_CustomerID
1	90	1
2	90	2
3	90	3

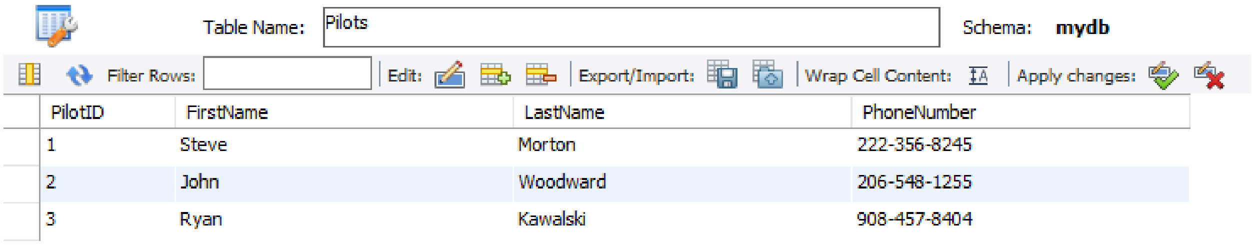


Table Name: Pilots

Schema: **mydb**

PilotID	FirstName	LastName	PhoneNumber
1	Steve	Morton	222-356-8245
2	John	Woodward	206-548-1255
3	Ryan	Kawalski	908-457-8404

b) Feedback by the peer reviewer

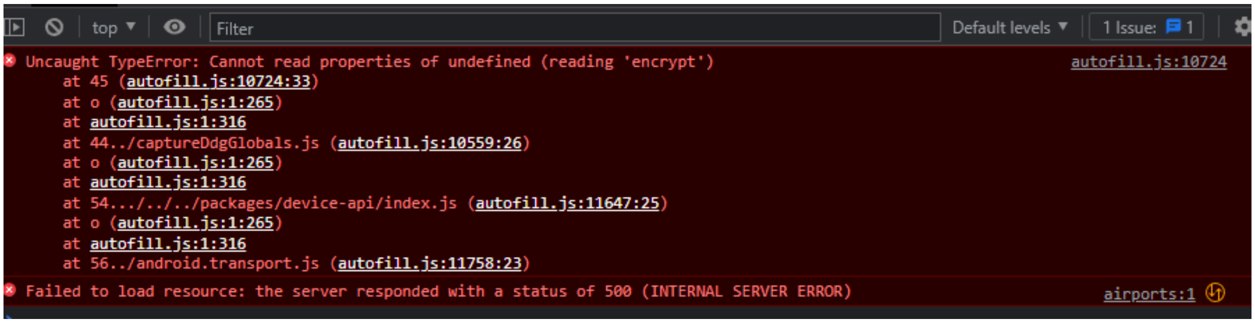
Review 1:

- **Do the implemented CRUD steps function as the team expects (e.g. if the team stated that a CRUD step worked but you found an error, please tell them)?**
  - The Create, Read, Update, and Delete operations for the Airports page all seem to be working as intended. I read in your outline that the search/filter operation will be implemented in the Flights page so I wasn't able to test that, but I imagine your team will have no issues implementing this operation in the Flights page. I'm not sure if this is important, but your project outline seems to indicate that the Name, City, and State attributes are all "not NULL". However, I am able to add a new Airport to the table without these attributes. I would think that this shouldn't be allowed when attempting to add a new Airport (or edit an existing Airport). I'm not sure how you would code this but maybe if those attributes were left blank a popup warning the user that those specific attributes are required would be nice. Also, I'm not sure if it's a problem on my end but sometimes during my review I would get an "Internal Server Error" stating that the server is overloaded. It wasn't a big deal since I was able to try out all the CRUD functions on the Airports page.
- **Would a user easily be able to use the UI to complete the step? If not or you have suggestions for how the UI can be improved, please elaborate.**
  - The Airports are displayed at the top of the page, I'm able to easily add a new airport (attributes are easily identified), and I'm able to edit and delete a specific Airport. I think the UI is very well presented and it is very easy to determine how to perform a specific operation. One thing I ran into when entering a new Airport Name was that I seemed to hit the character limit when entering the real name of a local Airport. This isn't a necessary enhancement since the site looks and performs great without it but it would be nice if I couldn't type past the character limit in the attribute fields (I'm not sure how difficult that would be to implement though since I have no idea how to do that myself).

- **What suggestions do you have for the team in any areas where they are blocked or having difficulty? Detailed helpful feedback will receive higher credit.**
    - I agree that your team certainly doesn't seem to be blocked by anything. I am very impressed with your website. It looks great and everything seems to perform as intended. I noted some suggestions in my previous comments. I don't really have anything to add besides those comments. It's unfortunate that I would sometimes encounter an "Internal Server Error" but I have no idea if that was just due to an issue on my side.

Review 2:

- **Do the implemented CRUD steps function as the team expects (e.g. if the team stated that a CRUD step worked but you found an error, please tell them)?**
    - The team stated that the Airports page CRUD operations were fully functional. However, when I tried to access the Airports page I received an internal server error so I was not able to test any of those functions. However, I was able to access the rest of the pages just fine.



```
Uncaught TypeError: Cannot read properties of undefined (reading 'encrypt')                autofill.js:10724
    at 45 (autofill.js:10724:33)
    at o (autofill.js:1:265)
    at autofill.js:1:316
    at 44../captureDdgGlobals.js (autofill.js:10559:26)
    at o (autofill.js:1:265)
    at autofill.js:1:316
    at 54.../../../packages/device-api/index.js (autofill.js:11647:25)
    at o (autofill.js:1:265)
    at autofill.js:1:316
    at 56../android.transport.js (autofill.js:11758:23)
Failed to load resource: the server responded with a status of 500 (INTERNAL SERVER ERROR)     airports:1
```

- **Would a user easily be able to use the UI to complete the step? If not or you have suggestions for how the UI can be improved, please elaborate.**
    - I was not able to test the Airports CRUD operations. The rest of the site's UI looks nice and intuitive though.
- **What suggestions do you have for the team in any areas where they are blocked or having difficulty? Detailed helpful feedback will receive higher credit.**

  - The team has stated that they are not blocked by anything so I am assuming that this error might have to do with them currently working on the project? I would suggest debugging and making sure that nothing is sending a status 500 error.

Review 3:

- **Do the implemented CRUD steps function as the team expects (e.g. if the team stated that a CRUD step worked but you found an error, please tell them)?**
  - All of the Read functions appear to work for all the pages. However, I was not able to check the airports page because I ran into an error: "Internal Server Error. The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application. "Labeled "500 Internal Service Error."
- **Would a user easily be able to use the UI to complete the step? If not or you have suggestions for how the UI can be improved, please elaborate.**
  - The UI is intuitive, makes sense and is easy to use (assuming that it functions as it is currently set up.) I like your use of placeholders and drop-down boxes.
- **What suggestions do you have for the team in any areas where they are blocked or having difficulty?**
  - Not sure what's going on with Airports, it may be related to using forever or another problem. It seems that there is a good amount of documentation on this ed page concerning forever. Otherwise it seems that you are doing a great job. Your website it coming together nicely, with just some more CRUD implementation, addressing the error for airports, and then maybe add some styling to make it prettier?

Review 4:

- **Do the implemented CRUD steps function as the team expects (e.g. if the team stated that a CRUD step worked but you found an error, please tell them)?**
  - Yes, the CRUD steps work as expected on the Airports page. I was able to insert, update and delete records from the database. I did notice that new

    - airports could be added with just an AirportID. Same thing with blanking out fields during update. This goes against the DDL, which shows every field as required. You can use the "required" attribute in the input field, and that should automatically stop a user from submitting a form when the field isn't filled in. Additionally, on the DML, I noticed that there's a search by city query. I didn't see that implemented here, but maybe it'll be used somewhere else in the ui.

- **Would a user easily be able to use the UI to complete the step? If not or you have suggestions for how the UI can be improved, please elaborate.**
    - Yes, the UI and icons used are very intuitive.
- **What suggestions do you have for the team in any areas where they are blocked or having difficulty? Detailed helpful feedback will receive higher credit.**
    - Internal Server Error came up a few times during my use of the webapp. It would eventually come back, but I'm curious what is causing the issue. For me, it seemed to start after I inserted a duplicate PK on the airports page. If it has to do with handling mysql exceptions, then it might be good to add some try/except blocks somewhere in the app so you can handle them yourself and keep the webapp available. I believe flask is currently handling them but to the user its a black box as to what is going wrong

b) Fixes based on Feedback from Previous Steps:

We received a lot of constructive feedback from our grader and our peers during the review process. Based on our last submission, there were many items that needed to be fixed or improved. Below is a cumulative list of the fixes and improvements that we made to our project.

1. We were critiqued that it was possible to add blank entries to attributes that are marked NOT NULL. To prevent this from happening, we modified our HTML to so forms whose values are inserted into a NOT NULL attribute are not required to be entered before form submission.
2. Many of our reviewers encountered a 500 Internal Service Error when attempting to test our CRUD functionality on airports. We haven’t 100% figured out the root cause of this error but we think that our link to the SQL database is timing out and not

reconnecting. This will need to be investigated further as to how we can better maintain uptime with our database connection so it doesn’t timeout.