Roberto Noel
Neal Shu
Databases
Ratan Dey

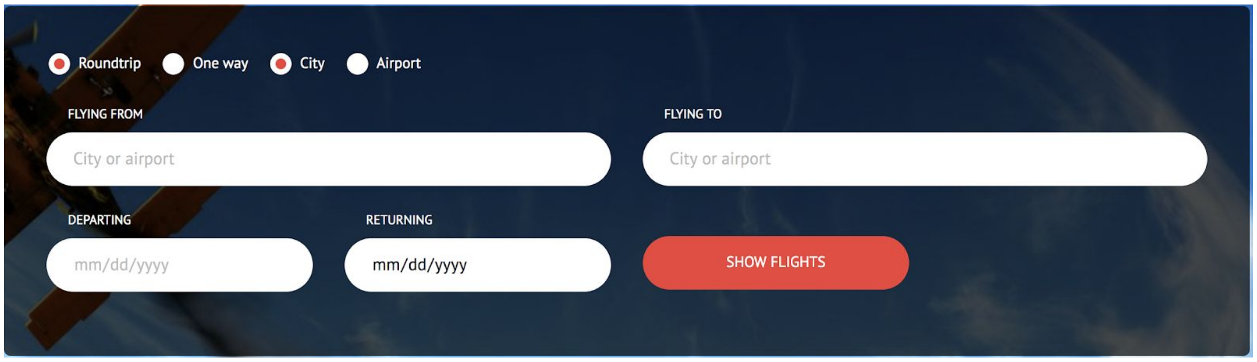
Use Case Summary With Queries

General Use Cases

1. View Public Info:

The landing page allows anyone (regardless of whether they are logged in or not) to search for one-way and round-trip flights by airport and city on specific dates.

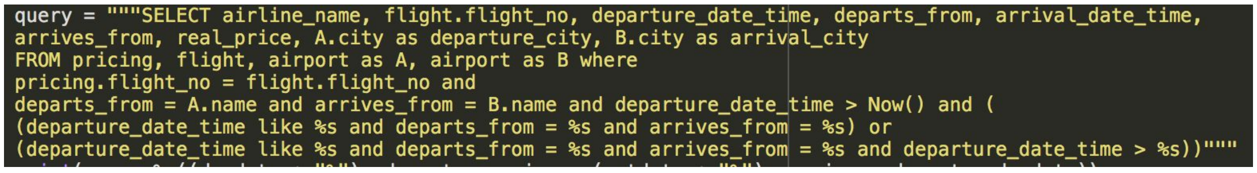
Once the user inputs data into the provided fields:



The data is retrieved through the HTML form, and four possible queries are executed.

The queries are all very similar, the only difference between airport and city is that the query uses the html form's input as an airport vs a city search. The way to get round trip info is to use an or statement so the arrival airport can also be the departure airport if the departure date is greater than the first flight's departure date.

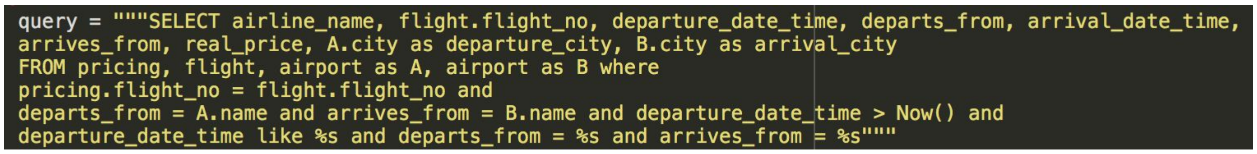
If search by airport + roundtrip:



```
query = """SELECT airline_name, flight.flight_no, departure_date_time, departs_from, arrival_date_time,
arrives_from, real_price, A.city as departure_city, B.city as arrival_city
FROM pricing, flight, airport as A, airport as B where
pricing.flight_no = flight.flight_no and
departs_from = A.name and arrives_from = B.name and departure_date_time > Now() and (
(departure_date_time like %s and departs_from = %s and arrives_from = %s) or
(departure_date_time like %s and departs_from = %s and arrives_from = %s and departure_date_time > %s))"""
```

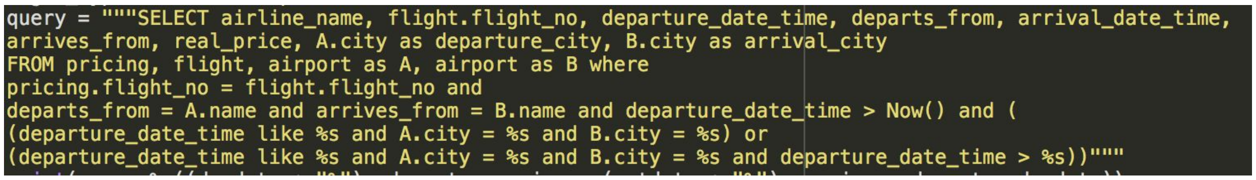
If search by airport + oneway:

Roberto Noel
Neal Shu
Databases
Ratan Dey



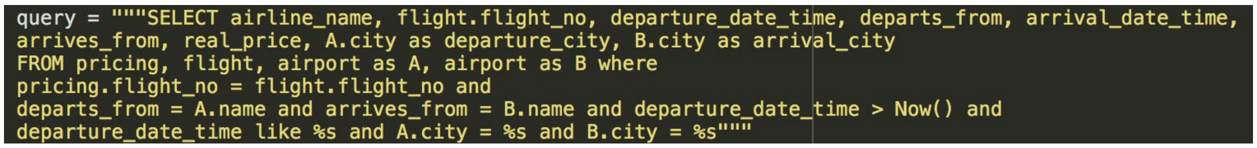
```
query = """SELECT airline_name, flight.flight_no, departure_date_time, departs_from, arrival_date_time,
arrives_from, real_price, A.city as departure_city, B.city as arrival_city
FROM pricing, flight, airport as A, airport as B where
pricing.flight_no = flight.flight_no and
departs_from = A.name and arrives_from = B.name and departure_date_time > Now() and
departure_date_time like %s and departs_from = %s and arrives_from = %s"""
```

If search by city + roundtrip:



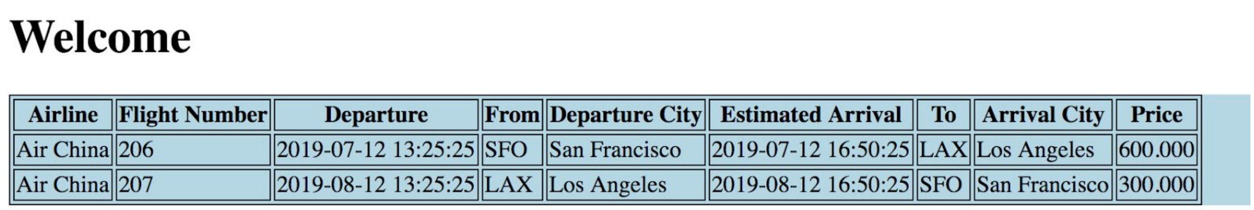
```
query = """SELECT airline_name, flight.flight_no, departure_date_time, departs_from, arrival_date_time,
arrives_from, real_price, A.city as departure_city, B.city as arrival_city
FROM pricing, flight, airport as A, airport as B where
pricing.flight_no = flight.flight_no and
departs_from = A.name and arrives_from = B.name and departure_date_time > Now() and (
(departure_date_time like %s and A.city = %s and B.city = %s) or
(departure_date_time like %s and A.city = %s and B.city = %s and departure_date_time > %s))"""
```

If search by city + oneway:



```
query = """SELECT airline_name, flight.flight_no, departure_date_time, departs_from, arrival_date_time,
arrives_from, real_price, A.city as departure_city, B.city as arrival_city
FROM pricing, flight, airport as A, airport as B where
pricing.flight_no = flight.flight_no and
departs_from = A.name and arrives_from = B.name and departure_date_time > Now() and
departure_date_time like %s and A.city = %s and B.city = %s"""
```

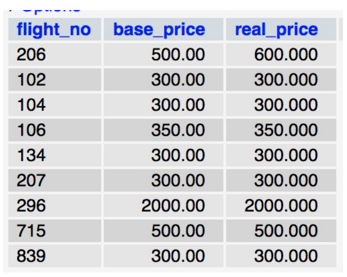
The results are displayed as follows:



Welcome

Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price
Air China	206	2019-07-12 13:25:25	SFO	San Francisco	2019-07-12 16:50:25	LAX	Los Angeles	600.000
Air China	207	2019-08-12 13:25:25	LAX	Los Angeles	2019-08-12 16:50:25	SFO	San Francisco	300.000

Note: for both the client result page and the public result page, the price displayed will be 1.2x the base price of the flight if percentage full > 70. These prices can be seen in a view named "pricing".

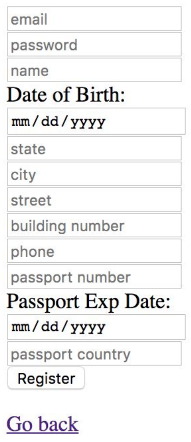


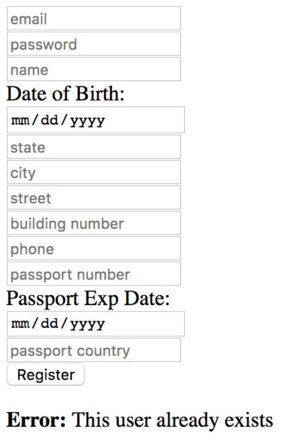
flight_no	base_price	real_price
206	500.00	600.000
102	300.00	300.000
104	300.00	300.000
106	350.00	350.000
134	300.00	300.000
207	300.00	300.000
296	2000.00	2000.000
715	500.00	500.000
839	300.00	300.000

Roberto Noel
Neal Shu
Databases
Ratan Dey

2. Customer Register:

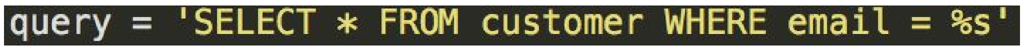
New customers can register using their email along with additional personal information that they fill out on the following form:





Error: This user already exists

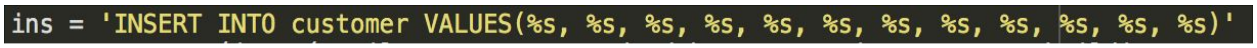
First, python retrieves the data from the HTML form and checks if there is already a user using the following query:



```
query = 'SELECT * FROM customer WHERE email = %s'
```

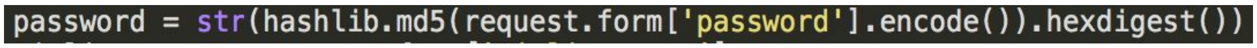
If there is data in the query, the page is refreshed and error message above is displayed.

Otherwise, the data is input into the database with the following insert query:



```
ins = 'INSERT INTO customer VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)'
```

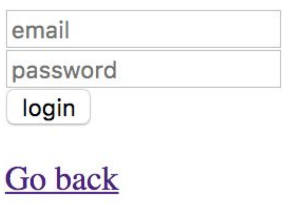
Note: when passwords are registered or checked against the database, they are hashed using MD5 as follows:

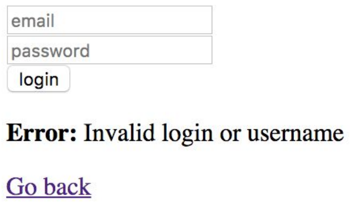


```
password = str(hashlib.md5(request.form['password'].encode()).hexdigest())
```

3. Customer Login:

Existing clients can register by filling out their email and password in the following form:

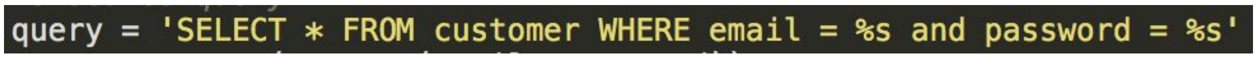




Error: Invalid login or username

Roberto Noel
Neal Shu
Databases
Ratan Dey

Python retrieves the login data from the HTML form and queries to see if there is a match as follows:

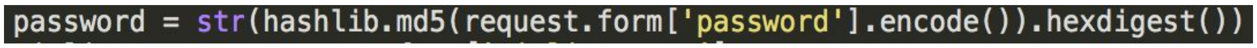


```
query = 'SELECT * FROM customer WHERE email = %s and password = %s'
```

If there is a match, the customer is redirected to the client home and their email is stored in the session.

Otherwise, the page refreshes with the above error message.

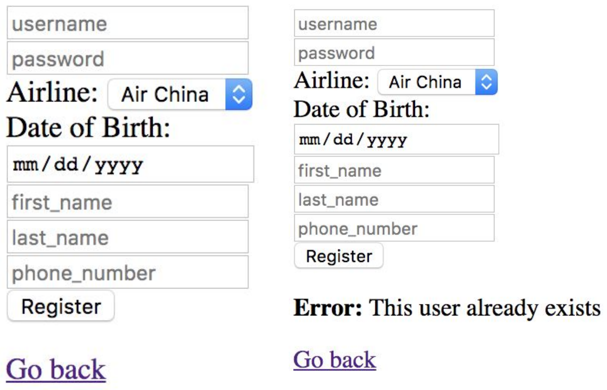
Note: when passwords are registered or checked against the database, they are hashed using MD5 as follows:



```
password = str(hashlib.md5(request.form['password'].encode()).hexdigest())
```

4. Staff Register:

New airline staff can register using their email along with additional personal information that they fill out on the following form:



username
password
Airline: Air China
Date of Birth:
mm/dd/yyyy
first_name
last_name
phone_number
Register

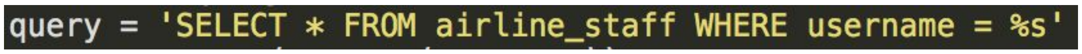
Go back

username
password
Airline: Air China
Date of Birth:
mm/dd/yyyy
first_name
last_name
phone_number
Register

Error: This user already exists

Go back

First, python retrieves the data from the HTML form and checks if there is already a user using the following query:



```
query = 'SELECT * FROM airline_staff WHERE username = %s'
```

If there is data in the query, the page is refreshed and error message above is displayed.

Otherwise, the data is input into the database with the following insert query:

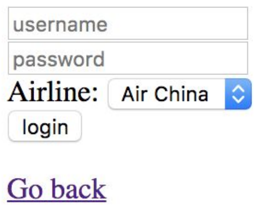
Roberto Noel
Neal Shu
Databases
Ratan Dey

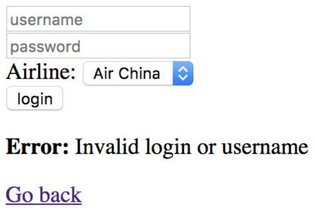
```
ins = 'INSERT INTO airline_staff VALUES(%s, %s, %s, %s, %s, %s)'
```

Note: when passwords are registered or checked against the database, they are hashed using MD5 as follows:

```
password = str(hashlib.md5(request.form['password'].encode()).hexdigest())
```

5. Staff Login:
Existing staff can login by filling out their email, password and airline in the following form:





Error: Invalid login or username

Python retrieves the login data from the HTML form and queries to see if there is a match as follows:

```
query = 'SELECT * FROM airline_staff WHERE username = %s and password = %s and airline_name = %s'
```

If there is a match, the staff member is redirected to the staff home and their email and airline name is stored in the session.

Otherwise, the page refreshes with the above error message.

Note: when passwords are registered or checked against the database, they are hashed using MD5 as follows:

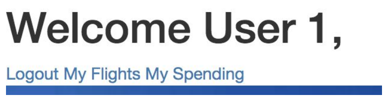
```
password = str(hashlib.md5(request.form['password'].encode()).hexdigest())
```

Customer Use Cases

1. View My Flights:

Roberto Noel
Neal Shu
Databases
Ratan Dey

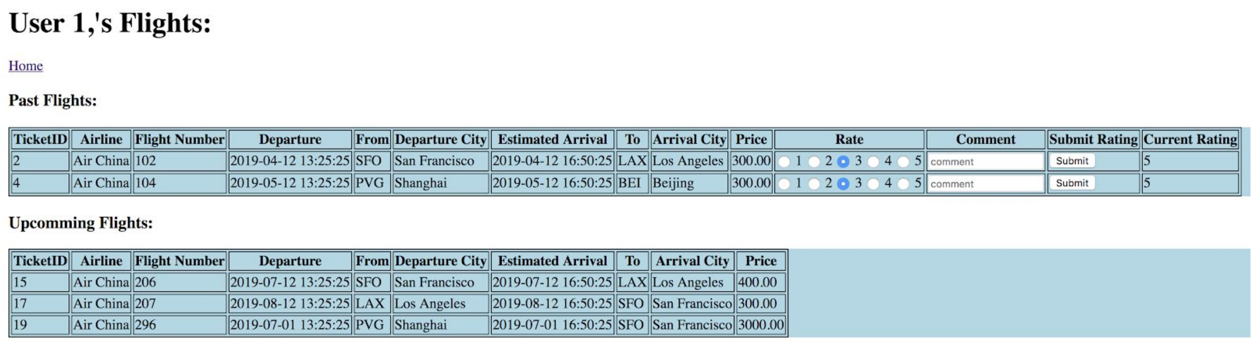
Once a customer is logged in, he/she can view the flights they have already purchased by clicking the “My Flights” button on their homepage:



Welcome User 1,

Logout My Flights My Spending

This redirects them to the following page which shows past and upcoming flights:



User 1,'s Flights:

Home

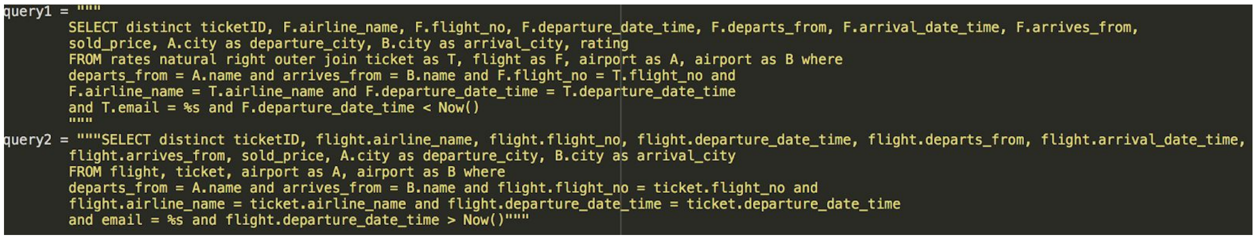
Past Flights:

TicketID	Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price	Rate	Comment	Submit Rating	Current Rating
2	Air China	102	2019-04-12 13:25:25	SFO	San Francisco	2019-04-12 16:50:25	LAX	Los Angeles	300.00	1 2 [✓ 3] 4 5	comment	Submit	5
4	Air China	104	2019-05-12 13:25:25	PVG	Shanghai	2019-05-12 16:50:25	BEI	Beijing	300.00	1 2 [✓ 3] 4 5	comment	Submit	5

Upcomming Flights:

TicketID	Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price
15	Air China	206	2019-07-12 13:25:25	SFO	San Francisco	2019-07-12 16:50:25	LAX	Los Angeles	400.00
17	Air China	207	2019-08-12 13:25:25	LAX	Los Angeles	2019-08-12 16:50:25	SFO	San Francisco	300.00
19	Air China	296	2019-07-01 13:25:25	PVG	Shanghai	2019-07-01 16:50:25	SFO	San Francisco	3000.00

To get these results, two queries are used, one for the past flights and one for the upcoming flights:



```
query1 = """
        SELECT distinct ticketID, F.airline_name, F.flight_no, F.departure_date_time, F.departs_from, F.arrival_date_time, F.arrives_from,
        sold_price, A.city as departure_city, B.city as arrival_city, rating
        FROM rates natural right outer join ticket as T, flight as F, airport as A, airport as B where
        departs_from = A.name and arrives_from = B.name and F.flight_no = T.flight_no and
        F.airline_name = T.airline_name and F.departure_date_time = T.departure_date_time
        and T.email = %s and F.departure_date_time < Now()
        """
query2 = """SELECT distinct ticketID, flight.airline_name, flight.flight_no, flight.departure_date_time, flight.departs_from, flight.arrival_date_time,
        flight.arrives_from, sold_price, A.city as departure_city, B.city as arrival_city
        FROM flight, ticket, airport as A, airport as B where
        departs_from = A.name and arrives_from = B.name and flight.flight_no = ticket.flight_no and
        flight.airline_name = ticket.airline_name and flight.departure_date_time = ticket.departure_date_time
        and email = %s and flight.departure_date_time > Now()"""
```

The first query simply searches for ticket purchases with departing dates prior to Now() with the user’s email on them and joins that data with the flight data to display what is shown on the HTML page.

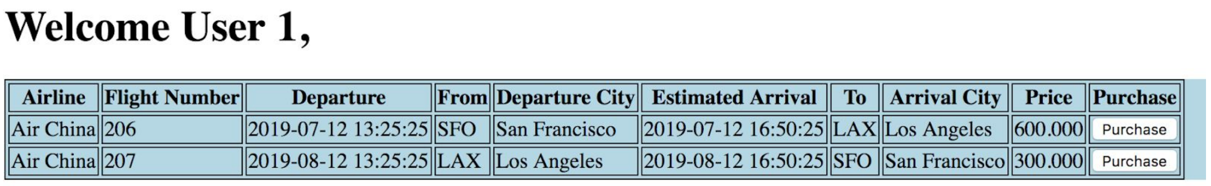
The second does the same thing but for tickets with departing dates greater than Now()

2. Search for Flights:

Roberto Noel
Neal Shu
Databases
Ratan Dey

Customers can search for flights from their home page. The search function runs exactly the same as the public search function except that when python detects an email in the session, an alternate results page is rendered.

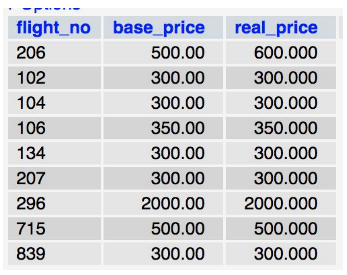
Unlike in the public results page, this results page allows the client to purchase flights because their email is in the session.



Welcome User 1,

Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price	Purchase
Air China	206	2019-07-12 13:25:25	SFO	San Francisco	2019-07-12 16:50:25	LAX	Los Angeles	600.000	Purchase
Air China	207	2019-08-12 13:25:25	LAX	Los Angeles	2019-08-12 16:50:25	SFO	San Francisco	300.000	Purchase

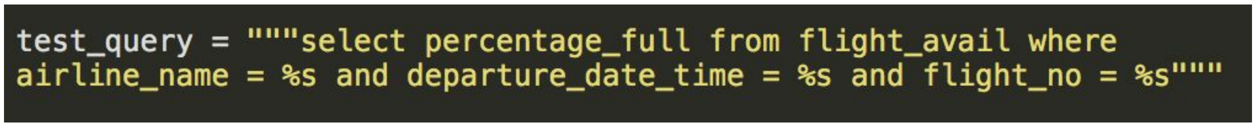
Note: for both the client result page and the public result page, the price displayed will be 1.2x the base price of the flight if percentage full > 70. These prices can be seen in a view named “pricing”.



flight_no	base_price	real_price
206	500.00	600.000
102	300.00	300.000
104	300.00	300.000
106	350.00	350.000
134	300.00	300.000
207	300.00	300.000
296	2000.00	2000.000
715	500.00	500.000
839	300.00	300.000

3. Purchase Tickets:

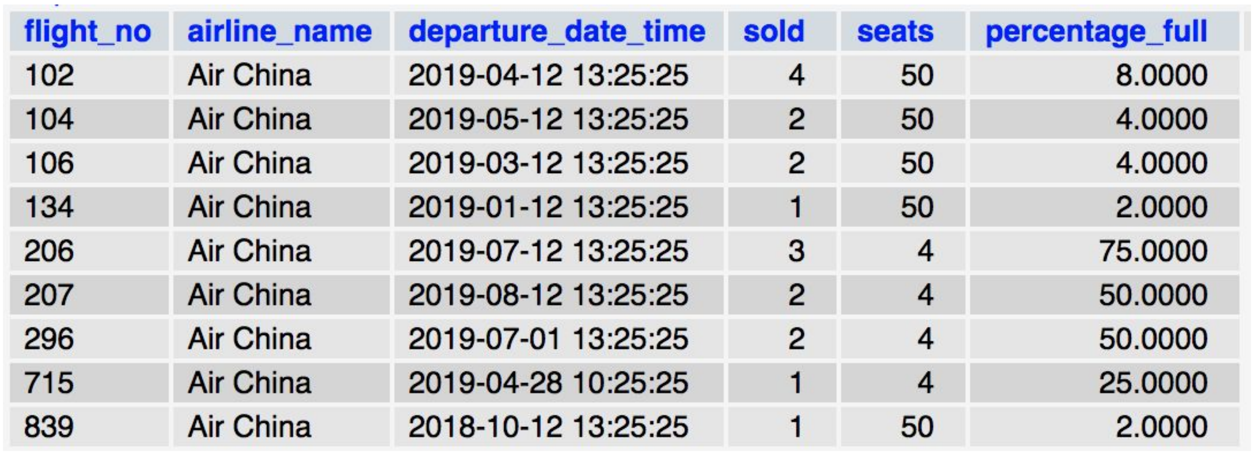
As seen above, existing customers can purchase tickets. Once a customer clicks “purchase” next to the flight they want to buy, the flight is tested to see whether it is full or not using the following query:



```
test_query = """select percentage_full from flight_avail where
airline_name = %s and departure_date_time = %s and flight_no = %s"""
```

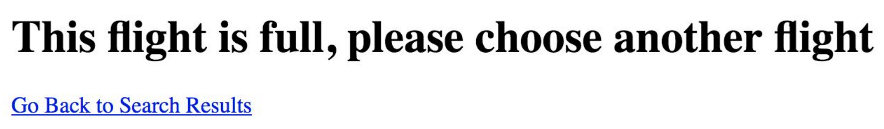
This query checks a view named flight_avail which displays the number of seats on a flight, the number of tickets purchased for that flight, and the percentage of seats that have been sold along with the primary keys for flight:

Roberto Noel
Neal Shu
Databases
Ratan Dey



flight_no	airline_name	departure_date_time	sold	seats	percentage_full
102	Air China	2019-04-12 13:25:25	4	50	8.0000
104	Air China	2019-05-12 13:25:25	2	50	4.0000
106	Air China	2019-03-12 13:25:25	2	50	4.0000
134	Air China	2019-01-12 13:25:25	1	50	2.0000
206	Air China	2019-07-12 13:25:25	3	4	75.0000
207	Air China	2019-08-12 13:25:25	2	4	50.0000
296	Air China	2019-07-01 13:25:25	2	4	50.0000
715	Air China	2019-04-28 10:25:25	1	4	25.0000
839	Air China	2018-10-12 13:25:25	1	50	2.0000

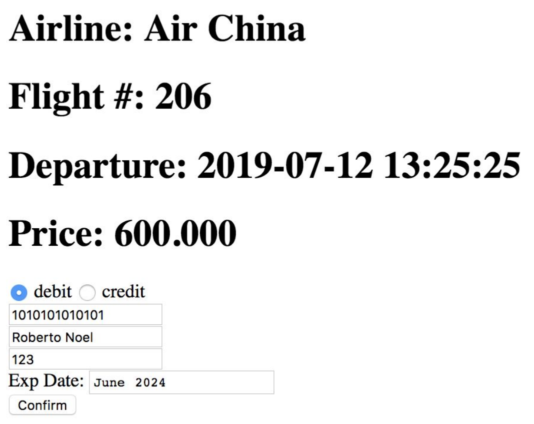
If the query returns percentage_full = 100, the client is redirected to a page telling them that the flight is full:



This flight is full, please choose another flight

Go Back to Search Results

Otherwise, the client is redirected to a page where he/she can fill out their payment information to complete the purchase:



Airline: Air China

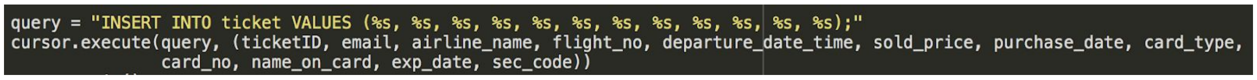
Flight #: 206

Departure: 2019-07-12 13:25:25

Price: 600.000

[✓ debit] credit
1010101010101
Roberto Noel
123
Exp Date: June 2024
Confirm

Once the client fills out the fields and hits confirmed, the following query is executed with that information:

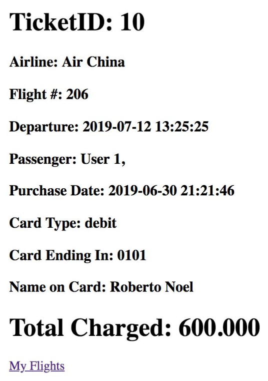


```
query = "INSERT INTO ticket VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s);"
cursor.execute(query, (ticketID, email, airline_name, flight_no, departure_date_time, sold_price, purchase_date, card_type,
               card_no, name_on_card, exp_date, sec_code))
```

This query sends the ticket information to the database, confirming the purchase.
Note: the expiration date is stored as the input month and year + “-01”

Roberto Noel
Neal Shu
Databases
Ratan Dey

After this, the client is redirected to a confirmation page with all of his/her useful information:



TicketID: 10

Airline: Air China

Flight #: 206

Departure: 2019-07-12 13:25:25

Passenger: User 1,

Purchase Date: 2019-06-30 21:21:46

Card Type: debit

Card Ending In: 0101

Name on Card: Roberto Noel

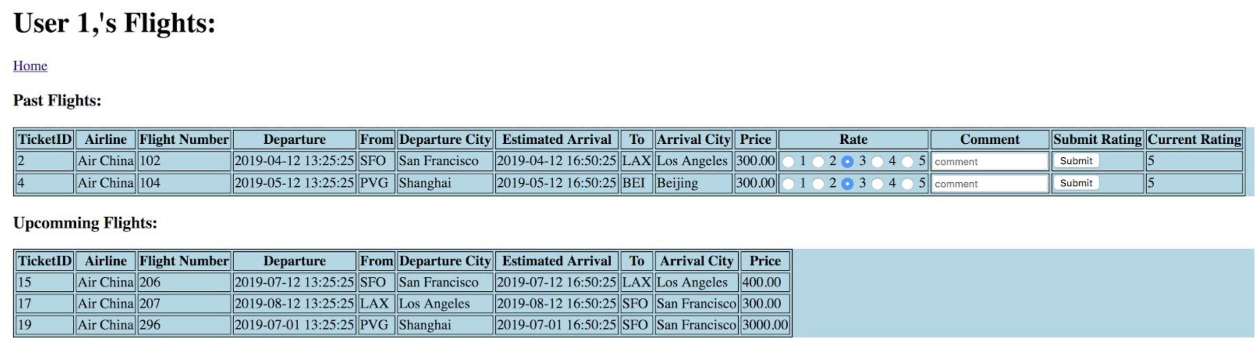
Total Charged: 600.000

My Flights

They can then see this flight on "My_Flights"

4. Give Ratings and Comments on Previous Flights:

As you may have noticed in the "My_Flights" tab, customers have the ability to rate and comment on their previous flights.



User 1,'s Flights:

Home

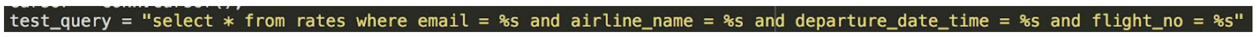
Past Flights:

TicketID	Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price	Rate	Comment	Submit Rating	Current Rating
2	Air China	102	2019-04-12 13:25:25	SFO	San Francisco	2019-04-12 16:50:25	LAX	Los Angeles	300.00	1 2 [✓ 3] 4 5	comment	Submit	5
4	Air China	104	2019-05-12 13:25:25	PVG	Shanghai	2019-05-12 16:50:25	BEI	Beijing	300.00	1 2 [✓ 3] 4 5	comment	Submit	5

Upcomming Flights:

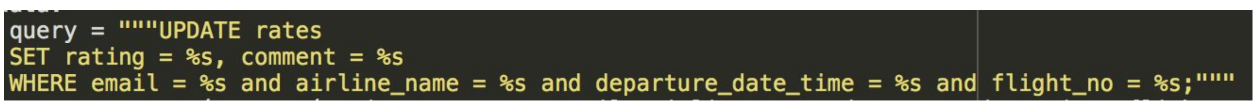
TicketID	Airline	Flight Number	Departure	From	Departure City	Estimated Arrival	To	Arrival City	Price
15	Air China	206	2019-07-12 13:25:25	SFO	San Francisco	2019-07-12 16:50:25	LAX	Los Angeles	400.00
17	Air China	207	2019-08-12 13:25:25	LAX	Los Angeles	2019-08-12 16:50:25	SFO	San Francisco	300.00
19	Air China	296	2019-07-01 13:25:25	PVG	Shanghai	2019-07-01 16:50:25	SFO	San Francisco	3000.00

Once they fill out the form above with their rating and comment, they may hit the submit button. Then, the following query is executed to test if the flight has already been rated before by them:



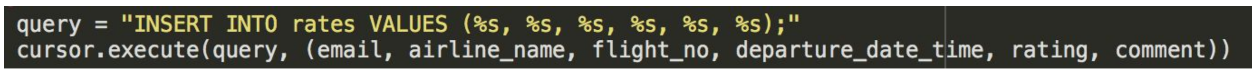
```
test_query = "select * from rates where email = %s and airline_name = %s and departure_date_time = %s and flight_no = %s"
```

If this query returns data, the rating is updated using the following query:



```
query = """UPDATE rates
SET rating = %s, comment = %s
WHERE email = %s and airline_name = %s and departure_date_time = %s and flight_no = %s;"""
```

Otherwise, a new rating is input into the system with the following query:



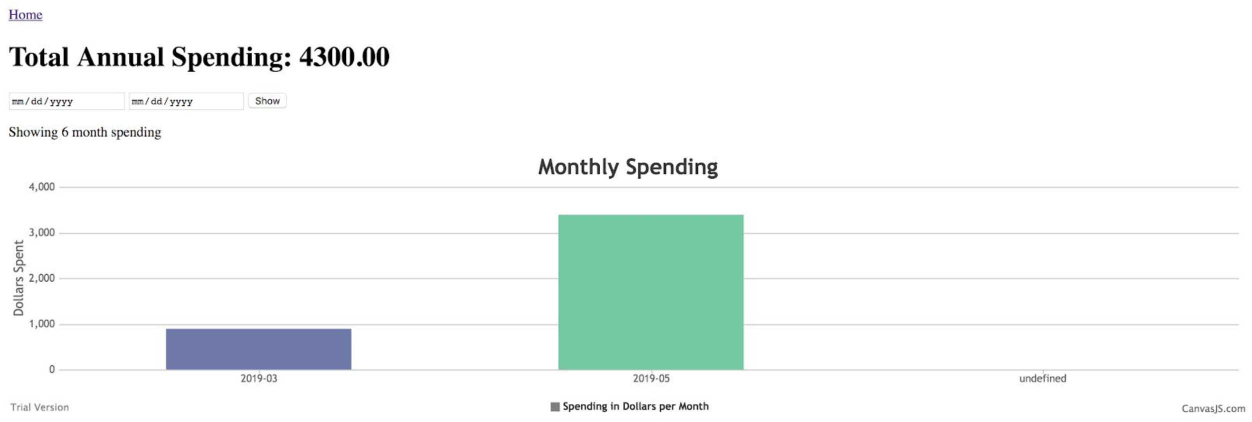
```
query = "INSERT INTO rates VALUES (%s, %s, %s, %s, %s, %s);"
cursor.execute(query, (email, airline_name, flight_no, departure_date_time, rating, comment))
```

After the rating is inserted or updated, the page refreshes with their new rating visible in the past flights table.

Note: In the table display of previous flights, I used a natural right outer join which will make the rating “null” (“None” in HTML) if it hasn’t been rated yet.

5. Track My Spending:

Clients may view their past spending by clicking the “My Spending” tab:

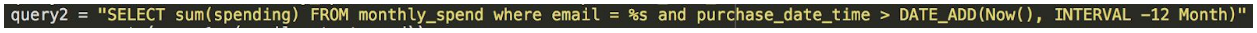


Here, their total annual spending is displayed along with a 6 month summary of their monthly spending on a bar chart created with Chart.js. Clients are also able to select a date range for the spending that they want displayed on the chart.

Two queries are needed for the initial rendering of this page, both use the “monthly_spend” view which pulls ticket data to show how much each user spends per month.

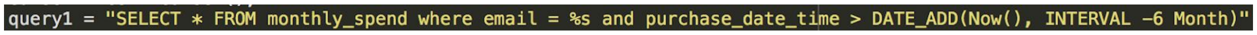
Roberto Noel
Neal Shu
Databases
Ratan Dey

The first query requests the sum of all the monthly spending for the specific client between Now() - 12 months and Now():



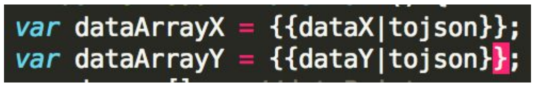
```
query2 = "SELECT sum(spending) FROM monthly_spend where email = %s and purchase_date_time > DATE_ADD(Now(), INTERVAL -12 Month)"
```

The second query requests all the monthly data for the specific client between Now() - 6 months and Now():



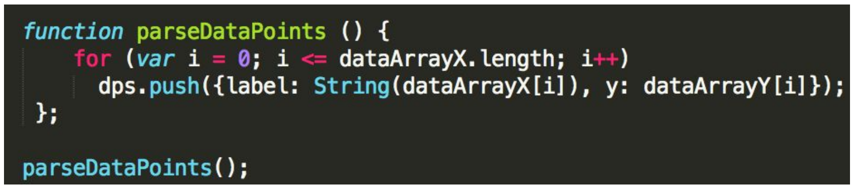
```
query1 = "SELECT * FROM monthly_spend where email = %s and purchase_date_time > DATE_ADD(Now(), INTERVAL -6 Month)"
```

The data from the first query is simply displayed on the screen as text, while the data from the other query is passed to a javascript function embedded in the HTML of the file, to do this we must convert the data to JSON format with Flask:



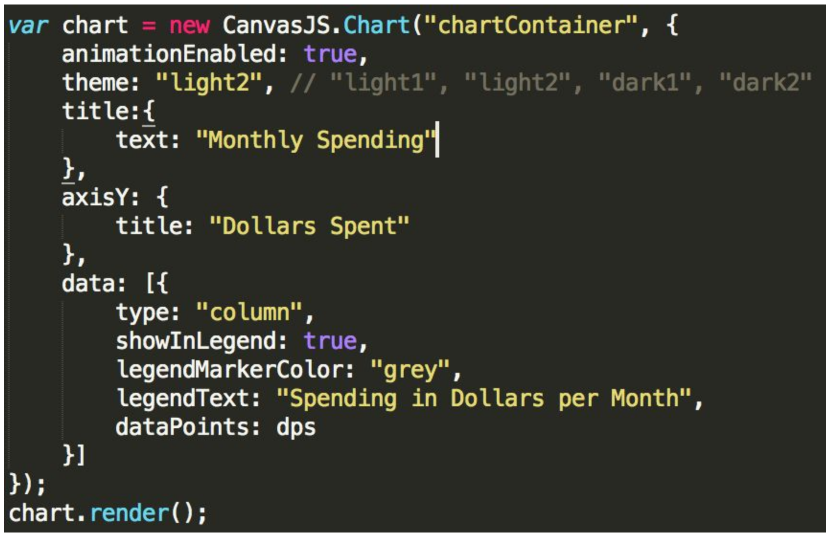
```
var dataArrayX = {{dataX|tojson}};
var dataArrayY = {{dataY|tojson}};
```

We then parse the data points from the JSON into a format that canvas JS can use:



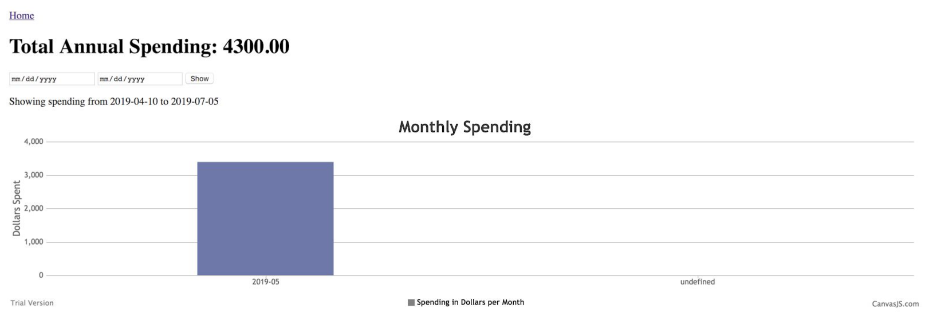
```
function parseDataPoints () {
    for (var i = 0; i <= dataArrayX.length; i++)
      dps.push({label: String(dataArrayX[i]), y: dataArrayY[i]});
};

parseDataPoints();
```

The chart is then displayed with canvas JS:



```
var chart = new CanvasJS.Chart("chartContainer", {
    animationEnabled: true,
    theme: "light2", // "light1", "light2", "dark1", "dark2"
    title:{
        text: "Monthly Spending"
    },
    axisY: {
        title: "Dollars Spent"
    },
    data: [{
        type: "column",
        showInLegend: true,
        legendMarkerColor: "grey",
        legendText: "Spending in Dollars per Month",
        dataPoints: dps
    }]
});
chart.render();
```

Roberto Noel
Neal Shu
Databases
Ratan Dey
The last feature of this page is to be able to select date ranges for the chart. As you can see below, here I restricted the range to only see the month of May:

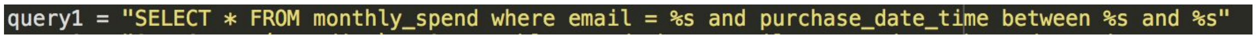

Home

Total Annual Spending: 4300.00

mm/dd/yyyy mm/dd/yyyy Show

Showing spending from 2019-04-10 to 2019-07-05

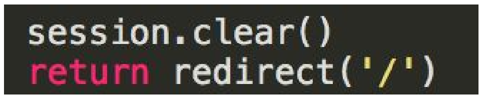
For this we simply get the data from the form above the chart, and use the following query to get the spending between those two dates:


```
query1 = "SELECT * FROM monthly_spend where email = %s and purchase_date_time between %s and %s"
```

We then pass the data from this query to flask and reload the page.

6. Logout:

Customers can logout from the client home page, this simply pops all of the session variables and redirect them to the landing page:


```
session.clear()
return redirect('/')
```

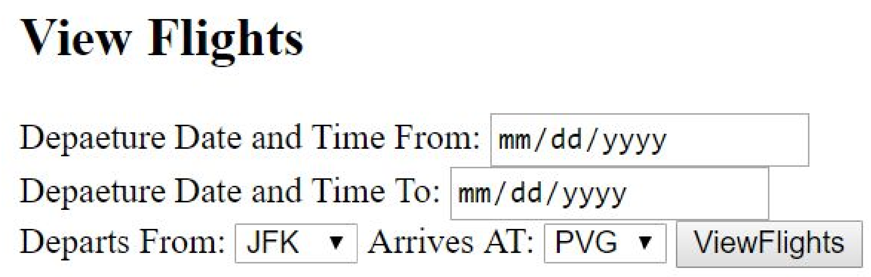
Staff Use Cases

1. View flights:

After a staff is logged in, the flights info within the next 30 days is displayed in the staff home page, with the following query “SELECT * FROM flight WHERE airline_name = %s and departure_date_time BETWEEN Now() and DATE_ADD(NOW(), INTERVAL 30 DAY)”

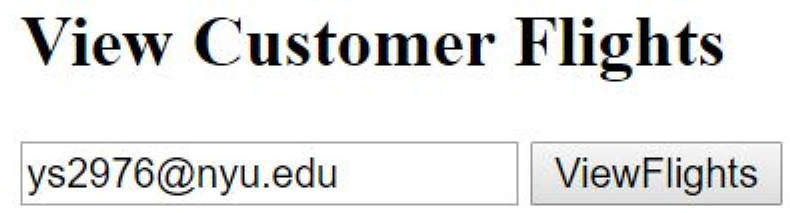
On the staff home page, a staff can use view flights section to see flights of that specific airline based on the range of dates and departure and arrival destinations that the staff inputs

Roberto Noel
Neal Shu
Databases
Ratan Dey


View Flights
Depaeture Date and Time From: mm/dd/yyyy
Depaeture Date and Time To: mm/dd/yyyy
Departs From: JFK ▼ Arrives AT: PVG ▼ ViewFlights

With the following query: "SELECT flight_no, departure_date_time, departs_from, arrival_date_time, arrives_from, base_price, flight_status
FROM flight
WHERE airline_name = %s AND departure_date_time BETWEEN %s AND %s AND departs_from = %s AND arrives_from = %s"

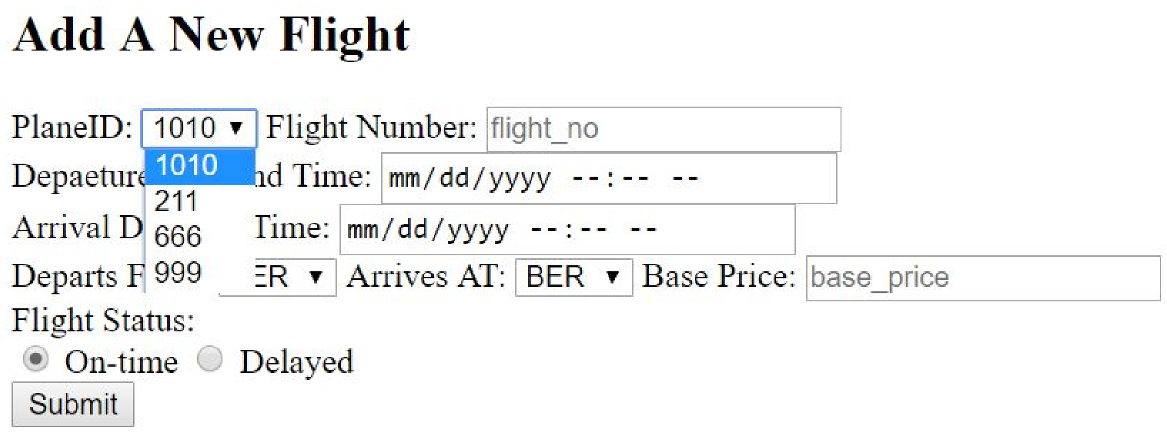
A staff is also able to see flights taken by a specific customer by inputting his or her email



With the following query: "SELECT airline_name, flight_no, departure_date_time, email
FROM ticket
WHERE airline_name = %s and email = %s"

2. Create new flights:
A staff will only be able to add new flights within his or her airline, and to prevent unauthorized action, a dropdown menu for planeID under that specific airline is displayed.

Roberto Noel
Neal Shu
Databases
Ratan Dey

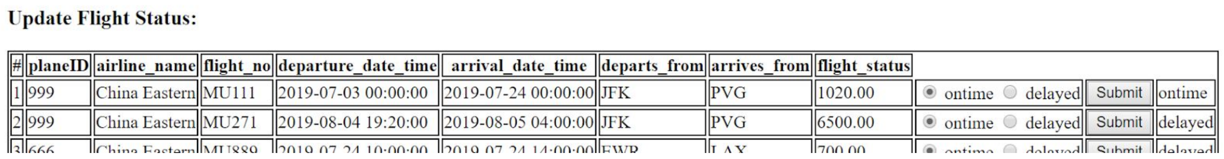


Add A New Flight

PlaneID: 1010 ▾ Flight Number: flight_no
Depaeture 1010 nd Time: mm/dd/yyyy --:-- --
Arrival D 211 Time: mm/dd/yyyy --:-- --
Departs F 666 ER ▾ Arrives AT: BER ▾ Base Price: base_price
999
Flight Status:
● On-time ○ Delayed
Submit

With the following query:"INSERT INTO flight (\`planeID\`, \`airline_name\`, \`flight_no\`, \`departure_date_time\`, \`arrival_date_time\`, \`departs_from\`, \`arrives_from\`, \`base_price\`, \`flight_status\`) VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s)"

3. Change Status of flights:
A staff is able to change flight status of a flight in the /ChangeFlightStatus page, The page displays all the data of flights within that airline with two radios for 'ontime' and 'delayed' and a submit button to commit the change.



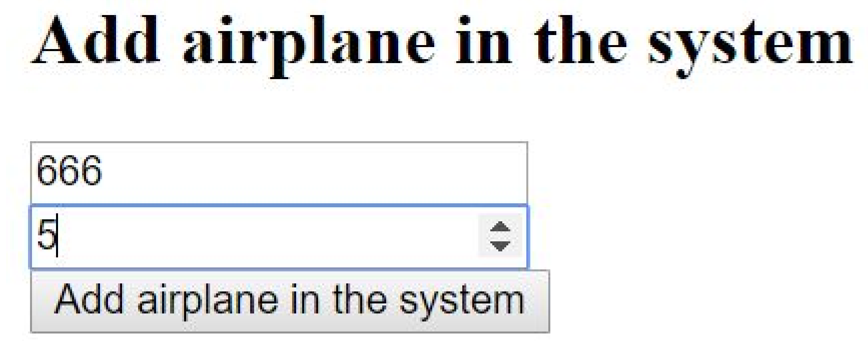
Update Flight Status:

#	planeID	airline_name	flight_no	departure_date_time	arrival_date_time	departs_from	arrives_from	flight_status			
1	999	China Eastern	MU111	2019-07-03 00:00:00	2019-07-24 00:00:00	JFK	PVG	1020.00	● ontime ○ delayed	Submit	ontime
2	999	China Eastern	MU271	2019-08-04 19:20:00	2019-08-05 04:00:00	JFK	PVG	6500.00	● ontime ○ delayed	Submit	delayed
3	666	China Eastern	MU889	2019-07-24 10:00:00	2019-07-24 14:00:00	EWR	LAX	700.00	● ontime ○ delayed	Submit	delayed

With the following query:"UPDATE flight
SET flight_status = %s
WHERE airline_name = %s and flight_no = %s and departure_date_time = %s"

4. Add airplane in the system:
In staff home page, a staff is allowed to add new airplane by inputting its planeID and number of seats, and new airplanes can only be added to the airline that the staff works for since airline_name is grabbed from the session.

Roberto Noel
Neal Shu
Databases
Ratan Dey


Add airplane in the system

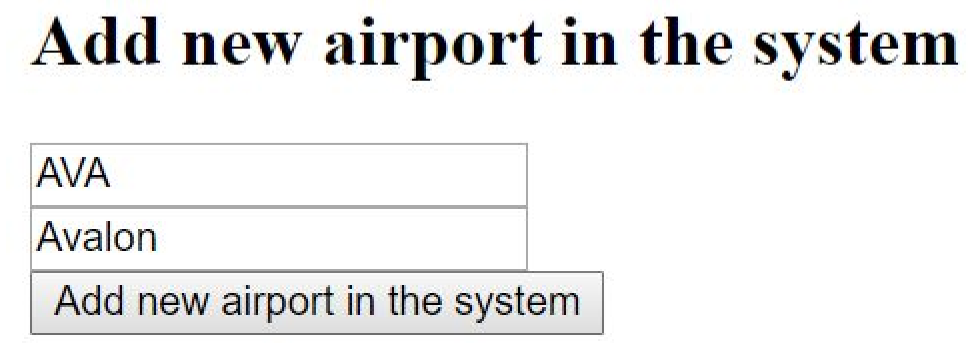
666

5

With the following query:"INSERT INTO `airplane` (`airline_name`, `planeID`, `seats`) VALUES (%s, %s, %s)"

5. Add new airport in the system:

A staff is able to add a new airport by inputting the name and city


Add new airport in the system

AVA

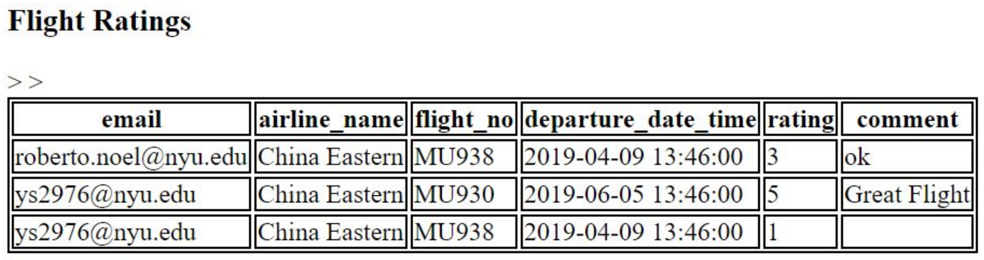
Avalon

With the following query:"INSERT INTO `airport` (`name`, `city`) VALUES (%s, %s)"

6. View flight ratings:

In /ViewFlightRatings page, a staff is able to see all the ratings and comments of the flights of the airline he or she works for, along with the average rage rating per flight.

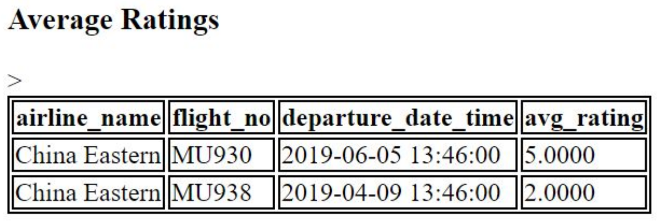
Roberto Noel
Neal Shu
Databases
Ratan Dey



Flight Ratings

> >

email	airline_name	flight_no	departure_date_time	rating	comment
roberto.noel@nyu.edu	China Eastern	MU938	2019-04-09 13:46:00	3	ok
ys2976@nyu.edu	China Eastern	MU930	2019-06-05 13:46:00	5	Great Flight
ys2976@nyu.edu	China Eastern	MU938	2019-04-09 13:46:00	1	



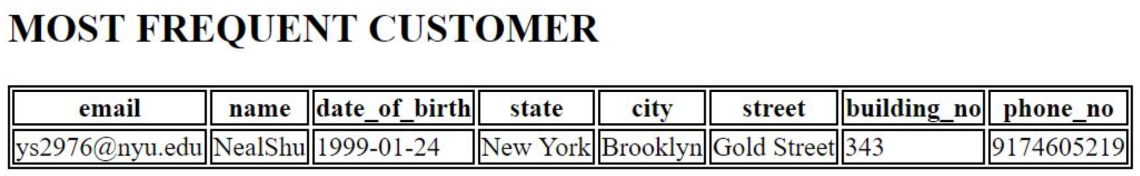
Average Ratings

>

airline_name	flight_no	departure_date_time	avg_rating
China Eastern	MU930	2019-06-05 13:46:00	5.0000
China Eastern	MU938	2019-04-09 13:46:00	2.0000

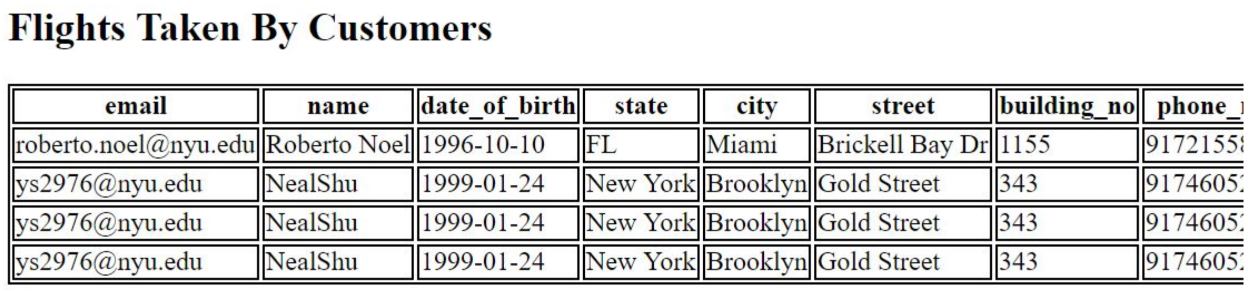
With the following query:”SELECT email, airline_name, flight_no, departure_date_time, comment, rating FROM `rates` WHERE airline_name = %s”
And
SELECT airline_name, flight_no, departure_date_time, AVG(rating) AS 'rating' FROM `rates` WHERE airline_name = %s group by airline_name, flight_no, departure_date_time

7. View frequent customers:
A staff is able to see the most frequent customer of the last year of the airline he or she works for in the /ViewFrequentCustomer page, plus all the customers from the airline



MOST FREQUENT CUSTOMER

email	name	date_of_birth	state	city	street	building_no	phone_no
ys2976@nyu.edu	NealShu	1999-01-24	New York	Brooklyn	Gold Street	343	9174605219



Flights Taken By Customers

email	name	date_of_birth	state	city	street	building_no	phone_
roberto.noel@nyu.edu	Roberto Noel	1996-10-10	FL	Miami	Brickell Bay Dr	1155	9172155
ys2976@nyu.edu	NealShu	1999-01-24	New York	Brooklyn	Gold Street	343	917460
ys2976@nyu.edu	NealShu	1999-01-24	New York	Brooklyn	Gold Street	343	917460
ys2976@nyu.edu	NealShu	1999-01-24	New York	Brooklyn	Gold Street	343	917460

With the following query:”SELECT email, name, date_of_birth, state, city, street, building_no, phone_no
FROM customer NATURAL JOIN ticket NATURAL JOIN flight

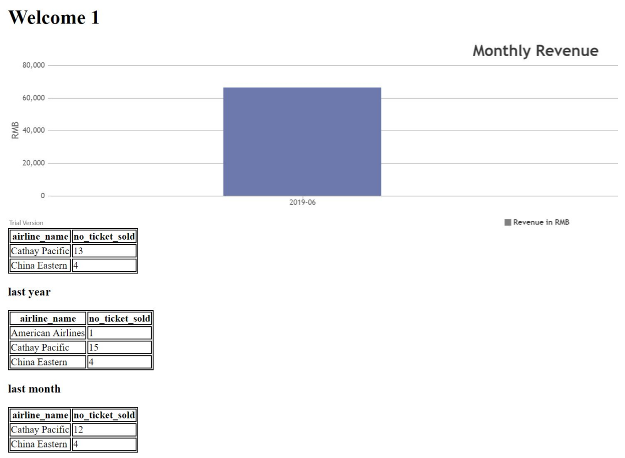
WHERE airline_name = %s AND email = (SELECT MAX(email) FROM customer) AND purchase_date_time BETWEEN DATE_SUB(NOW(), INTERVAL 365 DAY) and NOW() group by email”

And

SELECT email, name, date_of_birth, state, city, street, building_no, phone_no, ticketID, flight_no, departure_date_time, departs_from, arrives_from, purchase_date_time, base_price, sold_price, flight_status
FROM customer NATURAL JOIN ticket NATURAL JOIN flight
WHERE airline_name = %s
ORDER BY email

8. View reports:

By inputting the range of dates, a staff is able to see the number of tickets sold for all airlines, a bar chart showing the number of tickets sold month wise. By default a staff will also see the total number of tickets sold by each airline for last year and last month



Welcome 1

airline_name	no_ticket_sold
Cathay Pacific	13
China Eastern	4

last year

airline_name	no_ticket_sold
American Airlines	1
Cathay Pacific	15
China Eastern	4

last month

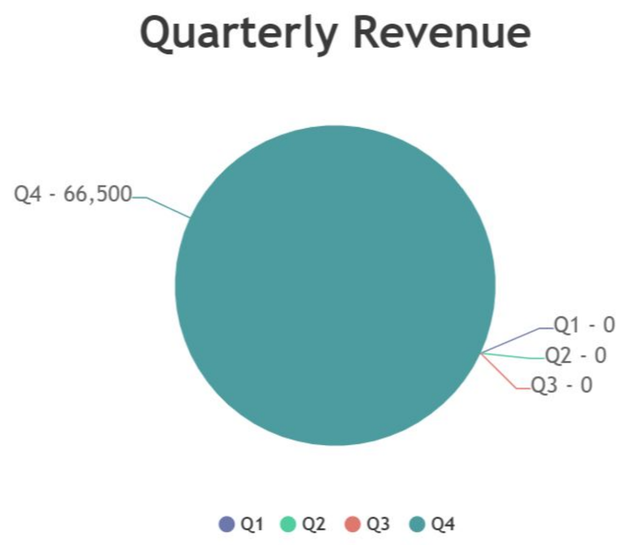
airline_name	no_ticket_sold
Cathay Pacific	12
China Eastern	4

With the following queries:
Query1: SELECT airline_name, COUNT(ticketID) as No_ticket_sold FROM ticket WHERE purchase_date_time > %s AND purchase_date_time < %s GROUP BY airline_name
Query2: SELECT airline_name, COUNT(ticketID) as No_ticket_sold FROM ticket WHERE purchase_date_time BETWEEN DATE_SUB(NOW(), INTERVAL 365 DAY) AND NOW() GROUP BY airline_name
Query3: SELECT airline_name, COUNT(ticketID) as No_ticket_sold FROM ticket WHERE purchase_date_time BETWEEN DATE_SUB(NOW(), INTERVAL 30 DAY) AND NOW() GROUP BY airline_name

Roberto Noel
Neal Shu
Databases
Ratan Dey
Query4: SELECT month, revenue FROM monthly_revenue WHERE airline_name = %s ORDER BY month
Note: query4 is used for the bar chart from the view monthly_revenue where the airline's revenue is grouped by month

9. View quarterly revenue earned:

A staff is able to see the quarterly revenue of the airline for the last 365 days, a pie chart is shown in the /ViewQuarterlyRevenueEarned page

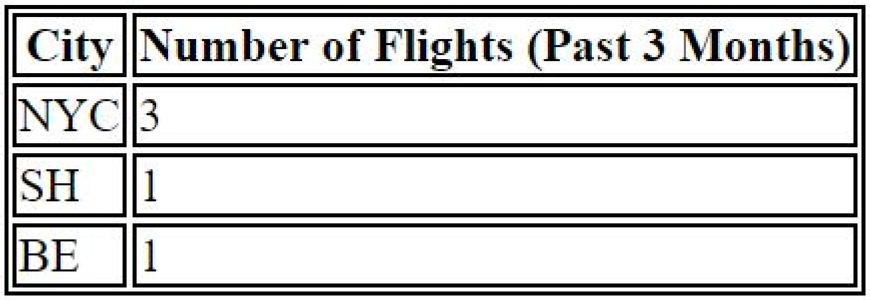


With the following query:"SELECT COALESCE(SUM(sold_price), 0) as 'Revenue'
FROM `ticket`
WHERE airline_name = %s AND purchase_date_time BETWEEN DATE_SUB(NOW(), INTERVAL 365 DAY) AND (DATE_SUB(NOW(), INTERVAL 275 DAY))
" which is the query for Q1, and changing the range of date will give data for Q23

10.View Top destinations:

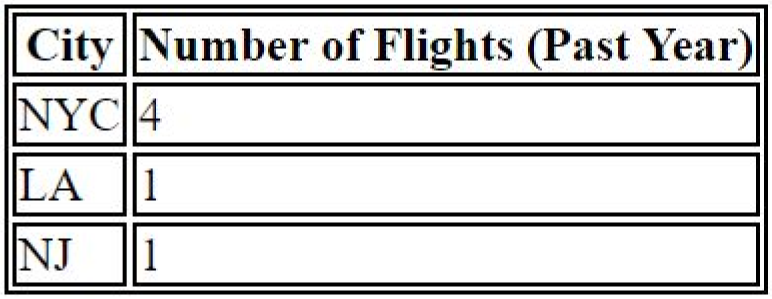
Top 3 destinations for the last three months and last year is displayed in this page, and the result is for all airlines.

Roberto Noel
Neal Shu
Databases
Ratan Dey



City	Number of Flights (Past 3 Months)
NYC	3
SH	1
BE	1

>



City	Number of Flights (Past Year)
NYC	4
LA	1
NJ	1

With the following query:”SELECT city, count

```
            FROM `topdest_last_three_months`
            order by count desc
            limit 3”
```

And

```
                            SELECT city, count
            FROM `topdest_last_year`
            order by count desc
            limit 3
```

Where topdest_last_three_months and topdest_last_year are two views we created that display the city names and the number of flights that have arriving destination as them.

11.Logout:

A link in the staff home page for a staff to log out, session will be popped and user will be redirected to index.html