

This file may not contain all the details of codes for error checking, but all the important queries are included.

1. General Home Page (Public) app.py + layout.html

Use cases are highlighted in the screenshot and will be explained below.

The screenshot shows the TripNow General Home Page. At the top, there are navigation links: TripNow, Flight Search, Flight Status, Login, Register, and About. The Flight Search and Flight Status links are highlighted with yellow circles. Below the navigation, there's a "Home Page" section with a "Weather" icon and a "Change City" input field. To the right, there's a "Delay Warning" box titled "Delayed Flights" with an exclamation mark icon. It lists flight information for Delta Air Lines flights delayed at PEK, ATL, LAX, and DTW airports.

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

1-1. Flight Search (search())

The screenshot shows the TripNow Flight Search page. At the top, there are navigation links: TripNow, Flight Search, Flight Status, Login, Register, and About. The Flight Search link is highlighted with a yellow circle. Below the navigation, there's a "Flight Search" section with fields for "From" (PVG/Shanghai), "To" (JFK/New York), "Date" (mm/dd/yyyy), and a "Search" button. The search results table shows flight information for Delta Air Lines flights from LAX to PEK, ATL, and LAX, and from DTW to JFK. To the right, there's a "Delay Warning" box titled "Delayed Flights" with an exclamation mark icon, showing the same delayed flight data as the home page.

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

[Go back](#)

```

else:
    cursor = conn.cursor()
    query = "SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, price, status \
    FROM airport AS a, flight AS b, airport AS c \
    WHERE a.airport_name = b.departure_airport\
    AND b.arrival_airport = c.airport_name\
    AND a.airport_city= '{}'\ \
    AND c.airport_city = '{}'\
    AND departure_time LIKE '{}'""
    cursor.execute(query.format(departure, arrival, '%'+str(date)+'%'))
    flight = cursor.fetchall()

```

1-2. Flight Status (status())

① 127.0.0.1:5000/status

TripNow Flight Search Flight Status Login Register About

Flight Status

Please Enter the Flight Info Below

Airline	China Southern	Flight No.	6996
Date	mm/dd/yyyy	<input type="button" value="Search"/>	

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

[Go back](#)

! Delay Warning

Delayed Flights

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

Main query under status():

```
.64     cursor = conn.cursor()
.65     query = "SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, status \
.66         FROM flight WHERE airline_name = \'{}\' AND flight_num = \'{}\' AND departure_time LIKE \'{}\'"
.67
.68     cursor.execute(query.format(airline, flight_num, '%'+str(date)+'%'))
.69     flight = cursor.fetchall()
```

1-3. Login → return prelogin()

① 127.0.0.1:5000/prelogin

TripNow Flight Search Flight Status Login Register About

Please Log In

Choose Your Login Type

Customer Login
Booking Agent Login
Airline Staff Login

! Delay Warning

Delayed Flights

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

Choose login type (customerlogin(), booklogin(), stafflogin())
Take customerlogin() for example:

① 127.0.0.1:5000/customerlogin?

TripNow Flight Search Flight Status Login Register About

Customer Login

Please enter your email and password

Email Address
<input type="text" value="Email"/>
Password
<input type="password" value="Password"/>

[Login](#)

Need An Account? [Register Now](#)

! Delay Warning

Delayed Flights

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

Main Query under customerlogin():

```

if request.method == 'POST':

    # Create variables for easy access
    email = request.form['email']
    password = request.form['password']

    cursor.execute('SELECT * FROM customer WHERE email = %s AND password = %s', (email, password))
    # Fetch one record and return result
    data = cursor.fetchone()

    # If account exists in accounts table in out database
    if data:
        # Create session data, we can access this data in other routes
        session['name'] = ''.join(name).strip(',')
        name = data[1]
        session['name'] = name.strip('')
        session['cus'] = 'yes'
        session['username'] = email
        # Redirect to home page
        return redirect(url_for('cushome'))
    else:
        # Account doesn't exist or username/password incorrect
        msg = 'The email address or password you entered is incorrect!'

return render_template('logincus.html', msg=msg, data1=data1)

```

Stafflogin() and booklogin() are similar. It will create unique session['cus'] for customer, session['id'] for booking agent, and session['airline name'] for staff to prevent them from accessing other account types.

1-4. Registering → Return preregister()

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

Choose registering type (customerreg(), bookreg(), staffreg())
Take staffreg() for example:

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Status
Delta Air Lines	22	PVG	PEK	2021-05-20 13:32:00	DELAYED
Delta Air Lines	186	ATL	PVG	2022-02-03 23:32:00	DELAYED
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	DELAYED
Delta Air Lines	1797	DTW	JFK	2020-05-11 18:06:53	DELAYED

Main query under staffreg():

```
1762     else:
1763         # Account doesn't exists and the form data is valid, now insert new account into accounts table
1764         ins = "INSERT INTO airline_staff VALUES( '{}', '{}', '{}', '{}', '{}', '{}')"
1765         cursor.execute(ins.format( email, password, first_name, last_name, date_of_birth, airline_name))
1766         conn.commit()
1767         msg = 'You have successfully registered! Log in to view your homepage!'
1768
1769     return render_template('registerstaff.html', msg=msg, data1=data1)
```

customerreg() and bookreg() are similar. We will make sure there's no identical email address beforehand:

```
#Check if account exists using MySQL
query = "SELECT * FROM booking_agent WHERE email = '{}'"
cursor.execute(query.format(email))
#cursor.execute('SELECT * FROM accounts WHERE username = %s', (username))
account = cursor.fetchone()
# If account exists show error and validation checks
if account:
    msg = 'Email already exists!'
```

1-5. Weather under home():

Manually convert the icons.

```
# source contain json data from api
try:
    source = urllib.request.urlopen('http://api.openweathermap.org/data/2.5/weather?q=' + city + '&appid=' + api).read()

    list_of_data = json.loads(source)
    icon = []
    "01d": "fas fa-sun",
    "01n": "fas fa-sun",
    "02n": "fas fa-cloud-moon",
    "02d": "fas fa-cloud-sun",
    "03d": "fas fa-cloud-meatball",
    "03n": "fas fa-cloud-meatball",
    "04d": "fas fa-cloud-meatball",
    "04n": "fas fa-cloud-meatball",
    "09d": "fas fa-cloud-showers-heavy",
    "09n": "fas fa-cloud-showers-heavy",
    "10d": "fas fa-cloud-sun-rain",
    "10n": "fas fa-cloud-sun-rain",
    "11d": "fas fa-poo-storm",
    "11n": "fas fa-poo-storm",
    "13d": "fas fa-snowflake",
    "13n": "fas fa-snowflake",
    "50d": "fas fa-smog",
    "50n": "fas fa-smog"
]

    data = {
        "cityname": str(list_of_data['name'].capitalize()),
        "country_code": str(list_of_data['sys']['country']),
        "coordinated": str(list_of_data['coord']['lon']) + ' ' +
        str(list_of_data['coord']['lat']),
        "weather": str(list_of_data['weather'][0]['description'].capitalize()),
        "temp": str(round(list_of_data['main']['temp']-273, 1)) + "C",
        "wind": str(list_of_data['wind']['speed']),
        "pressure": str(list_of_data['main']['pressure']),
        "humidity": str(list_of_data['main']['humidity'])+'%',
        "icon": str(icon[list_of_data['weather'][0]['icon']])
    }
```

1-6. Delay Warning (embedded in layout.html):

```
cursor = conn.cursor()
cursor.execute('SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time FROM flight WHERE status = "DELAYED"')
data1 = cursor.fetchall()
msg=''
data = {}
```

2. Customer Use Cases (app.py + layoutcus.html)

For every use case, check:

```
if 'username' in session and 'cus' in session:
```

C ① 127.0.0.1:5000/customer

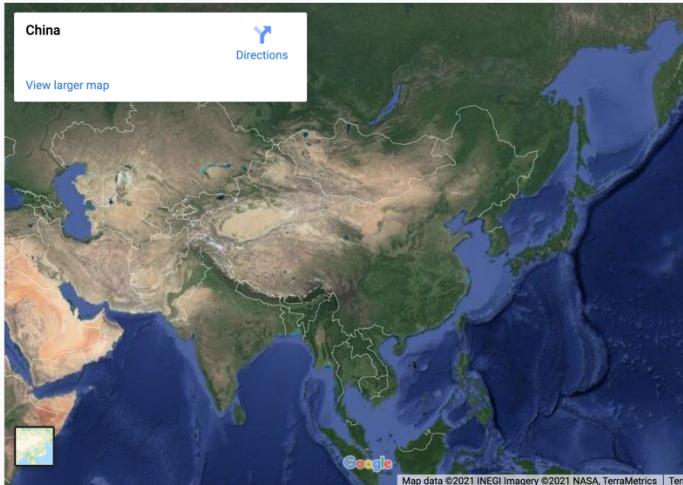
TripNow Book My Trips Flight Status My Account

My Profile Logout

Home Page

Welcome back, Mengshuo Ye!

Choose your next destination!



Current local time in

Shanghai, China

Fri, 14. May 2021

08:52:53 p.m.

2-1. Book (Purchase tickets + Search for flights)

C ① 127.0.0.1:5000/customer/book

TripNow Book My Trips Flight Status My Account

My Profile Logout

Book

Please enter your source city and destination

From PVG/Shanghai To JFK/New York

Date mm/dd/yyyy

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Arr. Time	Price	Remaining Seats
Delta Air Lines	88	LAX	PVG	2021-08-08 16:46:11	2021-08-11 16:46:11	1234	119
Delta Air Lines	185	LAX	PVG	2021-08-08 11:05:44	2021-08-09 11:05:44	5222	0
Delta Air Lines	188	LAX	PVG	2021-08-08 16:58:16	2021-08-08 16:58:16	4312	153

Current local time in
Shanghai, China

Fri, 14. May 2021

08:57:01 p.m.

Please enter the information below to confirm your choice

Airline United Airlines Flight No. 88

① 127.0.0.1:5000/customer/bookconfirm

TripNow Book My Trips Flight Status My Account

My Profile Logout

Booking Successful!

You have successfully booked the flight! Delta Air Lines 188 on 2021-08-08

Ticket ID: 665771227

[View My Trips](#)

Current local time in
Shanghai, China

Fri, 14. May 2021

08:57:24 p.m.

2-1-1. Search and confirm (under bookcus() and bookcusconfirm())

```
if departure.isupper():
    cursor = conn.cursor()
    query = "SELECT ALL airline_name, flight_num, departure_airport,
arrival_airport, departure_time, arrival_time, price, seats-count(ticket_id) AS seat_left \
              FROM flight LEFT JOIN ticket USING (airline_name, flight_num) JOIN
airplane USING (airline_name, airplane_id) \
              WHERE departure_airport = \'{}\\' \
              AND arrival_airport = \'{}\\' \
              AND departure_time LIKE \'{}\\' \
              GROUP BY airline_name, flight_num"

    cursor.execute(query.format(departure, arrival, '%' + str(flight_date) + '%'))
    flight = cursor.fetchall()
```

This one above is for searching with airport code.

```
query = "SELECT ALL airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, price,
seats-count(ticket_id) AS seat_left \
              FROM airport AS a, (flight LEFT JOIN ticket USING (airline_name, flight_num) JOIN airplane
USING (airline_name, airplane_id)), airport AS c \
              WHERE a.airport_name = departure_airport \
              AND arrival_airport = c.airport_name \
              AND a.airport_city= \'{}\\' \
              AND c.airport_city = \'{}\\' \
              AND departure_time LIKE \'{}\\' \
              GROUP BY airline_name, flight_num"
```

This one above is for searching with city name.

2-1-2. Checking available seat number and complete booking Under bookcusconfirm():

```
query = "SELECT seats-count(ticket_id) AS seat_left \
              FROM flight JOIN ticket USING (airline_name, flight_num) JOIN airplane USING (airline_name, airplane_id) \
              WHERE airline_name = \'{}\\' \
              AND flight_num = \'{}\\'"
cursor.execute(query.format(airline, flight_num))
#stores the results in a variable
data = cursor.fetchone()
if data[0] == None:
    msg = "The flight number does not exist!"
    return render_template('bookcus.html', msg=msg, flight=flight, date=flight_date)
elif data[0] == 0:
    #If the previous query returns data, then user exists
    msg = "This flight has been sold out!"
    return render_template('bookcus.html', msg=msg, flight=flight, date=flight_date)
else:
    a = 0
    while a == 0:
        tkt_id = randint(1000000000, 999999999)
        query = "SELECT * FROM ticket WHERE ticket_id = \'{}\\'"
        cursor.execute(query.format(tkt_id))
        data = cursor.fetchone()
        if not data:
            a = 1
try:
    ins1 = "INSERT INTO ticket VALUES(\'{}\', \'{}\', \'{}\')"
    cursor.execute(ins1.format(tkt_id, airline, flight_num))
    today = datetime.today().strftime('%Y-%m-%d')
    ins2 = "INSERT INTO purchases (ticket_id, customer_email, purchase_date) VALUES(\'{}\', \'{}\', \'{}\')"
    cursor.execute(ins2.format(tkt_id, email, today))
    conn.commit()
    cursor.close()
    msg = "You have successfully booked the flight! " + airline + " " + flight_num + " on " + flight_date
    msg1 = "Ticket ID: " + str(tkt_id)
    #flash("Booking Successful! Flight:" + airline + flight_num)
    return render_template('bookcusconfirm.html', msg=msg, msg1=msg1)
```

2.2. My trips (View my flights)

The screenshot shows a web browser window for the TripNow application at the URL 127.0.0.1:5000/customer/trips. The page title is "View my flights". A search form is present with fields for "From" (PVG/Shanghai) and "To" (JFK/New York), and date inputs for "Departure Date From" and "To". A "Search" button is also visible. To the right, a box displays the current local time in Shanghai, China, as "Fri, 14. May 2021 09:10:31 p.m.". Below the search form, a table titled "Upcoming flights:" lists five flight records:

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Arr. Time	Price	Status	Ticket ID
Delta Air Lines	88	LAX	PVG	2021-08-08 16:46:11	2021-08-11 16:46:11	1234	UPCOMING	123
Delta Air Lines	88	LAX	PVG	2021-08-08 16:46:11	2021-08-11 16:46:11	1234	UPCOMING	537218102
Delta Air Lines	88	LAX	PVG	2021-08-08 16:46:11	2021-08-11 16:46:11	1234	UPCOMING	762927132
Delta Air Lines	88	LAX	PVG	2021-08-08 16:46:11	2021-08-11 16:46:11	1234	UPCOMING	861443858
Delta Air Lines	185	LAX	PVG	2021-08-08 16:46:11	2021-08-09 16:46:11	5222	UPCOMING	41513

Queries under tripcus():

Basic default view

```
q = 'SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, price, status, ticket_id \
      FROM flight NATURAL JOIN ticket NATURAL JOIN purchases \
      WHERE departure_time > \'{}\' AND customer_email = \'{}\''
cursor.execute(q.format(start_date, customer_email))
t = cursor.fetchall()
```

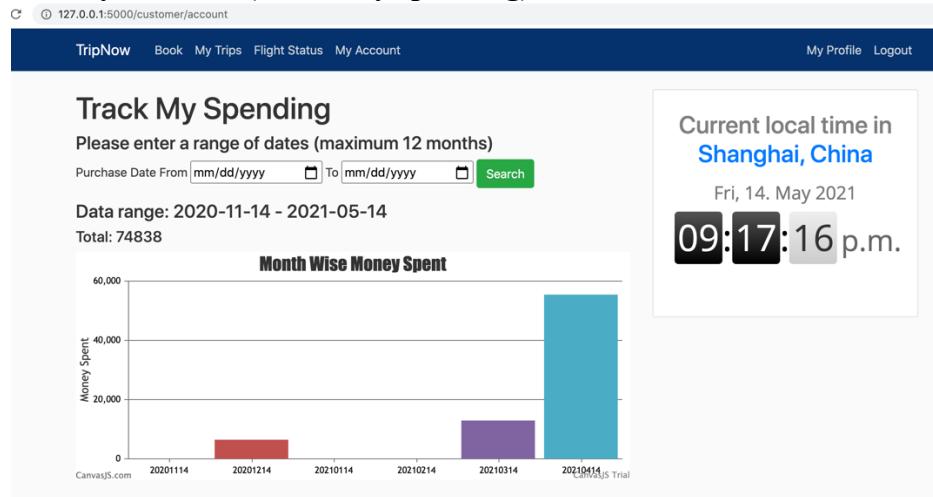
Advanced search:

```
if request.method == 'POST':
    start_date = request.form['start date']
    end_date = request.form['end date']
    departure = request.form['departure']
    arrival = request.form['arrival']

    if departure.isupper():
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time,
f.price, f.status, t.ticket_id \
              FROM flight f JOIN ticket t ON(f.airline_name = t.airline_name AND f.flight_num = t.flight_num) \
              JOIN purchases p ON(t.ticket_id = p.ticket_id) JOIN airport a ON(f.departure_airport = a.airport_name) \
              JOIN airport b ON(f.arrival_airport = b.airport_name) \
              WHERE p.customer_email = \'{}\' AND p.purchase_date <= \'{}\' AND p.purchase_date >= \'{}\' \
              AND f.departure_airport = \'{}\' AND f.arrival_airport = \'{}\' '
        cursor.execute(q.format(customer_email, end_date, start_date, departure, arrival))
        t = cursor.fetchall()
    else:
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time,
f.price, f.status, t.ticket_id \
              FROM flight f JOIN ticket t ON(f.airline_name = t.airline_name AND f.flight_num = t.flight_num) \
              JOIN purchases p ON(t.ticket_id = p.ticket_id) JOIN airport a ON(f.departure_airport = a.airport_name) \
              JOIN airport b ON(f.arrival_airport = b.airport_name) \
              WHERE p.customer_email = \'{}\' AND p.purchase_date <= \'{}\' AND p.purchase_date >= \'{}\' \
              AND a.airport_city = \'{}\' AND b.airport_city = \'{}\' '
        cursor.execute(q.format(customer_email, end_date, start_date, departure, arrival))
        t = cursor.fetchall()
```

2-3. Flight Status: Similar to the public flight status. Codes under statuscus().

2-4. My Account (Track my spending): Based on CanvasJS



Codes under accountcus():

Default 6 months:

```
curdate = start_date
curdate1 = start_date + relativedelta(months=+1)
i = 0
cursor = conn.cursor()
while curdate < end_date:

    q = "SELECT SUM(price) FROM purchases NATURAL JOIN ticket NATURAL JOIN flight\
        WHERE customer_email = '{}'\n        AND purchase_date > '{}' AND purchase_date <='{}'"
    cursor.execute(q.format(email, curdate, curdate1))
    result.append(cursor.fetchone()[0])
    i += 1
    curdate = curdate + relativedelta(months=+1)
    curdate1 = curdate + relativedelta(months=+1)
```

Advanced search:

```
else:
    i = 0
    curdate = datetime.strptime(start_date, '%Y-%m-%d')
    #curdate = start_date
    curdate1 = curdate + relativedelta(months=+1)

    cursor = conn.cursor()
    while curdate <= date1:
        q = "SELECT SUM(price) FROM purchases NATURAL JOIN ticket NATURAL JOIN flight\
            WHERE customer_email = '{}'\n            AND purchase_date > '{}' AND purchase_date <='{}'"
        cursor.execute(q.format(email, str(curdate)[0:10], str(curdate1)[0:10]))
        result_temp.append(cursor.fetchone()[0])

        i += 1
        curdate = curdate + relativedelta(months=+1)
        curdate1 = curdate + relativedelta(months=+1)

    if i == 13:
        msg = 'Please keep the date range within 12 months!'
        return render_template('accountcus.html', msg=msg, tot=tot, ran=ran, num=num)
```

2-5. My Profile

TripNow Book My Trips Flight Status My Account My Profile Logout

Profile

Account Details

Email	scott.ye0909@gmail.com
Name	Mengshuo Ye
Password	12345
Building No.	1
Street	99-114-302 East Guoquan rd
City	shanghai
State	shanghai
Phone No.	1730187657
Passport No.	wdawdawd
Passport Expiration	2022-12-31
Passport Country	China
Date of Birth	2000-09-09

Current local time in
Shanghai, China

Fri, 14. May 2021

09:21:25 p.m.

Codes under profilecus():

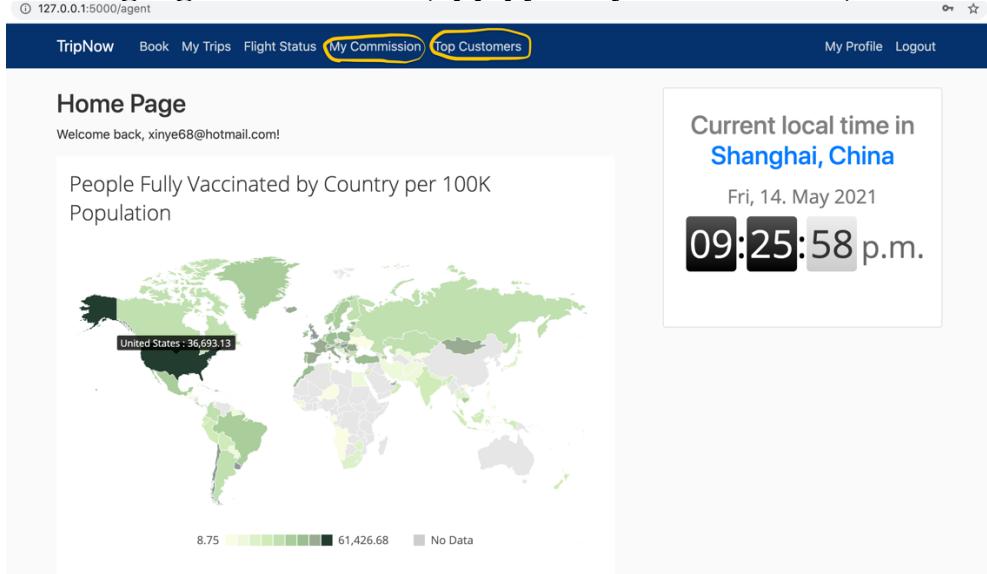
```
506     @app.route('/customer/profile')
507     def profilecus():
508         if 'username' in session and 'cus' in session:
509
510             cursor = conn.cursor()
511             username = session['username']
512             # We need all the account info for the user so we can display it on the profile page
513             q = 'SELECT * FROM customer WHERE email = \'{}\''
514             cursor.execute(q.format(username))
515
516             account = cursor.fetchone()
517             # Show the profile page with account info
518             return render_template('profilecus.html', account=account)
519         # User is notloggedin redirect to login page
520         return redirect(url_for('prelogin'))
```

2-6. Logout

Pop everything possible from session

```
572     @app.route('/logout')
573     def logout():
574         session.pop('username', None)
575         session.pop('name', None)
576         session.pop('airline name', None)
577         session.pop('id', None)
578         session.pop('cus', None)
579
580         # Redirect to login page
581         return redirect(url_for('prelogin'))
```

3. Booking Agent Use Cases (app.py + layoutbook.html)



3-1. My Trips (View My Flights)

Similar to the one in customer. Codes under tripsagent().

```
q = 'SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, price, status, ticket_id \
      FROM flight NATURAL JOIN ticket NATURAL JOIN purchases \
      WHERE departure_time >= \'{}\' AND booking_agent_id = \'{}\''
cursor.execute(q.format(start_date, booking_agent_id))
t = cursor.fetchall()

if request.method == 'POST':
    start_date = request.form['start date']
    end_date = request.form['end date']
    departure = request.form['departure']
    arrival = request.form['arrival']
    if departure.isupper():
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time, f.price, f.status, t.ticket_id \
              FROM flight f JOIN ticket t ON(f.airline_name = t.airline_name AND f.flight_num = t.flight_num) \
              JOIN purchases p ON(t.ticket_id = p.ticket_id) JOIN airport a ON(f.departure_airport = a.airport_name) \
              JOIN airport b ON(f.arrival_airport = b.airport_name) \
              WHERE p.booking_agent_id = \'{}\' AND p.purchase_date <= \'{}\' AND p.purchase_date >= \'{}\' \
              AND f.departure_airport = \'{}\' AND f.arrival_airport = \'{}\' '
        cursor.execute(q.format(booking_agent_id, end_date, start_date, departure, arrival))
        t = cursor.fetchall()
    else:
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time, f.price, f.status , t.ticket_id \
              FROM flight f JOIN ticket t ON(f.airline_name = t.airline_name AND f.flight_num = t.flight_num) \
              JOIN purchases p ON(t.ticket_id = p.ticket_id) JOIN airport a ON(f.departure_airport = a.airport_name) \
              JOIN airport b ON(f.arrival_airport = b.airport_name) \
              WHERE p.booking_agent_id = \'{}\' AND p.purchase_date <= \'{}\' AND p.purchase_date >= \'{}\' \
              AND a.airport_city = \'{}\' AND b.airport_city = \'{}\' '
        cursor.execute(q.format(booking_agent_id, end_date, start_date, departure, arrival))
        t = cursor.fetchall()
```

3-2. Book (Search for flights & Purchase tickets)

Similar to customer book. Should also enter the email address for customer. Codes under bookagent() and bookagentconfirm().

```
else:
    a = 0
    while a == 0:
        tkt_id = randint(100000000, 999999999)
        query = "SELECT * FROM ticket WHERE ticket_id = \'{\}\'"
        cursor.execute(query.format(tkt_id))
        data = cursor.fetchone()
        if not data:
            a = 1
try:
    ins1 = "INSERT INTO ticket VALUES(\'{\}', \'{\}', \'{\}')"
    cursor.execute(ins1.format(tkt_id, airline, flight_num))

    today = datetime.today().strftime('%Y-%m-%d')

    ins2 = "INSERT INTO purchases VALUES(\'{\}', \'{\}', \'{\}', \'{\}')"
    cursor.execute(ins2.format(tkt_id, cus_email, book_id, today))
    conn.commit()
    cursor.close()

    msg = "You have successfully booked the flight! " + airline + " " + flight_num
    msg1 = "Ticket ID: " + str(tkt_id)
    #flash("Booking Successful! Flight:" + airline + flight_num)
    return render_template('bookagentconfirm.html', msg=msg, msg1=msg1)
except NameError:
    redirect(url_for('bookagent'))
```

3-3. Flight Status

Similar to public flight status. Codes under statusagent().

3-4. My Commission

The screenshot shows a web application interface for managing flight bookings and commissions. At the top, there's a navigation bar with links for TripNow, Book, My Trips, Flight Status, My Commission, Top Customers, My Profile, and Logout. The main content area has a title 'My Commission'. It prompts the user to 'Choose the start and end date:' with two input fields for 'Start Date' and 'End Date', both set to 'mm/dd/yyyy'. Below these fields is a blue button labeled 'See Total Commission'. To the right, a box displays the 'Current local time in Shanghai, China' as 'Fri, 14. May 2021 09:33:17 p.m.'. Further down, it shows the 'Total amount of commission received between 2021-04-14 and 2021-05-14 is: 1724.80', the 'Average commission received between 2021-04-14 and 2021-05-14 is: 107.80', and the 'Total number of tickets sold between 2021-04-14 and 2021-05-14 is: 4'.

Codes under commissionagent():

Default view and advanced search:

```
q = 'SELECT CAST(0.1*SUM(price) as decimal(10,2)), CAST((0.1*price)/COUNT(*) as decimal(10,2)), COUNT(*) \
      FROM purchases NATURAL JOIN ticket NATURAL JOIN flight \
      WHERE booking_agent_id = \'{\}' AND purchase_date <= \'{\}' AND purchase_date >= \'{\}\''
cursor.execute(q.format(booking_agent_id, end_date, start_date))
commission = cursor.fetchone()

if request.method == 'POST':
    start_date = request.form['start date']
    end_date = request.form['end date']

    q = 'SELECT CAST(0.1*SUM(price) as decimal(10,2)), CAST((0.1*price)/COUNT(*) as decimal(10,2)), COUNT(*) \
          FROM purchases NATURAL JOIN ticket NATURAL JOIN flight \
          WHERE booking_agent_id = \'{\}' AND purchase_date <= \'{\}' AND purchase_date >= \'{\}\''
    cursor.execute(q.format(booking_agent_id, end_date, start_date))
    commission = cursor.fetchone()
```

3-5. Top Customers

① 127.0.0.1:5000/agent/topcus

TripNow Book My Trips Flight Status My Commission Top Customers My Profile Logout

View Top Customers

Top 5 customers based on number of tickets bought from the booking agent in the past 6 months:

Mengshuo Ye
S
sda
Tom

Top 5 customers based on the amount of commission received in the last year:

Paul Ye
AB
Mengshuo Ye

Current local time in Shanghai, China
Fri, 14. May 2021
09:35:45 p.m.

Top 5 Customers Based on #Ticket in Past 6 Months

Customer	#Ticket Bought
Mengshuo Ye	3
S	1
sda	1
Tom	1

Top 5 Customers Based on Commission Last Year

Customer	Amount of Commission
Paul Ye	~950
AB	~100
Mengshuo Ye	~100

Queries under topcus():

```
q1 = "SELECT name, count(ticket_id) FROM purchases JOIN customer ON (customer.email = purchases.customer_email)\\
      WHERE booking_agent_id = \'{\}\}'\\
      AND purchase_date <= \'{\}\}' AND purchase_date >= \'{\}\}'\\
      GROUP BY customer_email\\
      ORDER BY COUNT(ticket_id) DESC LIMIT 5"
```

```
q2 = "SELECT name, CAST(0.1*SUM(price) as decimal(10,2))\\
      FROM (purchases NATURAL JOIN ticket NATURAL JOIN flight) JOIN customer ON (customer.email =\\
      customer_email)\\
      WHERE booking_agent_id = \'{\}\}' AND purchase_date <= \'{\}\}' AND purchase_date >= \'{\}\}'\\
      GROUP BY customer_email\\
      ORDER BY COUNT(ticket_id) DESC LIMIT 5"
```

3-6. Logout: logout()

3-7. My Profile: Similar to customer profile. Queries under profileagent().

4. Airline Staff Use Cases (app.py + layoutstaff.html)

Welcome back, Scott!

This is the place where you can manage the flights and view the reports!

See your upcoming flights.

[View My Trips](#)

4-1. My Trips (View My Flights)

Airline	Flight No.	Dep. Airport	Arr. Airport	Dep. Time	Arr. Time	Price Status
Delta Air Lines 22	PVG	PEK	2021-05-20 13:32:00	2021-05-20 15:39:00	132	DELAYED
Delta Air Lines 154	SEA	DTW	2021-05-30 12:21:00	2021-05-30 14:30:00	275	UPCOMING

[Go back](#)

Similar to customer and booking agent. Default 30 days view. Queries under tripsstaff().

```
if 'username' in session and 'airline name' in session:
    cursor = conn.cursor()
    airline_name = session['airline name']
    #default time is showing all the upcoming flights
    start_date = datetime.today()
    end_date = start_date + timedelta(days=30)
    msg = ''
    condition = ''

    q = 'SELECT airline_name, flight_num, departure_airport, arrival_airport, departure_time, arrival_time, price, status \
        FROM flight WHERE departure_time >= \'{}\' AND departure_time < \'{}\' AND airline_name = \'{}\''
    cursor.execute(q.format(start_date, end_date, airline_name))
    t = cursor.fetchall()
```

Advanced Search:

```
request.method == 'POST':
    start_date = request.form['start date']
    end_date = request.form['end date']
    departure = request.form['departure']
    arrival = request.form['arrival']

    if departure.isupper():
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time, f.price, f.status \
            FROM flight f \
            WHERE f.departure_time >= \'{}\' AND f.departure_time < \'{}\' \
            AND f.departure_airport = \'{}\' AND f.arrival_airport = \'{}\' AND f.airline_name = \'{}\' '
        cursor.execute(q.format(start_date, end_date, departure, arrival, airline_name))
        t = cursor.fetchall()
    else:
        cursor = conn.cursor()
        q = 'SELECT f.airline_name, f.flight_num, f.departure_airport, f.arrival_airport, f.departure_time, f.arrival_time, f.price, f.status \
            FROM flight f JOIN airport a ON(f.departure_airport = a.airport_name) \
            JOIN airport b ON(f.arrival_airport = b.airport_name) \
            WHERE f.departure_time >= \'{}\' AND f.departure_time < \'{}\' \
            AND a.airport_city = \'{}\' AND b.airport_city = \'{}\' AND f.airline_name = \'{}\' '
        cursor.execute(q.format(start_date, end_date, departure, arrival, airline_name))
        t = cursor.fetchall()
```

4-2. Booking Agents (View All the booking agents)

① 127.0.0.1:5000/staff/viewagents

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations

My Profile Logout

Booking Agents

Top 5 Sales in the Past Month:

xinye68@hotmail.com

Top 5 Sales in the Past Year:

xinye68@hotmail.com

Top 5 Commission in the Last Year:

xinye68@hotmail.com

List of All Agents:

1@2.com

ilbamfha@outlook.com

pfle@126.com

xinye68@hotmail.com

Queries under viewagents():

```
889 @app.route('/staff/viewagents')
890 def viewagents():
891     if 'username' in session and 'airline name' in session:
892         end_date = date.today()
893         start_date_month = end_date-timedelta(days=30)
894         start_date_year = end_date-timedelta(days=365)
895         end_date_last = start_date_year.strftime("%Y")+'-12-31'
896         start_date_last = start_date_year.strftime("%Y")+'-01-01'
897
898         cursor = conn.cursor()
899         q = 'SELECT email FROM purchases NATURAL JOIN booking_agent \
900             WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \
901             GROUP BY booking_agent_id ORDER BY COUNT(*) DESC LIMIT 5'
902         cursor.execute(q.format(end_date, start_date_month))
903         s = cursor.fetchall()
904
905
906         q = 'SELECT email FROM purchases NATURAL JOIN booking_agent \
907             WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \
908             GROUP BY booking_agent_id ORDER BY COUNT(*) DESC LIMIT 5'
909         cursor.execute(q.format(end_date, start_date_year))
910         y = cursor.fetchall()
911
912         q = 'SELECT email FROM purchases NATURAL JOIN ticket NATURAL JOIN flight NATURAL JOIN booking_agent \
913             WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \
914             GROUP BY booking_agent_id ORDER BY SUM(price) DESC LIMIT 5'
915         cursor.execute(q.format(end_date_last, start_date_last))
916         c = cursor.fetchall()
917
918         q = 'SELECT email FROM booking_agent'
919         cursor.execute(q)
920         t = cursor.fetchall()
921
922         return render_template('viewagents.html', s = s, y = y, c = c, t = t)
923         return redirect(url_for('prelogin'))
```

Management Tools

[Create New Flights](#)

[Flight Status Management](#)

[Add Airplane](#)

[Add Airport](#)

4-3. Frequent Customers (View frequent customers):

① 127.0.0.1:5000/staff/viewcus

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations

My Profile Logout

The most frequent customer in the last year is:

Name Email
Paul Ye carolyn.lefferts@nyu.edu

All the Delta Air Lines flights your customers have taken:

Email Flight No.
carolyn.lefferts@nyu.edu 179

Management Tools

[Create New Flights](#)
[Flight Status Management](#)
[Add Airplane](#)
[Add Airport](#)

Queries under viewcus():

```
q = 'SELECT name, email FROM customer JOIN purchases ON (customer.email = purchases.customer_email) \  
WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \  
GROUP BY email ORDER BY COUNT(ticket_id) DESC'  
cursor.execute(q.format(end_date_last, start_date_last))  
cus = cursor.fetchone()  
  
if cus is None:  
    msg1 = 'No customer bought ticket last year!'  
  
q = 'SELECT DISTINCT customer_email, flight_num FROM flight NATURAL JOIN ticket NATURAL JOIN purchases \  
WHERE airline_name = \'{}\' AND departure_time <= \'{}\' GROUP BY customer_email'  
cursor.execute(q.format(airline_name, taken_time))  
lst = cursor.fetchall()
```

4-4. Reports

① 127.0.0.1:5000/staff/viewreports

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations

My Profile Logout

View Reports

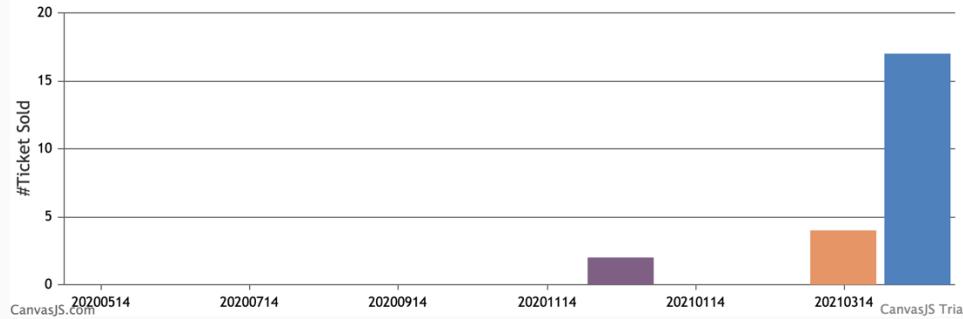
Please enter a range of dates (maximum 12 months)

Date From

Data range: 2020-05-14 - 2021-05-14

Total: 23

Month Wise Tickets Sold



Management Tools

[Create New Flights](#)
[Flight Status Management](#)
[Add Airplane](#)
[Add Airport](#)

Queries under viewreports():

Default View 12 months

```
q = "SELECT COUNT(ticket_id) FROM purchases NATURAL JOIN ticket\\
    WHERE airline_name = \'{\}\'
    AND purchase_date > \'{\}\' AND purchase_date <= \'{\}\'"

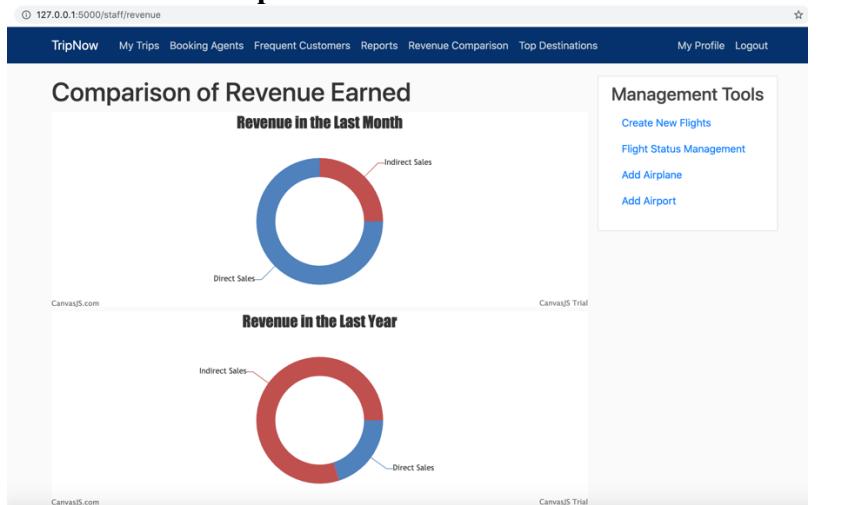
cursor.execute(q.format(airline, curdate, curdate1))
result.append(cursor.fetchone()[0])
```

Customized date range

```
while curdate <= date1:
    q = "SELECT COUNT(ticket_id) FROM purchases NATURAL JOIN ticket\\
        WHERE airline_name = \'{\}\'
        AND purchase_date > \'{\}\' AND purchase_date <= \'{\}\'"

    cursor.execute(q.format(airline, str(curdate)[0:10], str(curdate1)[0:10]))
    result_temp.append(cursor.fetchone()[0])
```

4-5. Revenue Comparison



Queries under revenue(): q1 q2 for last month, q3 q4 for last year

```
q1 = "SELECT count(ticket_id) FROM purchases NATURAL JOIN ticket\\
    WHERE booking_agent_id is NULL\\
    AND airline_name = \'{\}\'
    AND purchase_date <= \'{\}\' AND purchase_date >= \'{\}\'"

cursor.execute(q1.format(airline, end_date_last_month, start_date_last_month))
out1 = cursor.fetchall()

q2 = "SELECT count(ticket_id) FROM purchases NATURAL JOIN ticket\\
    WHERE booking_agent_id is not NULL\\
    AND airline_name = \'{\}\'
    AND purchase_date <= \'{\}\' AND purchase_date >= \'{\}\'"

cursor.execute(q2.format(airline, end_date_last_month, start_date_last_month))
out2 = cursor.fetchall()

num1["Direct Sales"] = out1[0][0]
num1["Indirect Sales"] = out2[0][0]

q3 = "SELECT count(ticket_id) FROM purchases NATURAL JOIN ticket\\
    WHERE booking_agent_id is NULL\\
    AND airline_name = \'{\}\'
    AND purchase_date <= \'{\}\' AND purchase_date >= \'{\}\'"

cursor.execute(q3.format(airline, end_date_last_year, start_date_last_year))
out3 = cursor.fetchall()

q4 = "SELECT count(ticket_id) FROM purchases NATURAL JOIN ticket\\
    WHERE booking_agent_id is NOT NULL\\
    AND airline_name = \'{\}\'
    AND purchase_date <= \'{\}\' AND purchase_date >= \'{\}\'"

cursor.execute(q4.format(airline, end_date_last_year, start_date_last_year))
out4 = cursor.fetchall()

num2["Direct Sales"] = out3[0][0]
num2["Indirect Sales"] = out4[0][0]
```

4-6. Top Destinations

127.0.0.1:5000/staff/topdest

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations My Profile Logout

Top Destinations

Top 3 destinations from 2021-02-13 to 2021-05-14:

Shanghai

Top 3 destinations from 2020-05-14 to 2021-05-14:

Shanghai

Management Tools

Create New Flights
Flight Status Management
Add Airplane
Add Airport

Queries under topdest():

```
q = 'SELECT airport_city FROM flight JOIN airport ON (flight.arrival_airport = airport.airport_name) \
    NATURAL JOIN ticket NATURAL JOIN purchases \
    WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \
    GROUP BY airport_city ORDER BY COUNT(*) DESC LIMIT 3'
cursor.execute(q.format(end_date, start_date_year))
year = cursor.fetchall()

q = 'SELECT airport_city FROM flight JOIN airport ON (flight.arrival_airport = airport.airport_name) \
    NATURAL JOIN ticket NATURAL JOIN purchases \
    WHERE purchase_date <= \'{}\' AND purchase_date >= \'{}\' \
    GROUP BY airport_city ORDER BY COUNT(*) DESC LIMIT 3'
cursor.execute(q.format(end_date, start_date_3month))
month = cursor.fetchall()
```

4-7. Create New Flights

You can preview the flights in the coming 30 days.

127.0.0.1:5000/staff/createflights

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations My Profile Logout

Create Flights

Flights in the Next 30 Days

Airline	Flight No.	Dep. Airport	Dep. Time	Arr. Airport	Arr. Time	Price	Status	Airplane ID
Delta Air Lines	22	PVG	2021-05-20 13:32:00	PEK	2021-05-20 15:39:00	132	DELAYED	123
Delta Air Lines	154	SEA	2021-05-30 12:21:00	DTW	2021-05-30 14:30:00	275	UPCOMING	123

Management Tools

Create New Flights
Flight Status Management
Add Airplane
Add Airport

Queries under createflights():

Preview coming flights:

```
q = 'SELECT * \
    FROM flight WHERE departure_time >= \'{}\' AND departure_time <= \'{}\' AND airline_name = \'{}\''
cursor.execute(q.format(start_date, end_date, airline_name))
data = cursor.fetchall()
```

Insert after error checking:

```
msg = 'Number of seats should not exceed 11 characters!'
else:
    # Account doesn't exist and the form data is valid, now insert new account into accounts table
    ins = "INSERT INTO flight VALUES( \'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\' )"
    cursor.execute(ins.format(airline_name, flight_num, departure_airport, departure_time, arrival_airport, \
        arrival_time, price, status, airplane_id))
    conn.commit()
    msg = 'You have successfully added the flight!'
```

4-8. Flight Status Management (Change status)

Queries under statuschange():

Change status while displaying all the flight status

```
cursor.execute("SELECT * FROM flight")
data = cursor.fetchall()

if request.method == 'POST':
    airline_name = session['airline name']
    flight_num = request.form['flight num']
    changed_status = request.form['status']

    query1 = "SELECT status FROM flight WHERE airline_name = \'{}\' AND flight_num = \'{}\'"
    cursor.execute(query1.format(airline_name, flight_num))
    current_status = cursor.fetchone()

    if (not current_status):
        msg = 'This flight does not exist!'
    elif (current_status[0] == changed_status):
        msg = 'The status does not need to be changed!'
    else:
        up = "UPDATE flight SET status = \'{}\' WHERE airline_name = \'{}\' AND flight_num = \'{}\'"
        cursor.execute(up.format(changed_status, airline_name, flight_num))
        conn.commit()
        msg = 'You have successfully changed the status!'

cursor.execute("SELECT * FROM flight")
data = cursor.fetchall()
```

4-9. Add Airplane

127.0.0.1:5000/staff/addplane

TripNow My Trips Booking Agents Frequent Customers Reports Revenue Comparison Top Destinations My Profile Logout

Add Airplane

Current Airplanes

Airline	Airplane ID	Seats
Delta Air Lines	123	123
Delta Air Lines	144	2
Delta Air Lines	532	2
Delta Air Lines	7162	281
Delta Air Lines	9234	172

Management Tools

- Create New Flights
- Flight Status Management
- Add Airplane**
- Add Airport

Airplane ID
Enter Airplane ID

Number of seats
Enter the number of seats

Add Airplane

Queries under addplane():

Add plane while displaying all the existing planes

```
cursor.execute("SELECT * FROM airplane")
data = cursor.fetchall()
if request.method == 'POST':
    # Create variables for easy access
    airline_name = session['airline name']
    airplane_id = request.form['airplane id']
    num_of_seats = request.form['seats']

    #Check if airplane exists using MySQL
    query1 = "SELECT * FROM airplane WHERE airline_name = \'{}\' AND airplane_id = \'{}\'"
    cursor.execute(query1.format(airline_name, airplane_id))
    airplane = cursor.fetchone()

    query2 = "SELECT * FROM airline WHERE airline_name = \'{}\'"
    cursor.execute(query2.format(airline_name))
    airline = cursor.fetchone()

    # If airplane exists show error and validation checks
    if airplane:
        msg = 'This airplane already exists!'
    elif (not airline):
        msg = 'This airline does not exist!'
    # elif len(airline_name) > 50:
    #     msg = 'The Airline Name should be less than 50 characters!'
    elif len(airplane_id) > 11:
        msg = 'The Airplane ID should not exceed 11 characters!'
    # type changed to varchar(11)
    elif len(num_of_seats) > 11:
        msg = 'Number of seats should not exceed 11 characters!'

    else:
        # Account doesn't exist and the form data is valid, now insert new account into accounts table
        ins = "INSERT INTO airplane VALUES( \'{}\', \'{}\', \'{}\' )"
        cursor.execute(ins.format(airline_name, airplane_id, num_of_seats))
        conn.commit()
        msg = 'You have successfully added the airplane!'

cursor.execute("SELECT * FROM airplane")
data = cursor.fetchall()
```

4-10. Add Airport

YVR	Vancouver
YYZ	Toronto

Airport Name

Airport City

Add Airport

Queries under addairport():

Add airport while displaying all the existing airports

```
cursor.execute("SELECT * FROM airport")
data = cursor.fetchall()
if request.method == 'POST':

    # Create variables for easy access
    airport_name = request.form['airport name']
    airport_city = request.form['airport city']

    #Check if airport_name exists using MySQL
    query1 = "SELECT * FROM airport WHERE airport_name = \'{0\'"
    cursor.execute(query1.format(airport_name))
    airport = cursor.fetchone()

    # If airplane exists show error and validation checks
    if airport:
        msg = 'This airport already exists!'
    elif len(airport_name) >= 50:
        msg = 'The Airport Name should be less than 50 characters!'
    elif len(airport_city) >= 50:
        msg = 'The Airport City Name should be less than 50 characters!'
    else:
        # Account doesn't exist and the form data is valid, now insert new account into accounts table
        ins = "INSERT INTO airport VALUES( \'{0\'', \'{1\'}"
        cursor.execute(ins.format(airport_name, airport_city))
        conn.commit()
        msg = 'You have successfully added the airport!'

    cursor.execute("SELECT * FROM airport")
    data = cursor.fetchall()
```

4-11. My Profile

Similar to booking agent and customer. Queries under profilestaff().

4-12. Logout: logout()