Assignment 3 Individual Report

SID:

530490838

Part A

Contents:

Note: copy paste of code is in appendix (pg 26)

Security Goals for Aircrafts

Confidentiality	Any user is able to view all aircrafts and their details.
Integrity	Only admin users are able to modify and delete aircrafts from the database This ensures that unauthorised users don't modify information they are not supposed to. Additionally, when an admin is modifying an aircraft, the ID (primary key) is set to be read only so that admins don't change the information of another aircraft.
Availability	Users and admins are able to view all aircrafts, count of manufacturers and manufacturers. Only the admin is allowed to add, modify and delete aircrafts. Additionally, the application should be available at all times however due to server overload, if many users try to access the same database, it results in an error. Unfortunately, we could not mitigate this issue as it is related to the server.

HTML Pages added:

- **add_aircraft.html**: Utilised the add_user.py page as a template
- **count_aircraft.html**
- **list_aircrafts.html**: Utilised the list_consolidated_users.html as a template
- **list_manufacturers.html**
- **modify_aircraft.html**: Utilised the edit_users.py as a template

add_aircrafts.html

```
{% include 'top.html' %}
<div class="content">
    <div class="container my-4">

        <h2 class="title"> Add Aircraft </h2>

        <form class="needs-validation" method="POST" action="{{ url_for('add_aircraft_route') }}" novalidate>

            <div class="form-group">
                <label for="aircraft_id">Aircraft ID:</label>
                <input type="text" class="form-control" id="aircraft_id" name="aircraft_id" placeholder="Enter aircraft ID here" required>
                <div class="invalid-feedback">Please enter an aircraft ID.</div>
            </div>

            <div class="form-group">
                <label for="icao_code">ICAO Code:</label>
                <input type="text" class="form-control" id="icao_code" name="icao_code" placeholder="Enter ICAO code here" required>
                <div class="invalid-feedback">Please enter an ICAO code.</div>
            </div>

            <div class="form-group">
                <label for="aircraft_registration">Aircraft Registration:</label>
                <input type="text" class="form-control" id="aircraft_registration" name="aircraft_registration" placeholder="Enter aircraft registration here" required>
                <div class="invalid-feedback">Please enter an aircraft registration.</div>
            </div>

            <div class="form-group">
                <label for="manufacturer">Manufacturer:</label>
                <input type="text" class="form-control" id="manufacturer" name="manufacturer" placeholder="Enter manufacturer here" required>
                <div class="invalid-feedback">Please enter a manufacturer.</div>
            </div>

            <div class="form-group">
                <label for="model">Model:</label>
                <input type="text" class="form-control" id="model" name="model" placeholder="Enter model here" required>
                <div class="invalid-feedback">Please enter a model.</div>
            </div>

            <div class="form-group">
                <label for="name">Capacity:</label>
                <input type="text" class="form-control" id="capacity" name="capacity" placeholder="Enter aircraft capacity here" required>
                <div class="invalid-feedback">Please enter the aircraft capacity.</div>
            </div>

            <button class="btn btn-primary" type="submit">Add Aircraft</button>

        </form>

    </div>
</div>
{% include 'end.html' %}
```

A form for adding an aircraft to a system. A submit button is used to send the data via POST to the add_aircraft_route

count_aircrafts.html

```
{% include 'top.html' %}

<div class="container my-4">
    <h2 class="page-title">Aircraft Count by Manufacturer</h2>

    <table class="table table-bordered">
        <thead class="thead-light">
            <tr>
                <th>Manufacturer</th>
                <th>Number of Aircraft</th>
            </tr>
        </thead>
        <tbody>
            {% for manufacturer, count in counts %}
            <tr>
                <td>{{ manufacturer }}</td>
                <td>{{ count }}</td>
            </tr>
            {% endfor %}
        </tbody>
    </table>
</div>

{% include 'end.html' %}
```

A table showing the number of aircraft grouped by manufacturer

list_aircrafts.html

```
{% include 'top.html' %}

<div id="content" class="container my-4">
    <h1 class="page-title">{{page.get('title', 'Aircraft List')}}</h1>

    <!-- Search Bar -->
    <div class="mb-3">
        <form action="{{ url_for('search_aircraft_route') }}" method="GET">
            <div class="input-group">
                <input type="text" class="form-control" name="query" placeholder="Search aircraft..." aria-label="Search aircraft">
                <button class="btn btn-outline-secondary" type="submit">Search</button>
            </div>
        </form>
    </div>

    <!-- Table -->
    <table class="table table-striped">
        <!-- Ascending/Descending patterns -->
        <thead>
            <tr>
                {% if session['isadmin'] == True %}
                    <th>Delete</th>
                {% endif %}
                <th>
                    Aircraft ID
                    <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>
                <th>
                    ICAO Code
                    <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>
                <th>
                    Registration
                    <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>
                <th>
                    Manufacturer
                    <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>
                <th>
                    Model
                    <a href="{{ url_for('list_aircrafts_route', sort='model', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='model', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>
                <th>
                    Capacity
                    <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
                    <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='desc') }}" aria-label="Sort descending">&#8595;</a>
                </th>

                {% if session['isadmin'] == True %}
                    <th>Update</th>
                {% endif %}
            </tr>
        </thead>
```

```
        <!-- List Aircrafts -->
        <tbody>
        {% for aircraft in aircrafts %}
            <tr class="align-items-center">
                {% if session['isadmin'] == True %}
                    <!-- Delete button for admin users only -->
                    <td>
                        <form action="{{ url_for('remove_aircraft_route') }}" method="POST" style="display:inline;">
                            <input type="hidden" name="aircraft_id" value="{{ aircraft['aircraftid'] }}">
                            <button type="submit" class="btn btn-danger">Delete</button>
                        </form>
                    </td>
                    <td class="align-middle">{{ aircraft['aircraftid'] }}</td>
                    <td class="align-middle">{{ aircraft['icaocode'] }}</td>
                    <td class="align-middle">{{ aircraft['aircraftregistration'] }}</td>
                    <td class="align-middle">{{ aircraft['manufacturer'] }}</td>
                    <td class="align-middle">{{ aircraft['model'] }}</td>
                    <td class="align-middle">{{ aircraft['capacity'] }}</td>

                    <td><a class="btn btn-primary" href="{{ url_for('modify_aircraft_route', aircraft_id=aircraft['aircraftid']) }}">Edit</a></td>
                {% else %}
                    <td class="align-middle">{{ aircraft['aircraftid'] }}</td>
                    <td class="align-middle">{{ aircraft['icaocode'] }}</td>
                    <td class="align-middle">{{ aircraft['aircraftregistration'] }}</td>
                    <td class="align-middle">{{ aircraft['manufacturer'] }}</td>
                    <td class="align-middle">{{ aircraft['model'] }}</td>
                    <td class="align-middle">{{ aircraft['capacity'] }}</td>

                {% endif %}
            </tr>
        {% endfor %}
        </tbody>
    </table>
<nav aria-label="Aircraft pagination">
    <ul class="pagination">
        <!-- Previous Button -->
        {% if page.current_page > 1 %}
            <li class="page-item">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page - 1) }}" aria-label="Previous">
                    <span aria-hidden="true">&laquo;</span>
                </a>
            </li>
        {% endif %}

        <!-- First Page -->
        <li class="page-item {% if page.current_page == 1 %}active{% endif %}">
            <a class="page-link" href="{{ url_for('list_aircrafts_route', page=1) }}">1</a>
        </li>

        <!-- Ellipsis after Page 1 -->
        {% if page.current_page > 3 %}
            <li class="page-item disabled">
                <span class="page-link">...</span>
            </li>
        {% endif %}

        <!-- Middle Pages  -->
        {% set start_page = 2 if page.current_page - 1 < 2 else page.current_page - 1 %}
        {% set end_page = page.total_pages if page.current_page + 1 >= page.total_pages else page.current_page + 1 %}

        {% for p in range(start_page, end_page + 1) %}
            <li class="page-item {% if page.current_page == p %}active{% endif %}">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=p) }}">{{ p }}</a>
            </li>
        {% endfor %}

        <!-- Ellipsis before Last Page -->
        {% if page.current_page < page.total_pages - 2 %}
            <li class="page-item disabled">
                <span class="page-link">...</span>
            </li>
        {% endif %}

        <!-- Last Page -->
        {% if page.total_pages > 1 %}
            <li class="page-item {% if page.current_page == page.total_pages %}active{% endif %}">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.total_pages) }}">{{ page.total_pages }}</a>
            </li>
        {% endif %}

        <!-- Next Button -->
        {% if page.current_page < page.total_pages %}
            <li class="page-item">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page + 1) }}" aria-label="Next">
                    <span aria-hidden="true">&raquo;</span>
                </a>
            </li>
        {% endif %}
    </ul>
</nav>
```

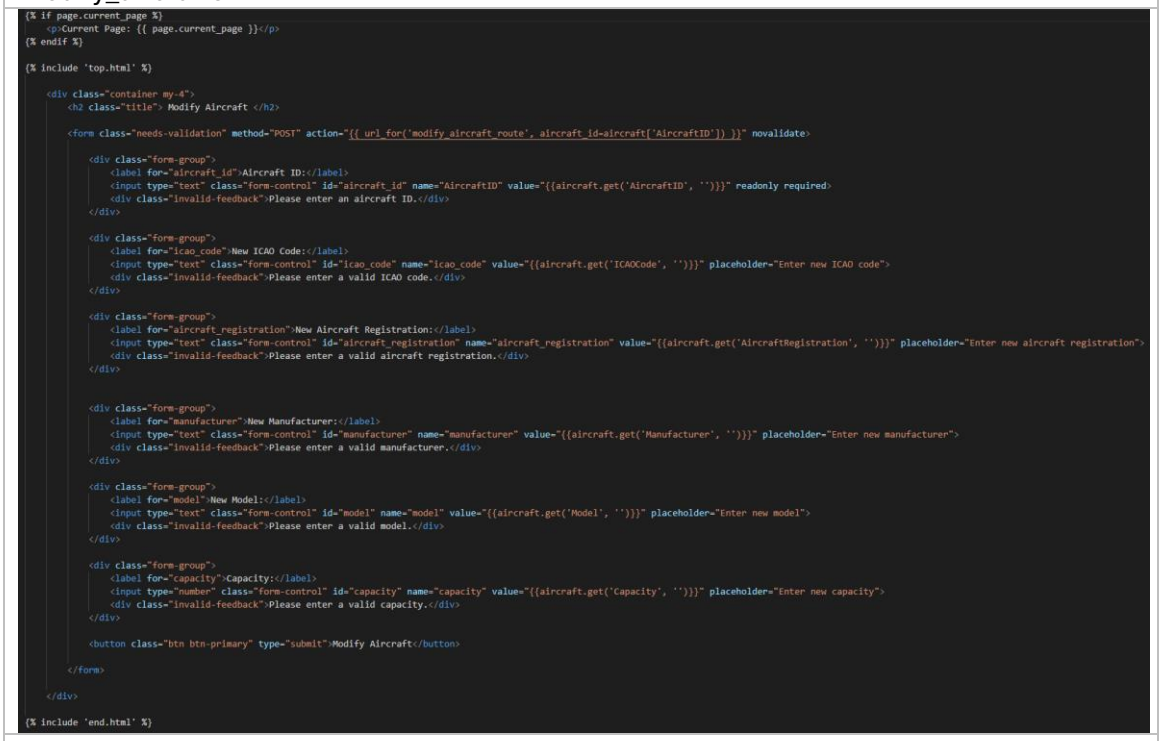
Displays a list of aircraft with a search bar, sortable columns, and numbered pages to list 5 aircrafts at a time. Admin users can delete or update aircraft records, while other users can only view the aircraft details. To view 5 aircrafts at a time, pagination was used to allow users to navigate through pages of aircraft.

list_manufacturers.html

```
{% include 'top.html' %}

<div class="container my-4">
    <h2 class="page-title">Manufacturers</h2>

    <ul class="list-group">
        {% for manufacturer in manufacturers %}
        <li class="list-group-item">{{ manufacturer }}</li>
        {% endfor %}
    </ul>
</div>

{% include 'end.html' %}
```

Displays a list of manufacturers

modify_aircrafts.html



```
{% if page.current_page %}
    <p>Current Page: {{ page.current_page }}</p>
{% endif %}

{% include 'top.html' %}

    <div class="container my-4">
        <h2 class="title"> Modify Aircraft </h2>

        <form class="needs-validation" method="POST" action="{{ url_for('modify_aircraft_route', aircraft_id=aircraft['AircraftID']) }}" novalidate>

            <div class="form-group">
                <label for="aircraft_id">Aircraft ID:</label>
                <input type="text" class="form-control" id="aircraft_id" name="AircraftID" value="{{aircraft.get('AircraftID', '')}}" readonly required>
                <div class="invalid-feedback">Please enter an aircraft ID.</div>
            </div>

            <div class="form-group">
                <label for="icao_code">New ICAO Code:</label>
                <input type="text" class="form-control" id="icao_code" name="icao_code" value="{{aircraft.get('ICAOCode', '')}}" placeholder="Enter new ICAO code">
                <div class="invalid-feedback">Please enter a valid ICAO code.</div>
            </div>

            <div class="form-group">
                <label for="aircraft_registration">New Aircraft Registration:</label>
                <input type="text" class="form-control" id="aircraft_registration" name="aircraft_registration" value="{{aircraft.get('AircraftRegistration', '')}}" placeholder="Enter new aircraft registration">
                <div class="invalid-feedback">Please enter a valid aircraft registration.</div>
            </div>


            <div class="form-group">
                <label for="manufacturer">New Manufacturer:</label>
                <input type="text" class="form-control" id="manufacturer" name="manufacturer" value="{{aircraft.get('Manufacturer', '')}}" placeholder="Enter new manufacturer">
                <div class="invalid-feedback">Please enter a valid manufacturer.</div>
            </div>

            <div class="form-group">
                <label for="model">New Model:</label>
                <input type="text" class="form-control" id="model" name="model" value="{{aircraft.get('Model', '')}}" placeholder="Enter new model">
                <div class="invalid-feedback">Please enter a valid model.</div>
            </div>

            <div class="form-group">
                <label for="capacity">Capacity:</label>
                <input type="number" class="form-control" id="capacity" name="capacity" value="{{aircraft.get('Capacity', '')}}" placeholder="Enter new capacity">
                <div class="invalid-feedback">Please enter a valid capacity.</div>
            </div>

            <button class="btn btn-primary" type="submit">Modify Aircraft</button>

        </form>

    </div>

{% include 'end.html' %}
```

A form includeing pre-filled fields for the aircraft's ID (set to read-only) and fields available to update. When the user changes these details, a POST request is sent to the server to update the aircraft information

Aircrafts Schema:

In the original schema for Aircrafts, there was a name field which did not correspond to any column in the provided data. This was removed and replaced with capacity to better suit the given data.

Routes.py:

Additional code was added to handle aircraft pages and connecting the html to the database through python and flask. The functions that were implemented are described below:

Extracting data from the forms on the webpages

```
'''
Extract relevant data from the form (page) for aircrafts
Returns
    - a dictionary of updated values
    - boolean indicating if any updates were made
'''
def extract_aircraft_data(form):
    update_data = {
        'AircraftID': form.get('AircraftID'),
        'ICAOCode': form.get('icao_code'),
        'AircraftRegistration': form.get('aircraft_registration'),
        'Manufacturer': form.get('manufacturer'),
        'Model': form.get('model'),
        'Capacity': form.get('capacity')
    }
    valid_update = any(value for value in update_data.values() if value)
    return update_data, valid_update
```

This function is responsible for getting the information from the webpage and storing it into a hashmap of data representing fields in the database

Adding aircrafts

```python
'''
Add a new aircraft to the database
Return
    - a form to add new aircraft and handles submissions
'''
@app.route('/add_aircraft', methods=['GET', 'POST'])
def add_aircraft_route():
    if request.method == 'POST':
        print("Form data received:", request.form)

        aircraft_id = request.form.get('AircraftID', '').strip()
        icao_code = request.form.get('icao_code', '').strip()
        aircraft_registration = request.form.get('aircraft_registration', '').strip()
        capacity = request.form.get('capacity', '').strip()

        #check id is integer
        try:
            aircraft_id_value = int(aircraft_id)
        except ValueError:
            flash("Error: Aircraft ID must be an integer.", "error")
            return redirect(url_for('add_aircraft_route'))

        #check icao code
        if icao_code and len(icao_code) != 4:
            flash("Error: ICAO code must be exactly 4 characters.", "error")
            return redirect(url_for('add_aircraft_route'))

        #check aircraft registration
        if aircraft_registration and not re.match(r'^[A-Za-z0-9]{2}-[A-Za-z0-9]{3}$', aircraft_registration):
            flash("Error: Aircraft registration must be in the format ##-### (letters or numbers).", "error")
            return redirect(url_for('add_aircraft_route'))

        #check capacity
        try:
            capacity_value = int(capacity)
            if capacity_value < 0:
                flash("Error: Capacity cannot be negative.", "error")
                return redirect(url_for('add_aircraft_route'))
        except ValueError:
            flash("Error: Capacity must be a valid number.", "error")
            return redirect(url_for('add_aircraft_route'))

        #check aircraft ID exists
        if (aircraft_exists(aircraft_id)):
            flash("Error: Aircraft ID already exists.", "error")
            return redirect(url_for('add_aircraft_route'))


        aircraft_data, _ = extract_aircraft_data_add(request.form)

        add_aircraft(
            aircraft_data['AircraftID'],
            aircraft_data.get('ICAOCode'),
            aircraft_data.get('AircraftRegistration'),
            aircraft_data.get('Manufacturer'),
            aircraft_data.get('Model'),
            aircraft_data.get('Capacity')
        )

        return redirect(url_for('add_aircraft_route'))

    return render_template('add_aircraft.html', page={'title': 'Add Aircraft'})
```

The function is responsible for adding an aircraft entry to the database. The function listens for GET and POST requests at the URL '/add_aircraft'.

Deleting aircraft

```
'''
Delete an aircraft from the database
Return
    - redirects back to the aircraft list with message after deletion
'''
@app.route('/aircrafts/delete/<aircraft_id>', methods=['POST'])
def delete_aircraft(aircraft_id):
    if 'logged_in' not in session or not session['logged_in']:
        return redirect(url_for('login'))

    remove_aircraft(aircraft_id)
    flash(f'Aircraft {aircraft_id} has been successfully deleted.')
    return redirect(url_for('list_aircrafts'))
```

The function is responsible for removing an aircraft entry from the database. The function listens for a POST request at the URL '/aircrafts/delete/<aircraft_id>'.

Modifying aircraft

```
@app.route('/modify_aircraft', methods=['GET', 'POST'])
@app.route('/aircrafts/edit/<aircraft_id>', methods=['GET', 'POST'])
def modify_aircraft_route(aircraft_id=None):
    if 'logged_in' not in session or not session['logged_in']:
        return redirect(url_for('login'))

    page_title = 'Modify Aircraft'
    aircraft_details = {}

    if aircraft_id:
        page_title = 'Edit Aircraft Details'
        aircraft_details = get_aircraft_by_id(aircraft_id)
        if aircraft_details is None:
            flash(f'Error: No aircraft found with ID {aircraft_id}')
            return redirect(url_for('list_aircrafts'))

    icao_code = request.form.get('icao_code', '').strip()
    if icao_code and len(icao_code) != 4:
        flash("Error: ICAO code must be exactly 4 characters.", "error")
        return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))

    if request.method == 'POST':
        if 'AircraftID' not in request.form:
            flash('Error: Aircraft ID is required.')
            return redirect(url_for('list_aircrafts'))

        #check icao code (exactly 4 characters)
        icao_code = request.form.get('icao_code', '').strip()
        if icao_code and len(icao_code) != 4:
            flash("Error: ICAO code must be exactly 4 characters.", "error")
            return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))

        #chech regristration in the form ##-###
        aircraft_registration = request.form.get('aircraft_registration', '').strip()
        if aircraft_registration and not re.match(r'^[A-Za-z0-9]{2}-[A-Za-z0-9]{3}$', aircraft_registration):
            flash("Error: Aircraft registration must be in the format ##-### (letters or numbers).", "error")
            return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))
```

```
        #check capacity not negative
        capacity = request.form.get('capacity', '').strip()
        try:
            capacity_value = int(capacity)
            if capacity_value < 0:
                flash("Error: Capacity cannot be negative.", "error")
                return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))
        except ValueError:
            flash("Error: Capacity must be a valid number.", "error")
            return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))

        update_data, valid_update = extract_aircraft_data(request.form)

        if valid_update:
            modify_aircraft(
                update_data['AircraftID'],
                update_data.get('ICAOCode'),
                update_data.get('AircraftRegistration'),
                update_data.get('Manufacturer'),
                update_data.get('Model'),
                update_data.get('Capacity')
            )
            flash(f'Aircraft {update_data["AircraftID"]} updated successfully.')
            return redirect(url_for('list_aircrafts_route'))
        else:
            flash("No updated values for aircraft.")
            return redirect(url_for('modify_aircraft', aircraft_id=aircraft_id))

    current_page = 'modify_aircraft'
    return render_template(
        'modify_aircraft.html',
        session=session,
        page={'title': page_title, 'current_page': current_page},
        aircraft=aircraft_details
    )
```

The function is responsible for handling the modification of existing aircraft details. This function listens for GET and POST requests at the URLs '/modify_aircraft' and '/aircrafts/edit/<aircraft_id>'

Removing aircraft

```
'''
Remove an aircraft from the database
'''
@app.route('/remove_aircraft/<aircraft_id>', methods=['GET', 'POST'])
@app.route('/remove_aircraft', methods=['GET', 'POST'])
def remove_aircraft_route():
    if request.method == 'POST':
        aircraft_id = request.form['aircraft_id']
        remove_aircraft(aircraft_id)
        flash(f'Aircraft {aircraft_id} has been successfully deleted.')
        return redirect(url_for('list_aircrafts_route'))
    return render_template('remove_aircraft.html', page={'title': 'Remove Aircraft'})
```

The function handles the removal of an aircraft from the system. This function listens for both GET and POST requests at the URLs '/remove_aircraft/<aircraft_id>' and '/remove_aircraft'.

Lists the manufacturers of aircrafts

```
'''
List all unique aircraft manufacturers from the database
Return
    - a page showing the distinct manufacturers
'''
@app.route('/list_manufacturers')
def list_manufacturers_route():
    manufacturers = get_unique_manufacturers()
    return render_template('list_manufacturers.html', manufacturers=manufacturers, page={'title': 'List Manufacturers'})
```

The function is responsible for retrieving and displaying all unique aircraft manufacturers from the database. This function listens for GET requests at the URL '/list_manufacturers'.

List the number of aircrafts each manufacturer makes

```
'''
Count the number of aircrafts by each manufacturer
Return
    - a list of manufacturers and the number of aircraft each one has
'''
@app.route('/count_aircraft')
def count_aircraft_route():
    counts = get_aircraft_count_by_manufacturer()
    return render_template('count_aircraft.html', counts=counts, page={'title': 'Count Aircraft by Manufacturer'})
```

The function is designed to count the number of aircraft associated with each manufacturer and display the results. This function listens for GET requests at the URL '/count_aircraft'.

List all the aircrafts in the database

```
'''
List all aircraft currently in the database
Return
    - a list of aircraft with sorting options for fields
'''
@app.route('/list_aircrafts')
def list_aircrafts_route():
    page = request.args.get('page', 1, type=int)
    per_page = 5

    sort = request.args.get('sort', 'aircraftid')  #default sort by Aircraft ID asc
    order = request.args.get('order', 'asc')

    total_aircrafts = list_aircrafts()

    if order == 'asc':
        total_aircrafts.sort(key=lambda x: x.get(sort, ''))
    else:
        total_aircrafts.sort(key=lambda x: x.get(sort, ''), reverse=True)

    total_pages = (len(total_aircrafts) + per_page - 1) // per_page

    start = (page - 1) * per_page
    end = start + per_page
    aircraft_list = total_aircrafts[start:end]

    return render_template('list_aircrafts.html',
                           page={'title': 'List of Aircraft', 'current_page': page, 'total_pages': total_pages},
                           session=session,
                           aircrafts=aircraft_list)
```

The function is responsible for displaying a paginated list of aircraft currently in the database. The function also allows users to sort the list by various fields and listens for GET requests at the URL '/list_aircrafts'.

Searches all fields depending on user input

```python
'''
Search for an aircraft by all fields
Return
    - Filtered aircrafts
'''
@app.route('/search_aircraft', methods=['GET'])
def search_aircraft_route():
    query = request.args.get('query', '').strip().lower()

    aircrafts = list_aircrafts()

    filtered_aircrafts = [
        aircraft for aircraft in aircrafts if (
            query in str(aircraft['aircraftid']).lower() or
            query in aircraft['icaocode'].lower() or
            query in aircraft['aircraftregistration'].lower() or
            query in aircraft['manufacturer'].lower() or
            query in aircraft['model'].lower() or
            query in str(aircraft['capacity']).lower()
        )
    ]

    return render_template('list_aircrafts.html',
                           page={'title': 'Search Results', 'current_page': 1, 'total_pages': 1},
                           session=session,
                           aircrafts=filtered_aircrafts)
```

The function is designed to search for an aircraft by any field. It listens for GET requests at the URL '/search_aircraft'.

Display a single aircraft

```python
'''
Display the details of a single aircraft
Return
    - details of a specific aircraft by ID
'''
@app.route('/aircraft/<aircraft_id>')
def list_single_aircraft(aircraft_id):
    aircraft_details = get_aircraft_by_id(aircraft_id)
    if not aircraft_details:
        flash(f'Error: No aircraft found with ID {aircraft_id}')
        return redirect(url_for('list_aircrafts'))

    return render_template('list_aircrafts.html', aircraft=aircraft_details, page={'title': f'Aircraft {aircraft_id} Details'})
```

The function displays the details of a specific aircraft. It listens for GET requests at the URL '/aircraft/<aircraft_id>'

Top.html

```html
<!-- Dropdown for managing/viewing aircraft -->
{% if session.logged_in %}
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" href="#" id="aircraftDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expa
    View Aircrafts
  </a>
  <div class="dropdown-menu" aria-labelledby="aircraftDropdown">
    <a class="dropdown-item" href="{{ url_for('list_aircrafts_route') }}">List Aircrafts</a>
    <a class="dropdown-item" href="{{ url_for('list_manufacturers_route') }}">Count Manufacturers</a>
    <a class="dropdown-item" href="{{ url_for('count_aircraft_route') }}">Count Aircrafts</a>

    {% if session['isadmin'] == True %}
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="{{ url_for('add_aircraft_route') }}">Add Aircraft</a>
    {% endif %}
  </div>
</li>
{% endif %}
```

The above code was added to top.html so that users and admins can access other pages to view and manipulate aircraft data. The admin is able to see all pages and the user is able to see all except the add aircrafts page.

Aircrafts.py:

The code handling logic for adding, modifying, deleting and getting information about aircrafts were not added to database.py and instead was added to aircrafts.py to prevent code cluttering. The functions that were implemented are described below:

Adding aircrafts

```
'''
Adds aircraft to the table
'''
def add_aircraft(aircraft_id, icao_code, aircraft_registration, manufacturer, model, capacity):
    if aircraft_exists(aircraft_id):
        print(f"Error: fail to add aircraft. {aircraft_id} already exists")
        return
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            """
            INSERT INTO Aircraft (AircraftID, ICAOCode, AircraftRegistration, Manufacturer, Model, Capacity)
            VALUES (%s, %s, %s, %s, %s, %s)
            """,
            (aircraft_id, icao_code, aircraft_registration, manufacturer, model, capacity)
        )

        conn.commit()
        print("Aircraft added successfully.")

    except:
        print(f"Error: fail to add aircraft {aircraft_id}")
        conn.rollback()
    finally:
        cursor.close()
        conn.close()
```

This function was implemented to handle logic relating to adding a new aircraft into the database. This function checks to see if the aircraft exists before adding it to the table. Additionally, catch statements were implemented to rollback a transaction if an error occurs so that the data entered into the database is consistent.

Removing an aircraft

```
'''
Remove an aircraft by id
'''
def remove_aircraft(aircraft_id):
    if not aircraft_exists(aircraft_id):
        print(f"Error: fail to remove aircraft. {aircraft_id} does not exist")
        return
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "DELETE FROM Aircraft WHERE AircraftID = %s",
            (aircraft_id,)
        )

        conn.commit()
        print("Aircraft removed successfully.")

    except:
        print(f"Error: fail to remove aircraft {aircraft_id}")
        conn.rollback()
    finally:
        cursor.close()
        conn.close()
```

This function was implemented to handle the removal of an existing aircraft. This function checks to see if the aircraft exists before performing the deletion. Catch statements were implemented to rollback the transaction if an error occurs to ensure consistency of data in the database.

Modifying the data of an existing aircraft

```
'''
Edits an existing aircraft with a given id
'''
def modify_aircraft(aircraft_id, icao_code=None, aircraft_registration=None, manufacturer=None, model=None, capacity=None):
    if not aircraft_exists(aircraft_id):
        print(f"Error: fail to modify aircraft. {aircraft_id} does not exist")
        return

    try:
        conn = database_connect()
        cursor = conn.cursor()

        update_fields = []
        update_values = []

        if icao_code and icao_code.strip():
            update_fields.append("ICAOCode = %s")
            update_values.append(icao_code)

        if aircraft_registration and aircraft_registration.strip():
            update_fields.append("AircraftRegistration = %s")
            update_values.append(aircraft_registration)

        if manufacturer and manufacturer.strip():
            update_fields.append("Manufacturer = %s")
            update_values.append(manufacturer)

        if model and model.strip():
            update_fields.append("Model = %s")
            update_values.append(model)

        if capacity and capacity.strip():
            update_fields.append("Capacity = %s")
            update_values.append(capacity)
```

```
        if not update_fields:
            print("Error: No fields provided to update.")
            return

        update_values.append(aircraft_id)

        sql = f"UPDATE Aircraft SET {', '.join(update_fields)} WHERE AircraftID = %s"

        cursor.execute(sql, tuple(update_values))

        conn.commit()
        print("Aircraft updated successfully.")

    except:
        print(f"Error: Failed to update aircraft due to {aircraft_id}")
        conn.rollback()

    finally:
        cursor.close()
        conn.close()
```

This function was implemented to handle logic relating to modifying an existing aircraft in the database. This function checks to see if the aircraft exists before allowing the user to modify it. Additionally, catch statements were implemented to rollback a transaction if an error occurs so that the data entered is consistent.

Getting a list of manufacturers

```
'''
Gets a list of manufacturers
'''
def get_unique_manufacturers():
    manufacturers = []
    conn = None
    cursor = None

    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT DISTINCT Manufacturer FROM Aircraft"
        )

        manufacturers = cursor.fetchall()
        manufacturers = [manufacturer[0] for manufacturer in manufacturers]

    except Exception as e:
        manufacturers = []
        print(f"Error: fail to get manufacturers: {e}")
    finally:
        cursor.close()
        conn.close()

    return manufacturers
```

This function returns a list of manufacturers of the aircrafts.

Gets a count of the number of aircrafts with manufacturer

```
'''
Gets a list of manufacturers
'''
def get_unique_manufacturers():
    manufacturers = []
    conn = None
    cursor = None

    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT DISTINCT Manufacturer FROM Aircraft"
        )

        manufacturers = cursor.fetchall()
        manufacturers = [manufacturer[0] for manufacturer in manufacturers]

    except Exception as e:
        manufacturers = []
        print(f"Error: fail to get manufacturers: {e}")
    finally:
        cursor.close()
        conn.close()

    return manufacturers
```

This function returns the number of aircrafts with a manufacturer.

List aircrafts

```
'''
lists all aircrafts in database
'''
def list_aircrafts():
    conn = database_connect()
    if conn is None:
        return None

    cur = conn.cursor()
    returndict = None

    try:
        sql = """SELECT *
                 FROM Aircraft"""

        returndict = dictfetchall(cur, sql)

    except:
        import traceback
        traceback.print_exc()
        print("Error: fail to get aircrafts from database", sys.exc_info()[0])

    cur.close()
    conn.close()

    return returndict
```

This function returns all the aircrafts in the database

Gets aircrafts with id

```
'''
Gets aircraft details by aircraft ID
'''
def get_aircraft_by_id(aircraft_id):
    aircraft = None
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT * FROM Aircraft WHERE AircraftID = %s",
            (aircraft_id,)
        )

        aircraft = cursor.fetchone()

        if aircraft:
            aircraft = {
                "AircraftID": aircraft[0],
                "ICAOCode": aircraft[1],
                "AircraftRegistration": aircraft[2],
                "Manufacturer": aircraft[3],
                "Model": aircraft[4],
                "Capacity": aircraft[5]
            }
        else:
            print(f"Error: Aircraft with ID {aircraft_id} does not exist.")

    except Exception as e:
        print(f"Error: failed to get aircraft by ID {e}")
    finally:
        cursor.close()
        conn.close()

    print("AIRCRAFT WITH ID: ", aircraft)
    return aircraft
```

This function returns an aircrafts details with a provided id.

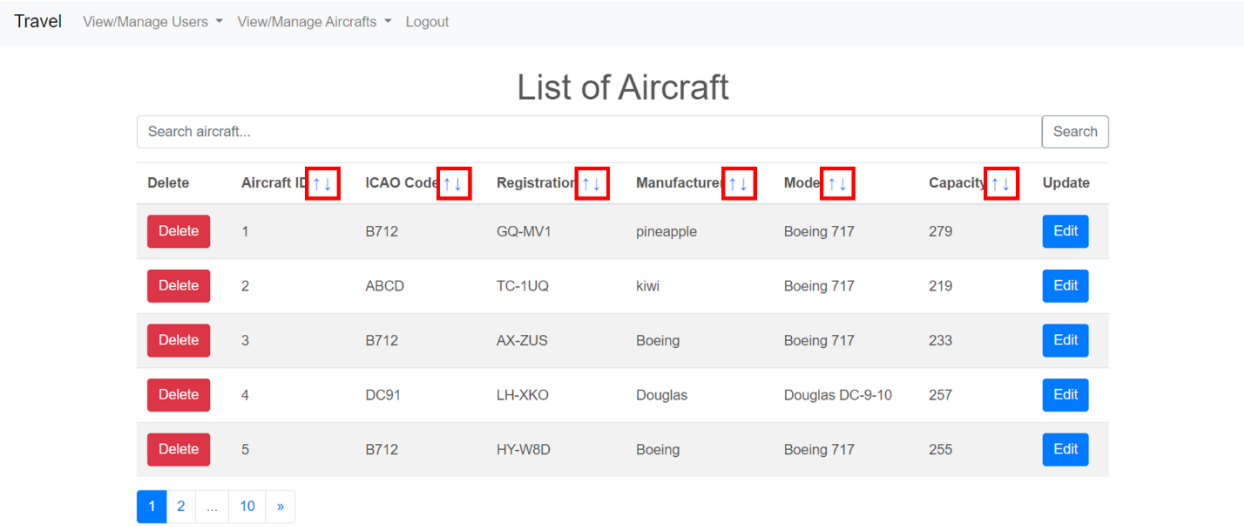
Checks if an aircraft exists in the database

```
'''
checks if an aircraft ID exists in the table
'''
def aircraft_exists(aircraft_id):
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT COUNT(*) FROM Aircraft WHERE AircraftID = %s",
            (aircraft_id,)
        )

        count = cursor.fetchone()[0]

        if count > 0:
            return True
        else:
            return False

    except Exception as e:
        print(f"Error: fail to check aircraft ID {e}")
        return False
    finally:
        cursor.close()
        conn.close()
```

This function returns a boolean to depending on whether an aircraft exists in the database or not.

Contents

Sorting by ascending/descending order:



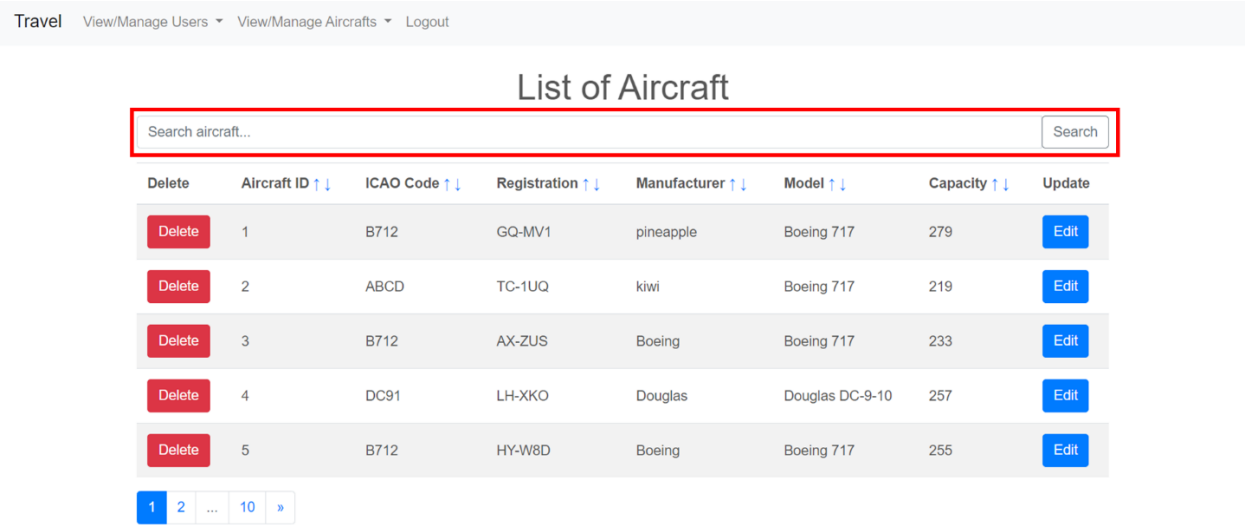
Routes.py

```
'''
List all aircraft currently in the database
Return
    - a list of aircraft with sorting options for fields
'''
@app.route('/list_aircrafts')
def list_aircrafts_route():
    page = request.args.get('page', 1, type=int)
    per_page = 5

    sort = request.args.get('sort', 'aircraftid')  #default sort by Aircraft ID asc
    order = request.args.get('order', 'asc')

    total_aircrafts = list_aircrafts()

    if order == 'asc':
        total_aircrafts.sort(key=lambda x: x.get(sort, ''))
    else:
        total_aircrafts.sort(key=lambda x: x.get(sort, ''), reverse=True)

    total_pages = (len(total_aircrafts) + per_page - 1) // per_page

    start = (page - 1) * per_page
    end = start + per_page
    aircraft_list = total_aircrafts[start:end]

    return render_template('list_aircrafts.html',
                           page={'title': 'List of Aircraft', 'current_page': page, 'total_pages': total_pages},
                           session=session,
                           aircrafts=aircraft_list)
```

List_aircrafts.html

```
<!-- Ascending/Descending patterns -->
<thead>
    <tr>
        {% if session['isadmin'] == True %}
            <th>Delete</th>
        {% endif %}
        <th>
            Aircraft ID
            <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
            ICAO Code
            <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
            Registration
            <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
            Manufacturer
            <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
            Model
            <a href="{{ url_for('list_aircrafts_route', sort='model', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='model', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
            Capacity
            <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
            <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>

        {% if session['isadmin'] == True %}
            <th>Update</th>
        {% endif %}
    </tr>
</thead>
```

Implementing search functionality



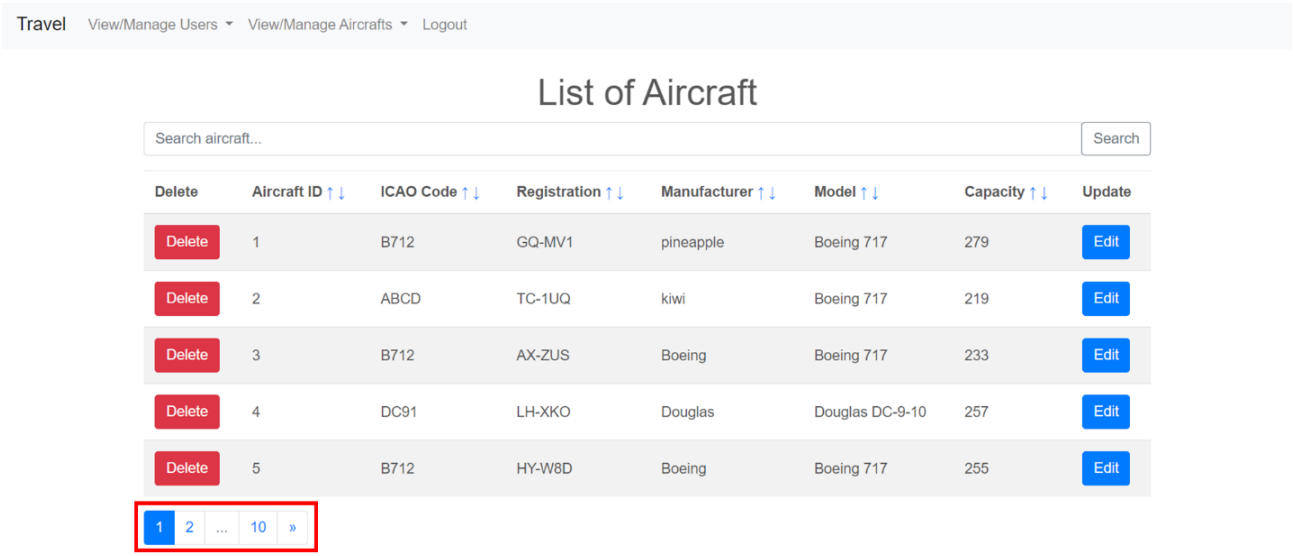
Routes.py

```
'''
Search for an aircraft by all fields
Return
    - Filtered aircrafts
'''
@app.route('/search_aircraft', methods=['GET'])
def search_aircraft_route():
    query = request.args.get('query', '').strip().lower()

    aircrafts = list_aircrafts()

    filtered_aircrafts = [
        aircraft for aircraft in aircrafts if (
            query in str(aircraft['aircraftid']).lower() or
            query in aircraft['icaocode'].lower() or
            query in aircraft['aircraftregistration'].lower() or
            query in aircraft['manufacturer'].lower() or
            query in aircraft['model'].lower() or
            query in str(aircraft['capacity']).lower()
        )
    ]

    return render_template('list_aircrafts.html',
                           page={'title': 'Search Results', 'current_page': 1, 'total_pages': 1},
                           session=session,
                           aircrafts=filtered_aircrafts)
```

List_aircrafts.html

```
<!-- Search Bar -->
<div class="mb-3">
    <form action="{{ url_for('search_aircraft_route') }}" method="GET">
        <div class="input-group">
            <input type="text" class="form-control" name="query" placeholder="Search aircraft..." aria-label="Search aircraft">
            <button class="btn btn-outline-secondary" type="submit">Search</button>
        </div>
    </form>
</div>
```

Adding limit to number of aircrafts per page when viewing all



Routes.py

```
'''
List all aircraft currently in the database
Return
    - a list of aircraft with sorting options for fields
'''
@app.route('/list_aircrafts')
def list_aircrafts_route():
    page = request.args.get('page', 1, type=int)
    per_page = 5

    sort = request.args.get('sort', 'aircraftid')  #default sort by Aircraft ID asc
    order = request.args.get('order', 'asc')

    total_aircrafts = list_aircrafts()

    if order == 'asc':
        total_aircrafts.sort(key=lambda x: x.get(sort, ''))
    else:
        total_aircrafts.sort(key=lambda x: x.get(sort, ''), reverse=True)

    total_pages = (len(total_aircrafts) + per_page - 1) // per_page

    start = (page - 1) * per_page
    end = start + per_page
    aircraft_list = total_aircrafts[start:end]

    return render_template('list_aircrafts.html',
                           page={'title': 'List of Aircraft', 'current_page': page, 'total_pages': total_pages},
                           session=session,
                           aircrafts=aircraft_list)
```

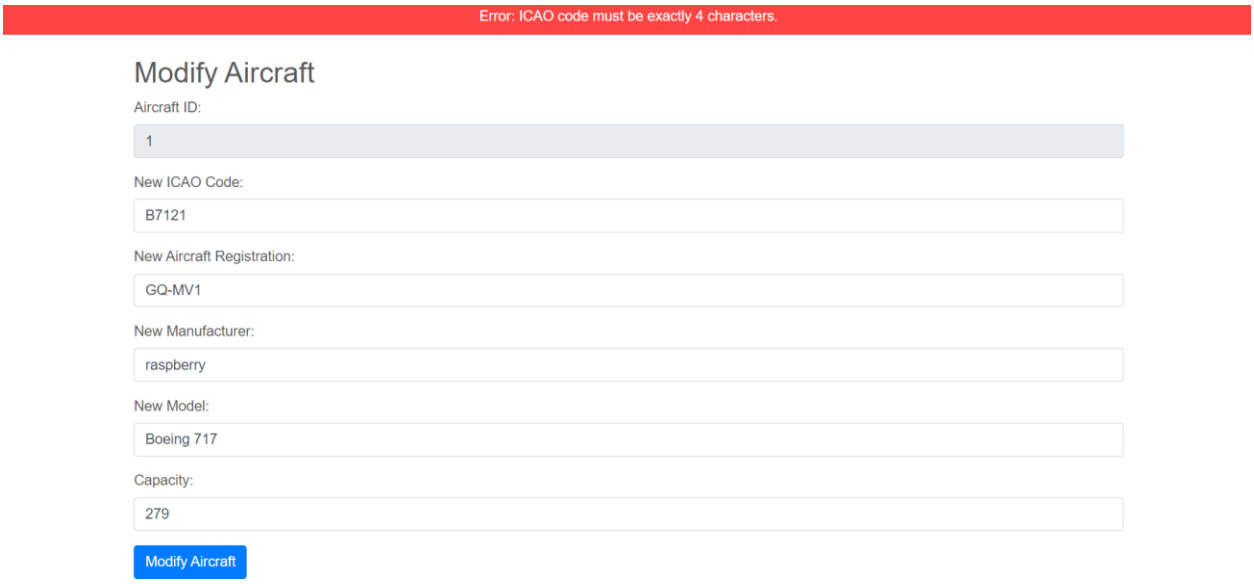
List_aircrafts.html

```
<nav aria-label="Aircraft pagination">
    <ul class="pagination">
        <!-- Previous Button -->
        {% if page.current_page > 1 %}
            <li class="page-item">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page - 1) }}" aria-label="Previous">
                    <span aria-hidden="true">&laquo;</span>
                </a>
            </li>
        {% endif %}

        <!-- First Page -->
        <li class="page-item {% if page.current_page == 1 %}active{% endif %}">
            <a class="page-link" href="{{ url_for('list_aircrafts_route', page=1) }}">1</a>
        </li>

        <!-- Ellipsis after Page 1 -->
        {% if page.current_page > 3 %}
            <li class="page-item disabled">
                <span class="page-link">...</span>
            </li>
        {% endif %}

        <!-- Middle Pages  -->
        {% set start_page = 2 if page.current_page - 1 < 2 else page.current_page - 1 %}
        {% set end_page = page.total_pages if page.current_page + 1 >= page.total_pages else page.current_page + 1 %}

        {% for p in range(start_page, end_page + 1) %}
            <li class="page-item {% if page.current_page == p %}active{% endif %}">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=p) }}">{{ p }}</a>
            </li>
        {% endfor %}

        <!-- Ellipsis before Last Page -->
        {% if page.current_page < page.total_pages - 2 %}
            <li class="page-item disabled">
                <span class="page-link">...</span>
            </li>
        {% endif %}
```

```
        <!-- Last Page -->
        {% if page.total_pages > 1 %}
            <li class="page-item {% if page.current_page == page.total_pages %}active{% endif %}">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.total_pages) }}">{{ page.total_pages }}</a>
            </li>
        {% endif %}

        <!-- Next Button -->
        {% if page.current_page < page.total_pages %}
            <li class="page-item">
                <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page + 1) }}" aria-label="Next">
                    <span aria-hidden="true">&raquo;</span>
                </a>
            </li>
        {% endif %}
    </ul>
</nav>
```

Error Messages:

Below is an example of error messages that display when an user enters something unexpected:



Routes.py

```
'''
Add a new aircraft to the database
Return
    - a form to add new aircraft and handles submissions
'''
@app.route('/add_aircraft', methods=['GET', 'POST'])
def add_aircraft_route():
    if request.method == 'POST':
        print("Form data received:", request.form)

        aircraft_id = request.form.get('AircraftID', '').strip()
        icao_code = request.form.get('icao_code', '').strip()
        aircraft_registration = request.form.get('aircraft_registration', '').strip()
        capacity = request.form.get('capacity', '').strip()

        #check id is integer
        try:
            aircraft_id_value = int(aircraft_id)
        except ValueError:
            flash("Error: Aircraft ID must be an integer.", "error")
            return redirect(url_for('add_aircraft_route'))

        #check icao code
        if icao_code and len(icao_code) != 4:
            flash("Error: ICAO code must be exactly 4 characters.", "error")
            return redirect(url_for('add_aircraft_route'))
```

Some other error messages include:

- Updated successfully
- Did not update successfully
- Formatting messages (user did not enter expected data)

My Takeaways from Flask

- Using routing to link views to specific pages using the @app.route() function. Additionally, methods defining GET and/or POST to determine what the page retrieves and/or returns (displays)
- The flash function used to display messages to inform users of successful or unsuccessful operations
- Utilising pagination to restrict the number of records shown on a page

Add_aircraft.html

```
{% include 'top.html' %}
<div class="content">
  <div class="container my-4">

    <h2 class="title"> Add Aircraft </h2>

    <form class="needs-validation" method="POST" action="{{ url_for('add_aircraft_route') }}"
novalidate>

      <div class="form-group">
        <label for="aircraft_id">Aircraft ID:</label>
        <input type="text" class="form-control" id="aircraft_id" name="aircraft_id"
placeholder="Enter aircraft ID here" required>
        <div class="invalid-feedback">Please enter an aircraft ID.</div>
      </div>

      <div class="form-group">
        <label for="icao_code">ICAO Code:</label>
        <input type="text" class="form-control" id="icao_code" name="icao_code"
placeholder="Enter ICAO code here" required>
        <div class="invalid-feedback">Please enter an ICAO code.</div>
      </div>

      <div class="form-group">
        <label for="aircraft_registration">Aircraft Registration:</label>
        <input type="text" class="form-control" id="aircraft_registration"
name="aircraft_registration" placeholder="Enter aircraft registration here" required>
        <div class="invalid-feedback">Please enter an aircraft registration.</div>
      </div>

      <div class="form-group">
        <label for="manufacturer">Manufacturer:</label>
        <input type="text" class="form-control" id="manufacturer" name="manufacturer"
placeholder="Enter manufacturer here" required>
        <div class="invalid-feedback">Please enter a manufacturer.</div>
      </div>

      <div class="form-group">
        <label for="model">Model:</label>
```

```
          <input type="text" class="form-control" id="model" name="model" placeholder="Enter model here" required>
          <div class="invalid-feedback">Please enter a model.</div>
        </div>

        <div class="form-group">
          <label for="name">Capacity:</label>
          <input type="text" class="form-control" id="capacity" name="capacity" placeholder="Enter aircraft capacity here" required>
          <div class="invalid-feedback">Please enter the aircraft capacity.</div>
        </div>

        <button class="btn btn-primary" type="submit">Add Aircraft</button>

      </form>

    </div>
</div>
{% include 'end.html' %}
```

Count_aircraft.html

```
{% include 'top.html' %}


<div class="container my-4">
  <h2 class="page-title">Aircraft Count by Manufacturer</h2>


  <table class="table table-bordered">
    <thead class="thead-light">
      <tr>
        <th>Manufacturer</th>
        <th>Number of Aircraft</th>
      </tr>
    </thead>
    <tbody>
      {% for manufacturer, count in counts %}
      <tr>
```

```
            <td>{{ manufacturer }}</td>
            <td>{{ count }}</td>
          </tr>
          {% endfor %}
        </tbody>
    </table>
</div>

{% include 'end.html' %}
```

List_aircrafts.html

```
{% include 'top.html' %}

<div id="content" class="container my-4">
    <h1 class="page-title">{{page.get('title', 'Aircraft List')}}</h1>

    <!-- Search Bar -->
    <div class="mb-3">
        <form action="{{ url_for('search_aircraft_route') }}" method="GET">
            <div class="input-group">
                <input type="text" class="form-control" name="query" placeholder="Search aircraft..."
aria-label="Search aircraft">
                <button class="btn btn-outline-secondary" type="submit">Search</button>
            </div>
        </form>
    </div>

    <!-- Table -->
    <table class="table table-striped">
        <!-- Ascending/Descending patterns -->
```

```
    <thead>
      <tr>
        {% if session['isadmin'] == True %}
          <th>Delete</th>
        {% endif %}
        <th>
          Aircraft ID
          <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
          <a href="{{ url_for('list_aircrafts_route', sort='aircraftid', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
          ICAO Code
          <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
          <a href="{{ url_for('list_aircrafts_route', sort='icaocode', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
          Registration
          <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
          <a href="{{ url_for('list_aircrafts_route', sort='aircraftregistration', order='desc') }}" aria-label="Sort descending">&#8595;</a>
        </th>
        <th>
          Manufacturer
          <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
          <a href="{{ url_for('list_aircrafts_route', sort='manufacturer', order='desc') }}" aria-label="Sort descending">&#8595;</a>
```

```
            </th>
            <th>
              Model
              <a href="{{ url_for('list_aircrafts_route', sort='model', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
              <a href="{{ url_for('list_aircrafts_route', sort='model', order='desc') }}" aria-label="Sort descending">&#8595;</a>
            </th>
            <th>
              Capacity
              <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='asc') }}" aria-label="Sort ascending">&#8593;</a>
              <a href="{{ url_for('list_aircrafts_route', sort='capacity', order='desc') }}" aria-label="Sort descending">&#8595;</a>
            </th>

            {% if session['isadmin'] == True %}
              <th>Update</th>
            {% endif %}
          </tr>
        </thead>

        <!-- List Aircrafts -->
        <tbody>
        {% for aircraft in aircrafts %}
          <tr class="align-items-center">
            {% if session['isadmin'] == True %}
              <!-- Delete button for admin users only -->
              <td>
                <form action="{{ url_for('remove_aircraft_route') }}" method="POST" style="display:inline;">
```

```
                <input type="hidden" name="aircraft_id" value="{{ aircraft['aircraftid'] }}">
                <button type="submit" class="btn btn-danger">Delete</button>
              </form>
            </td>
            <td class="align-middle">{{ aircraft['aircraftid'] }}</td>
            <td class="align-middle">{{ aircraft['icaocode'] }}</td>
            <td class="align-middle">{{ aircraft['aircraftregistration'] }}</td>
            <td class="align-middle">{{ aircraft['manufacturer'] }}</td>
            <td class="align-middle">{{ aircraft['model'] }}</td>
            <td class="align-middle">{{ aircraft['capacity'] }}</td>

            <td><a class="btn btn-primary" href="{{ url_for('modify_aircraft_route',
aircraft_id=aircraft['aircraftid']) }}">Edit</a></td>
          {% else %}
            <td class="align-middle">{{ aircraft['aircraftid'] }}</td>
            <td class="align-middle">{{ aircraft['icaocode'] }}</td>
            <td class="align-middle">{{ aircraft['aircraftregistration'] }}</td>
            <td class="align-middle">{{ aircraft['manufacturer'] }}</td>
            <td class="align-middle">{{ aircraft['model'] }}</td>
            <td class="align-middle">{{ aircraft['capacity'] }}</td>

          {% endif %}
        </tr>
      {% endfor %}
      </tbody>
    </table>

    <nav aria-label="Aircraft pagination">
      <ul class="pagination">
        <!-- Previous Button -->
```

```
        {% if page.current_page > 1 %}
          <li class="page-item">
            <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page - 1) }}" aria-label="Previous">
              <span aria-hidden="true">&laquo;</span>
            </a>
          </li>
        {% endif %}

        <!-- First Page -->
        <li class="page-item {% if page.current_page == 1 %}active{% endif %}">
          <a class="page-link" href="{{ url_for('list_aircrafts_route', page=1) }}">1</a>
        </li>

        <!-- Ellipsis after Page 1 -->
        {% if page.current_page > 3 %}
          <li class="page-item disabled">
            <span class="page-link">...</span>
          </li>
        {% endif %}

        <!-- Middle Pages  -->
        {% set start_page = 2 if page.current_page - 1 < 2 else page.current_page - 1 %}
        {% set end_page = page.total_pages if page.current_page + 1 >= page.total_pages else page.current_page + 1 %}

        {% for p in range(start_page, end_page + 1) %}
          <li class="page-item {% if page.current_page == p %}active{% endif %}">
            <a class="page-link" href="{{ url_for('list_aircrafts_route', page=p) }}">{{ p }}</a>
          </li>
```

```
        {% endfor %}

        <!-- Ellipsis before Last Page -->
        {% if page.current_page < page.total_pages - 2 %}
          <li class="page-item disabled">
            <span class="page-link">...</span>
          </li>
        {% endif %}

        <!-- Last Page -->
        {% if page.total_pages > 1 %}
          <li class="page-item {% if page.current_page == page.total_pages %}active{% endif %}">
            <a class="page-link" href="{{ url_for('list_aircrafts_route',
page=page.total_pages) }}">{{ page.total_pages }}</a>
          </li>
        {% endif %}

        <!-- Next Button -->
        {% if page.current_page < page.total_pages %}
          <li class="page-item">
            <a class="page-link" href="{{ url_for('list_aircrafts_route', page=page.current_page + 1) }}"
aria-label="Next">
              <span aria-hidden="true">&raquo;</span>
            </a>
          </li>
        {% endif %}
      </ul>
    </nav>
```

```
</div>

{% include 'end.html' %}
```

List_manufacturers.html

```
{% include 'top.html' %}

<div class="container my-4">
  <h2 class="page-title">Manufacturers</h2>

  <ul class="list-group">
    {% for manufacturer in manufacturers %}
    <li class="list-group-item">{{ manufacturer }}</li>
    {% endfor %}
  </ul>
</div>

{% include 'end.html' %}
```

Modify_aircraft.html

```
{% include 'top.html' %}

  <div class="container my-4">
    <h2 class="title"> Modify Aircraft </h2>

    <form class="needs-validation" method="POST" action="{{ url_for('modify_aircraft_route', aircraft_id=aircraft['AircraftID']) }}" novalidate>

      <div class="form-group">
        <label for="aircraft_id">Aircraft ID:</label>
```

```
        <input type="text" class="form-control" id="aircraft_id" name="AircraftID"
value="{{aircraft.get('AircraftID', '')}}" readonly required>
        <div class="invalid-feedback">Please enter an aircraft ID.</div>
    </div>

    <div class="form-group">
        <label for="icao_code">New ICAO Code:</label>
        <input type="text" class="form-control" id="icao_code" name="icao_code"
value="{{aircraft.get('ICAOCode', '')}}" placeholder="Enter new ICAO code">
        <div class="invalid-feedback">Please enter a valid ICAO code.</div>
    </div>

    <div class="form-group">
        <label for="aircraft_registration">New Aircraft Registration:</label>
        <input type="text" class="form-control" id="aircraft_registration"
name="aircraft_registration" value="{{aircraft.get('AircraftRegistration', '')}}" placeholder="Enter
new aircraft registration">
        <div class="invalid-feedback">Please enter a valid aircraft registration.</div>
    </div>

    <div class="form-group">
        <label for="manufacturer">New Manufacturer:</label>
        <input type="text" class="form-control" id="manufacturer" name="manufacturer"
value="{{aircraft.get('Manufacturer', '')}}" placeholder="Enter new manufacturer">
        <div class="invalid-feedback">Please enter a valid manufacturer.</div>
    </div>

    <div class="form-group">
        <label for="model">New Model:</label>
```

```
            <input type="text" class="form-control" id="model" name="model"
value="{{aircraft.get('Model', '')}}" placeholder="Enter new model">
            <div class="invalid-feedback">Please enter a valid model.</div>
        </div>

        <div class="form-group">
            <label for="capacity">Capacity:</label>
            <input type="number" class="form-control" id="capacity" name="capacity"
value="{{aircraft.get('Capacity', '')}}" placeholder="Enter new capacity">
            <div class="invalid-feedback">Please enter a valid capacity.</div>
        </div>
        <button class="btn btn-primary" type="submit">Modify Aircraft</button>
    </form>
  </div>
{% include 'end.html' %}
```

ONLY Changes to routes.py

```
'''
Extract relevant data from the form (page) for aircrafts
Returns
  - a dictionary of updated values
  - boolean indicating if any updates were made
'''
def extract_aircraft_data(form):
  update_data = {
    'AircraftID': form.get('AircraftID'),
    'ICAOCode': form.get('icao_code'),
    'AircraftRegistration': form.get('aircraft_registration'),
    'Manufacturer': form.get('manufacturer'),
```

```
        'Model': form.get('model'),
        'Capacity': form.get('capacity')
    }
    valid_update = any(value for value in update_data.values() if value)
    return update_data, valid_update

def extract_aircraft_data_add(form):
    update_data = {
        'AircraftID': form.get('aircraft_id'),
        'ICAOCode': form.get('icao_code'),
        'AircraftRegistration': form.get('aircraft_registration'),
        'Manufacturer': form.get('manufacturer'),
        'Model': form.get('model'),
        'Capacity': form.get('capacity')
    }
    valid_update = any(value for value in update_data.values() if value)
    return update_data, valid_update

'''
Delete an aircraft from the database
Return
    - redirects back to the aircraft list with message after deletion
'''
@app.route('/aircrafts/delete/<aircraft_id>', methods=['POST'])
def delete_aircraft(aircraft_id):
    if 'logged_in' not in session or not session['logged_in']:
        return redirect(url_for('login'))

    remove_aircraft(aircraft_id)
```

```
    flash(f'Aircraft {aircraft_id} has been successfully deleted.')
    return redirect(url_for('list_aircrafts'))

'''
Add a new aircraft to the database
Return
    - a form to add new aircraft and handles submissions
'''
@app.route('/add_aircraft', methods=['GET', 'POST'])
def add_aircraft_route():
    if request.method == 'POST':
        print("Form data received:", request.form)

        aircraft_data, _ = extract_aircraft_data_add(request.form)

        add_aircraft(
            aircraft_data['AircraftID'],
            aircraft_data.get('ICAOCode'),
            aircraft_data.get('AircraftRegistration'),
            aircraft_data.get('Manufacturer'),
            aircraft_data.get('Model'),
            aircraft_data.get('Capacity')
        )

        return redirect(url_for('add_aircraft_route'))

    return render_template('add_aircraft.html', page={'title': 'Add Aircraft'})
```

```
'''
Edit an existing aircraft's details
Return
  - a form for updating fields
'''
@app.route('/modify_aircraft', methods=['GET', 'POST'])
@app.route('/aircrafts/edit/<aircraft_id>', methods=['GET', 'POST'])
def modify_aircraft_route(aircraft_id=None):
  if 'logged_in' not in session or not session['logged_in']:
    return redirect(url_for('login'))

  page_title = 'Modify Aircraft'
  aircraft_details = {}

  if aircraft_id:
    page_title = 'Edit Aircraft Details'
    aircraft_details = get_aircraft_by_id(aircraft_id)
    if aircraft_details is None:
      flash(f'Error: No aircraft found with ID {aircraft_id}')
      return redirect(url_for('list_aircrafts'))

  icao_code = request.form.get('icao_code', '').strip()
  if icao_code and len(icao_code) != 4:
    flash("Error: ICAO code must be exactly 4 characters.", "error")
    return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))

  if request.method == 'POST':
    if 'AircraftID' not in request.form:
      flash('Error: Aircraft ID is required.')
```

```
            return redirect(url_for('list_aircrafts'))

        icao_code = request.form.get('icao_code', '').strip()
        if icao_code and len(icao_code) != 4:
            flash("Error: ICAO code must be exactly 4 characters.", "error")
            return redirect(url_for('modify_aircraft_route', aircraft_id=aircraft_id))

        update_data, valid_update = extract_aircraft_data(request.form)

        if valid_update:
            modify_aircraft(
                update_data['AircraftID'],
                update_data.get('ICAOCode'),
                update_data.get('AircraftRegistration'),
                update_data.get('Manufacturer'),
                update_data.get('Model'),
                update_data.get('Capacity')
            )
            flash(f'Aircraft {update_data["AircraftID"]} updated successfully.')
            return redirect(url_for('list_aircrafts_route'))
        else:
            flash("No updated values for aircraft.")
            return redirect(url_for('modify_aircraft', aircraft_id=aircraft_id))

    current_page = 'modify_aircraft'
    return render_template(
        'modify_aircraft.html',
        session=session,
        page={'title': page_title, 'current_page': current_page},
```

```
        aircraft=aircraft_details
    )


'''
Remove an aircraft from the database
'''
@app.route('/remove_aircraft/<aircraft_id>', methods=['GET', 'POST'])
@app.route('/remove_aircraft', methods=['GET', 'POST'])
def remove_aircraft_route():
    if request.method == 'POST':
        aircraft_id = request.form['aircraft_id']
        remove_aircraft(aircraft_id)
        flash(f'Aircraft {aircraft_id} has been successfully deleted.')
        return redirect(url_for('list_aircrafts_route'))
    return render_template('remove_aircraft.html', page={'title': 'Remove Aircraft'})


'''
List all unique aircraft manufacturers from the database
Return
    - a page showing the distinct manufacturers
'''
@app.route('/list_manufacturers')
def list_manufacturers_route():
    manufacturers = get_unique_manufacturers()
    return render_template('list_manufacturers.html', manufacturers=manufacturers, page={'title':
'List Manufacturers'})


'''
```

```
Count the number of aircrafts by each manufacturer
Return
  - a list of manufacturers and the number of aircraft each one has
'''
@app.route('/count_aircraft')
def count_aircraft_route():
  counts = get_aircraft_count_by_manufacturer()
  return render_template('count_aircraft.html', counts=counts, page={'title': 'Count Aircraft by Manufacturer'})

'''
List all aircraft currently in the database
Return
  - a list of aircraft with sorting options for fields
'''
@app.route('/list_aircrafts')
def list_aircrafts_route():
  page = request.args.get('page', 1, type=int)
  per_page = 5

  sort = request.args.get('sort', 'aircraftid')  #default sort by Aircraft ID asc
  order = request.args.get('order', 'asc')

  total_aircrafts = list_aircrafts()

  if order == 'asc':
    total_aircrafts.sort(key=lambda x: x.get(sort, ''))
  else:
    total_aircrafts.sort(key=lambda x: x.get(sort, ''), reverse=True)
```

```
    total_pages = (len(total_aircrafts) + per_page - 1) // per_page

    start = (page - 1) * per_page
    end = start + per_page
    aircraft_list = total_aircrafts[start:end]

    return render_template('list_aircrafts.html',
                           page={'title': 'List of Aircraft', 'current_page': page, 'total_pages': total_pages},
                           session=session,
                           aircrafts=aircraft_list)

'''
Search for an aircraft by all fields
Return
    - Filtered aircrafts
'''
@app.route('/search_aircraft', methods=['GET'])
def search_aircraft_route():
    query = request.args.get('query', '').strip().lower()

    aircrafts = list_aircrafts()

    filtered_aircrafts = [
        aircraft for aircraft in aircrafts if (
            query in str(aircraft['aircraftid']).lower() or
            query in aircraft['icaocode'].lower() or
            query in aircraft['aircraftregistration'].lower() or
            query in aircraft['manufacturer'].lower() or
            query in aircraft['model'].lower() or
```

```
        query in str(aircraft['capacity']).lower()
    )
  ]

  return render_template('list_aircrafts.html',
                page={'title': 'Search Results', 'current_page': 1, 'total_pages': 1},
                session=session,
                aircrafts=filtered_aircrafts)



'''
Display the details of a single aircraft
Return
  - details of a specific aircraft by ID
'''
@app.route('/aircraft/<aircraft_id>')
def list_single_aircraft(aircraft_id):
  aircraft_details = get_aircraft_by_id(aircraft_id)
  if not aircraft_details:
    flash(f'Error: No aircraft found with ID {aircraft_id}')
    return redirect(url_for('list_aircrafts'))

  return render_template('list_aircrafts.html', aircraft=aircraft_details, page={'title': f'Aircraft
{aircraft_id} Details'})
```

aircrafts.py

```
from database import database_connect, dictfetchall
from flask import *

import pg8000
```

```
import configparser
import sys


'''
Adds aircraft to the table
'''
def add_aircraft(aircraft_id, icao_code, aircraft_registration, manufacturer, model, capacity):
  if aircraft_exists(aircraft_id):
    print(f"Error: fail to add aircraft. {aircraft_id} already exists")
    return
  try:
    conn = database_connect()
    cursor = conn.cursor()


    cursor.execute(
      """
      INSERT INTO Aircraft (AircraftID, ICAOCode, AircraftRegistration, Manufacturer, Model,
Capacity)
      VALUES (%s, %s, %s, %s, %s, %s)
      """,
      (aircraft_id, icao_code, aircraft_registration, manufacturer, model, capacity)
    )


    conn.commit()
    print("Aircraft added successfully.")


  except Exception as e:
    print(f"Error: fail to add aircraft {aircraft_id} + {e}")
    conn.rollback()
  finally:
```

```
        cursor.close()
        conn.close()

'''
Remove an aircraft by id
'''
def remove_aircraft(aircraft_id):
    if not aircraft_exists(aircraft_id):
        print(f"Error: fail to remove aircraft. {aircraft_id} does not exist")
        return
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "DELETE FROM Aircraft WHERE AircraftID = %s",
            (aircraft_id,)
        )

        conn.commit()
        print("Aircraft removed successfully.")

    except Exception as e:
        print(f"Error: fail to remove aircraft {aircraft_id}. Exception: {e}")
        conn.rollback()
    finally:
        cursor.close()
        conn.close()
```

```
'''
Edits an existing aircraft with a given id
'''
def modify_aircraft(aircraft_id, icao_code=None, aircraft_registration=None, manufacturer=None, model=None, capacity=None):
    print("modifying aircraft")
    if not aircraft_exists(aircraft_id):
        print(f"Error: fail to modify aircraft. {aircraft_id} does not exist")
        return

    try:
        conn = database_connect()
        cursor = conn.cursor()

        update_fields = []
        update_values = []

        if icao_code and icao_code.strip():
            update_fields.append("ICAOCode = %s")
            update_values.append(icao_code)

        if aircraft_registration and aircraft_registration.strip():
            update_fields.append("AircraftRegistration = %s")
            update_values.append(aircraft_registration)

        if manufacturer and manufacturer.strip():
            update_fields.append("Manufacturer = %s")
            update_values.append(manufacturer)

        if model and model.strip():
```

```
        update_fields.append("Model = %s")
        update_values.append(model)

    if capacity and capacity.strip():
        update_fields.append("Capacity = %s")
        update_values.append(capacity)

    if not update_fields:
        print("Error: No fields provided to update.")
        return

    update_values.append(aircraft_id)

    sql = f"UPDATE Aircraft SET {', '.join(update_fields)} WHERE AircraftID = %s"

    cursor.execute(sql, tuple(update_values))

    conn.commit()
    print("Aircraft updated successfully.")

except:
    print(f"Error: Failed to update aircraft due to {aircraft_id}")
    conn.rollback()

finally:
    cursor.close()
    conn.close()
```

```
'''
Gets a list of manufacturers
'''
def get_unique_manufacturers():
    manufacturers = []
    conn = None
    cursor = None

    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT DISTINCT Manufacturer FROM Aircraft"
        )

        manufacturers = cursor.fetchall()
        manufacturers = [manufacturer[0] for manufacturer in manufacturers]

    except Exception as e:
        manufacturers = []
        print(f"Error: fail to get manufacturers: {e}")
    finally:
        cursor.close()
        conn.close()

    return manufacturers

'''
```

```
Get number of aircraft manufactured by each manufacturer
'''
def get_aircraft_count_by_manufacturer():
    counts = []
    conn = None
    cursor = None

    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            """
            SELECT Manufacturer, COUNT(*) AS AircraftCount
            FROM Aircraft
            GROUP BY Manufacturer
            ORDER BY AircraftCount DESC
            """
        )

        results = cursor.fetchall()

        if results:
            for manufacturer, count in results:
                counts.append((manufacturer, count))

    except Exception as e:
        print(f"Error: failed to get aircraft counts: {e}")
        counts = []
```

```
    finally:
        cursor.close()
        conn.close()

    return counts

'''
lists all aircrafts in database
'''
def list_aircrafts():
    conn = database_connect()
    if conn is None:
        return None

    cur = conn.cursor()
    returndict = None

    try:
        sql = """SELECT *
                 FROM Aircraft"""

        returndict = dictfetchall(cur, sql)

    except:
        import traceback
        traceback.print_exc()
        print("Error: fail to get aircrafts from database", sys.exc_info()[0])
```

```
    cur.close()
    conn.close()

    return returndict

'''
Gets aircraft details by aircraft ID
'''
def get_aircraft_by_id(aircraft_id):
    aircraft = None
    try:
        conn = database_connect()
        cursor = conn.cursor()

        cursor.execute(
            "SELECT * FROM Aircraft WHERE AircraftID = %s",
            (aircraft_id,)
        )

        aircraft = cursor.fetchone()

        if aircraft:
            aircraft = {
                "AircraftID": aircraft[0],
                "ICAOCode": aircraft[1],
                "AircraftRegistration": aircraft[2],
                "Manufacturer": aircraft[3],
                "Model": aircraft[4],
                "Capacity": aircraft[5]
```

```
        }
      else:
        print(f"Error: Aircraft with ID {aircraft_id} does not exist.")

    except Exception as e:
      print(f"Error: failed to get aircraft by ID {e}")
    finally:
      cursor.close()
      conn.close()

    print("AIRCRAFT WITH ID: ", aircraft)
    return aircraft

'''
checks if an aircraft ID exists in the table
'''
def aircraft_exists(aircraft_id):
    try:
      conn = database_connect()
      cursor = conn.cursor()

      cursor.execute(
        "SELECT COUNT(*) FROM Aircraft WHERE AircraftID = %s",
        (aircraft_id,)
      )

      count = cursor.fetchone()[0]

      if count > 0:
```

```
            return True
        else:
            return False

    except Exception as e:
        print(f"Error: fail to check aircraft ID {e}")
        return False
    finally:
        cursor.close()
        conn.close()
```

top.html

```
<!DOCTYPE html>
<html>

<head>
  <!-- this goes to the 'static' folder and grabs our CSS -->
  <!-- [Brief Intro:] CSS is how we make the websites look nicer -->
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/bootstrap.css') }}">
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/main.css') }}">
  <script type="application/javascript">

    function getsearchtarget() {

      // Get the select list and store in a variable
      var e = document.getElementById("searchtarget");

      // Get the selected value of the select list
      var formaction = e.options[e.selectedIndex].value;
```

```
    // Update the form action
    document.searchform.action = formaction;

  }
 </script>
 <title>{{ page.title }}</title>
</head>

<body>
 <!-- Generating the menu and what happens when the user is logged in VS logged out -->
 <nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="/">Travel</a>
  <div class="collapse navbar-collapse" id="navbarNav">
   <ul class="navbar-nav">

    <!-- Dropdown for managing/viewing users -->
    {% if session.logged_in %}
    <li class="nav-item dropdown">
     <a class="nav-link dropdown-toggle" href="#" id="usersDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      View/Manage Users
     </a>
     <div class="dropdown-menu" aria-labelledby="usersDropdown">
      <a class="dropdown-item" href="{{ url_for('list_users') }}">List Users</a>
      <a class="dropdown-item" href="{{ url_for('search_users_byname') }}">User Search</a>
      {% if session['isadmin'] == True %}
      <a class="dropdown-item" href="{{ url_for('add_user') }}">Add User</a>
      <div class="dropdown-divider"></div>
      <a class="dropdown-item" href="{{ url_for('list_user_stats') }}">User Stats</a>
```

```
        <a class="dropdown-item" href="{{ url_for('list_consolidated_users') }}">User Details (Advanced)</a>
        {% endif %}
      </div>
    </li>

    <!-- Dropdown for managing/viewing aircraft -->
    {% if session.logged_in %}
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="aircraftDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
        View Aircrafts
      </a>
      <div class="dropdown-menu" aria-labelledby="aircraftDropdown">
        <a class="dropdown-item" href="{{ url_for('list_aircrafts_route') }}">List Aircrafts</a>
        <a class="dropdown-item" href="{{ url_for('list_manufacturers_route') }}">Count Manufacturers</a>
        <a class="dropdown-item" href="{{ url_for('count_aircraft_route') }}">Count Aircrafts</a>

        {% if session['isadmin'] == True %}
        <div class="dropdown-divider"></div>
        <a class="dropdown-item" href="{{ url_for('add_aircraft_route') }}">Add Aircraft</a>
        {% endif %}
      </div>
    </li>
    {% endif %}

    <!-- Logout -->
    <li class="nav-item">
      <a class="nav-link" href="{{ url_for('logout') }}">Logout</a>
```

```
    </li>
    {% endif %}
   </ul>
  </div>
</nav>
<!--
        This is for our flashed messages
        Whenever we use flash('message in here')
        it will come out inside this list
    -->
{% with messages = get_flashed_messages() %}
{% if messages %}
<ul class="flashes">
 {% for message in messages %}
 <li>{{ message }}</li>
 {% endfor %}
</ul>
{% endif %}
{% endwith %}
```