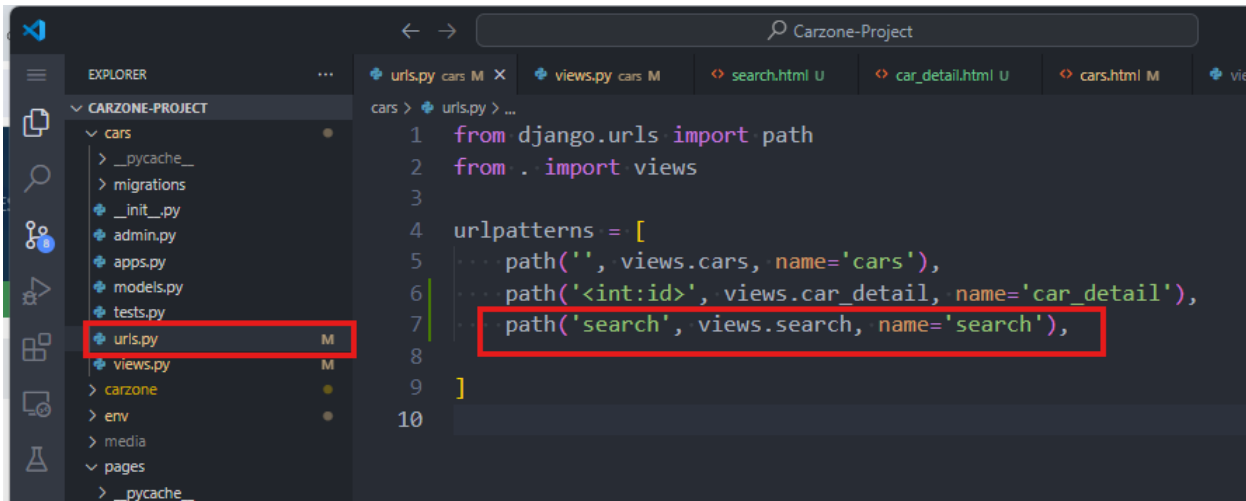


Topic: Car Listing: Search Functionality Part 9

Speaker: Udemy Instructor Rathan Kumar | Notebook: Django Project: Car Listing



1. Create a SEARCH functionality. Go to CARS\URLS.PY and create a new path.

A screenshot of a code editor showing the Django URL patterns in the 'urls.py' file. The file is located in the 'cars' directory. The code defines a list of URL patterns, including a search path. The search path is highlighted with a red box. The Explorer panel on the left shows the project structure, with 'urls.py' highlighted in red. The code editor shows the following code:

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('', views.cars, name='cars'),
6     path('<int:id>', views.car_detail, name='car_detail'),
7     path('search', views.search, name='search'),
8
9 ]
10
```

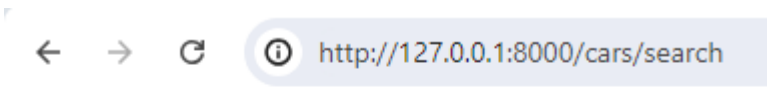
2. Create a new function in the CARS\VIEWS.PY

```
6
7
8 def cars(request):
9     cars = Car.objects.order_by('-created_date')
10
11     # Shows 3 cars on the page
12     paginator = Paginator(cars, 2)
13     #
14     page = request.GET.get('page')
15     paged_cars = paginator.get_page(page)
16
17     data = {
18         'cars': paged_cars,
19     }
20     return render(request, 'cars/cars.html', data)
21
22
23 def car_detail(request, id):
24     # finds the car record using the primary key (PK) = id
25     single_car = get_object_or_404(Car, pk=id)
26     data = {
27         'single_car': single_car,
28     }
29
30     return render(request, 'cars/car_detail.html', data)
31
32 def search(request):
33     return render(request, 'cars/search.html')
```

3. Create a new HTML file, TEMPLATES\CARS\SEARCH.HTML and create a simple code

```
templates > cars > search.html > h1
1 <h1>Search</h1>
```

4. Test it. In the the browser, type 127.0.0.1:8000\CARS\SEARCH



Search

5. Update the HOME.HTML to create a link to SEARCH.HTML

FROM:

```

<!-- Full Page Search -->
<div id="full-page-search">
  <button type="button" class="close"></button>
  <form action="https://storage.googleapis.com/theme-vessel/carhouse/index.html#" class="search-header">
    <input type="search" value="" placeholder="type keyword(s) here. Eg: audi, benz etc" />
    <button type="submit" class="btn btn-sm button-theme">Search</button>
  </form>
</div>

```

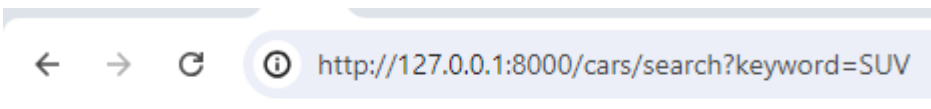
TO:

```

<!-- Full Page Search -->
<div id="full-page-search">
  <button type="button" class="close"></button>
  <form action="{% url 'search' %}" class="search-header" name="keyword" />
    <input type="search" value="" placeholder="type keyword(s) here. Eg: audi, benz etc" />
    <button type="submit" class="btn btn-sm button-theme">Search</button>
  </form>
</div>

```

So, when you search for something like 'SUV,' the search result will be like this with KEYWORD = 'SUV'



Search

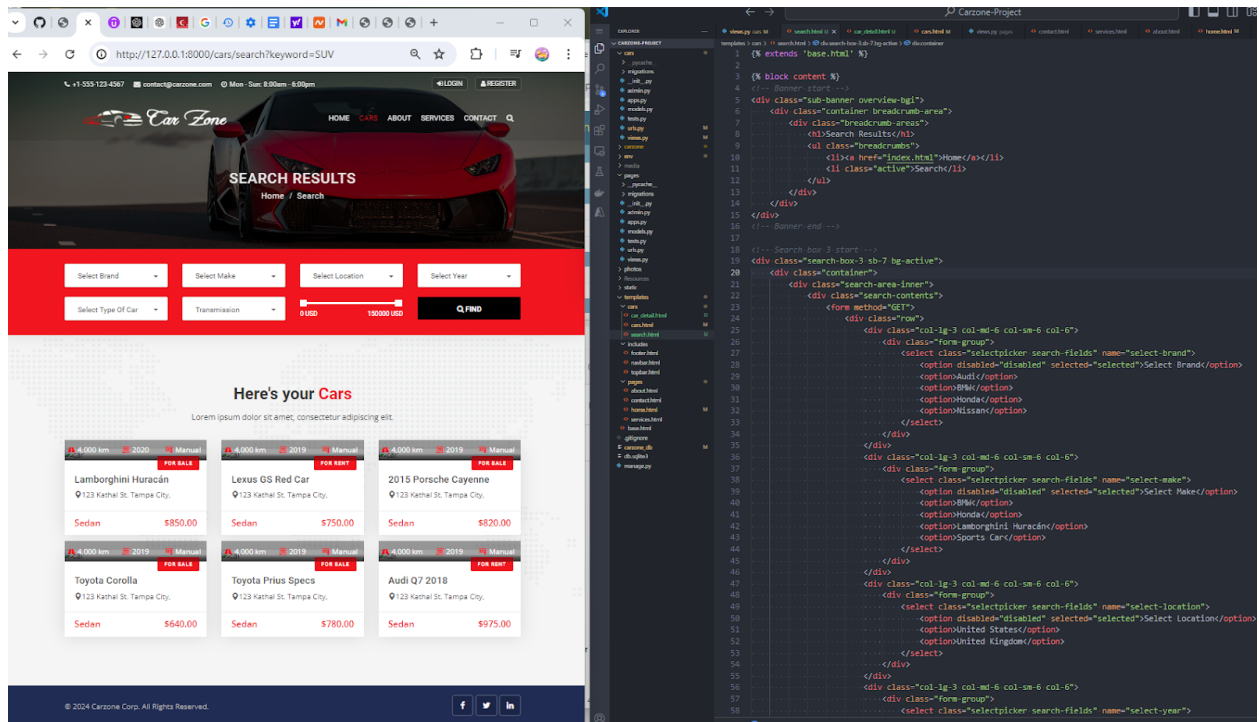
6. Add the Django tags and Copy the code of SEARCH.HTML from the RESOURCE folder.

{% extends 'base.html' %}

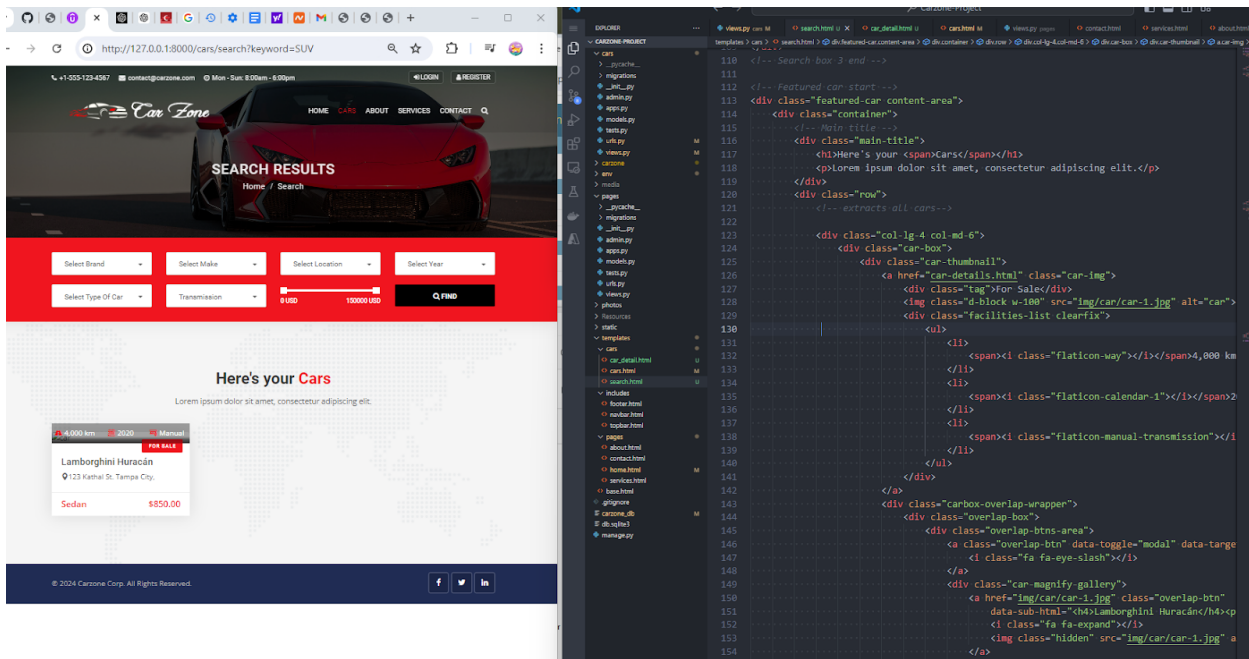
{% block content %}

<we add our OWN code here?>

{% endblock %}



7. Update SEARCH.HTML to remove duplicate blocks of car items.



8. Update the VIEWS.PY

```
def search(request):
    cars = Car.objects.order_by('-created_date')
    data = {
        'cars': cars,
    }
    return render(request, 'cars/search.html', data)
```

9. Update the SEARCH.HTML with the Django tags to call all dynamic data. This is similar to displaying your LATEST CARS (CARS.HTML)

10. Now, make the filter or search functionality work Update the VIEWS.PY as. This searches the cars using the DESCRIPTION FIELD if it contains what was being searched.

```
def search(request):
    cars = Car.objects.order_by('-created_date')

    if 'keyword' in request.GET:
        keyword = request.GET['keyword']

        # checks if keyword is not blank
        if keyword:
            # filters the cars containing keyword
            cars = cars.filter(description__icontains=keyword)

    data = {
        'cars': cars,
    }
    return render(request, 'cars/search.html', data)
```

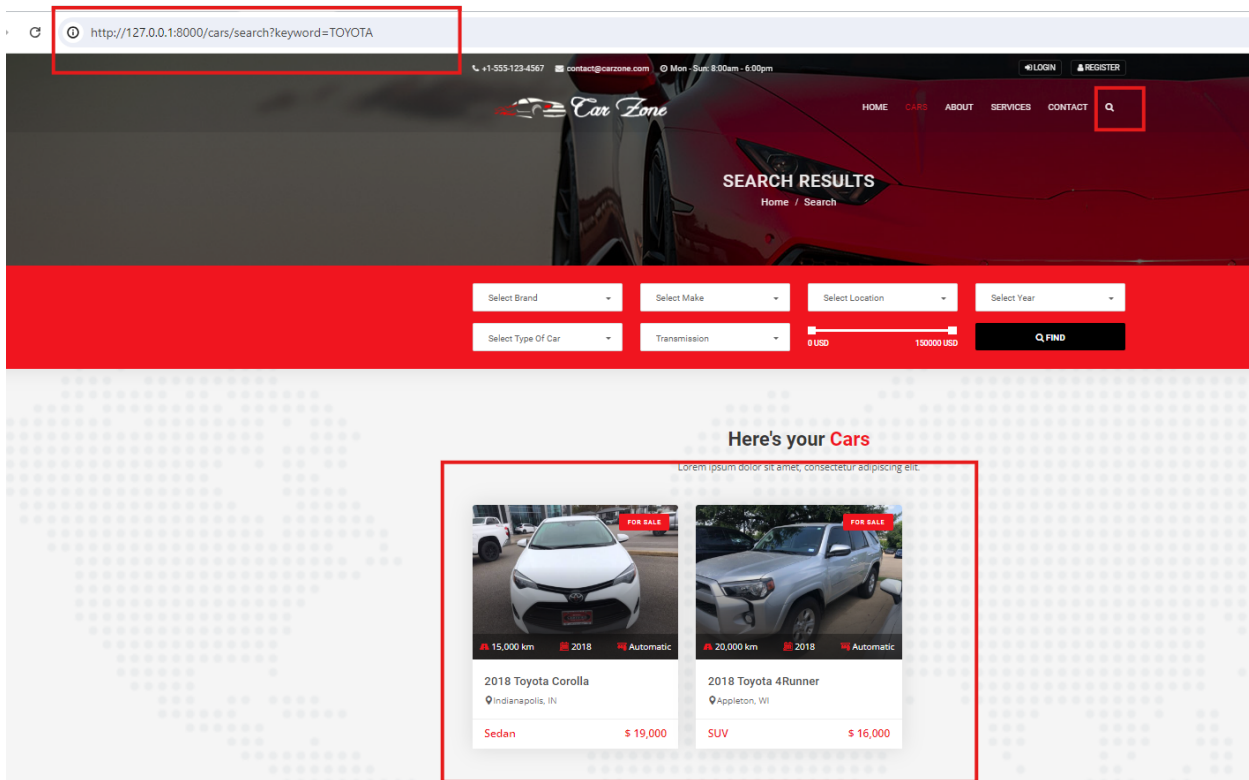
11. To make SEARCH FUNCTIONALITY work in every webpage, we move our SEARCH HTML CODE AND add this in the BASE.HTML

```

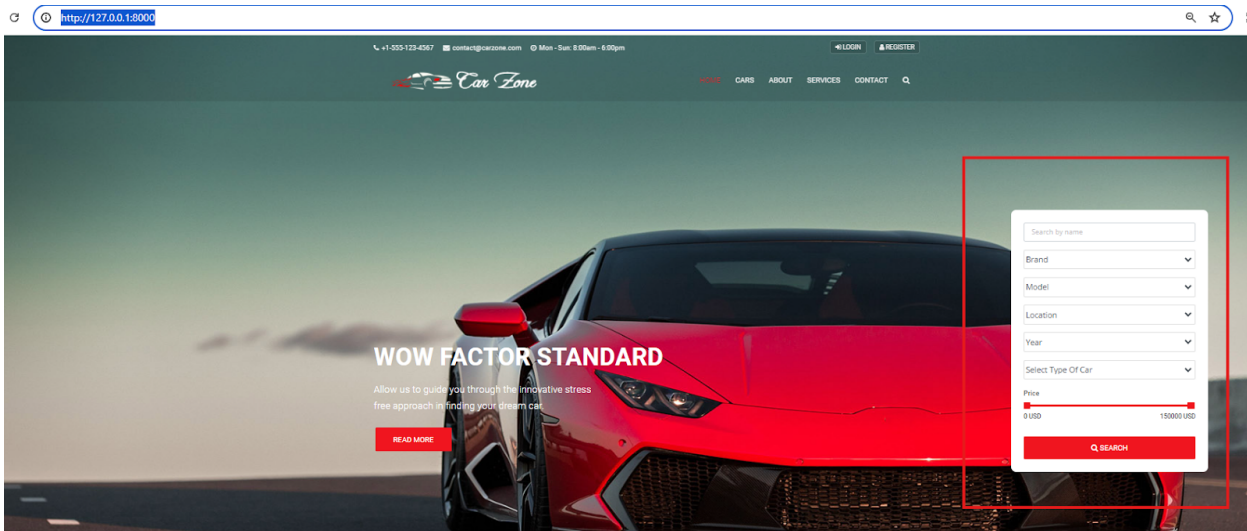
44 <body>
45
46
47 <!-- insert the content of another webpage, topbar.html in this section for the top header content-->
48 <!-- {% include 'includes/topbar.html' %} -->
49
50 <!-- insert the content of another webpage, navbar.html in this section for the navigation bar-->
51 <!-- {% include 'includes/navbar.html' %} -->
52
53 <!-- this block content is where each of the unique webpage content will be dynamically added-->
54 <!-- {% block content %} -->
55
56 <!-- {% endblock %} -->
57
58 <!-- this is the SEARCH OVERLAY -->
59 <!-- Full Page Search -->
60 <div id="full-page-search">
61 <button type="button" class="close"></button>
62 <form action="{% url 'search' %}" class="search-header">
63 <input type="search" value="" placeholder="type keyword(s) here. Eg: audi, benz etc" name="keyword" />
64 <button type="submit" class="btn btn-sm button-theme">Search</button>
65 </form>
66 </div>
67
68 <!-- insert the content of another webpage, footer.html in this section for the footer sections-->
69 <!-- {% include 'includes/footer.html' %} -->
70
71 <!-- {% include 'includes/footer.html' %} -->
72
73 <script src="{% static 'js/jquery-2.2.0.min.js' %}"></script>
74 <script src="{% static 'js/popper.min.js' %}"></script>
75 <script src="{% static 'js/bootstrap.min.js' %}"></script>
76 <script src="{% static 'js/bootstrap-submenu.js' %}"></script>
77 <script src="{% static 'js/rangeslider.js' %}"></script>
78 <script src="{% static 'js/jquery.mb.VTPlayer.js' %}"></script>
79 <script src="{% static 'js/bootstrap-select.min.js' %}"></script>
80 <script src="{% static 'js/jquery.easing.1.3.js' %}"></script>
81 <script src="{% static 'js/jquery.scrollUp.js' %}"></script>
82 <script src="{% static 'js/jquery.mCustomScrollbar.concat.min.js' %}"></script>

```

12. Run the server and search for a word like 'TOYOTA'



13. Add a SEARCH FUNCTIONALITY USING THE SEARCH FORM.



From the HOME.HTML, we remove this BRAND block

```

<form action="{% url 'search' %}" method="">
  <div class="form-group">
    <input type="text" name="keyword" placeholder="Search by name" class="form-control">
  </div>

  <div class="form-group">
    <select class="form-control search-fields" name="select-make">
      <option selected="true" disabled="disabled">Model</option>
      <option>BMW</option>
      <option>Honda</option>
      <option>Lamborghini Huracán</option>
      <option>Sports Car</option>
    </select>
  </div>

```

14. We need to search based on the existing MODEL, YEAR, LOCATION, Go to PAGES\VIEWS.PY and update HOME FUNCTION.

```

def home(request):
    # get all the team members
    teams = Team.objects.all()
    featured_cars = Car.objects.order_by(
        '-created_date').filter(is_featured=True)
    all_cars = Car.objects.order_by('-created_date')
    # extracts the values of these fields for searching
    search_fields = Car.objects.values('model', 'city', 'year', 'body_style')

    data = {
        'teams': teams,
        'featured_cars': featured_cars,
        'all cars': all cars,
        'search_fields': search_fields,
    }

    return render(request, 'pages/home.html', data)

```

15. To display dynamic and current values for searching, update the HOME.HTML

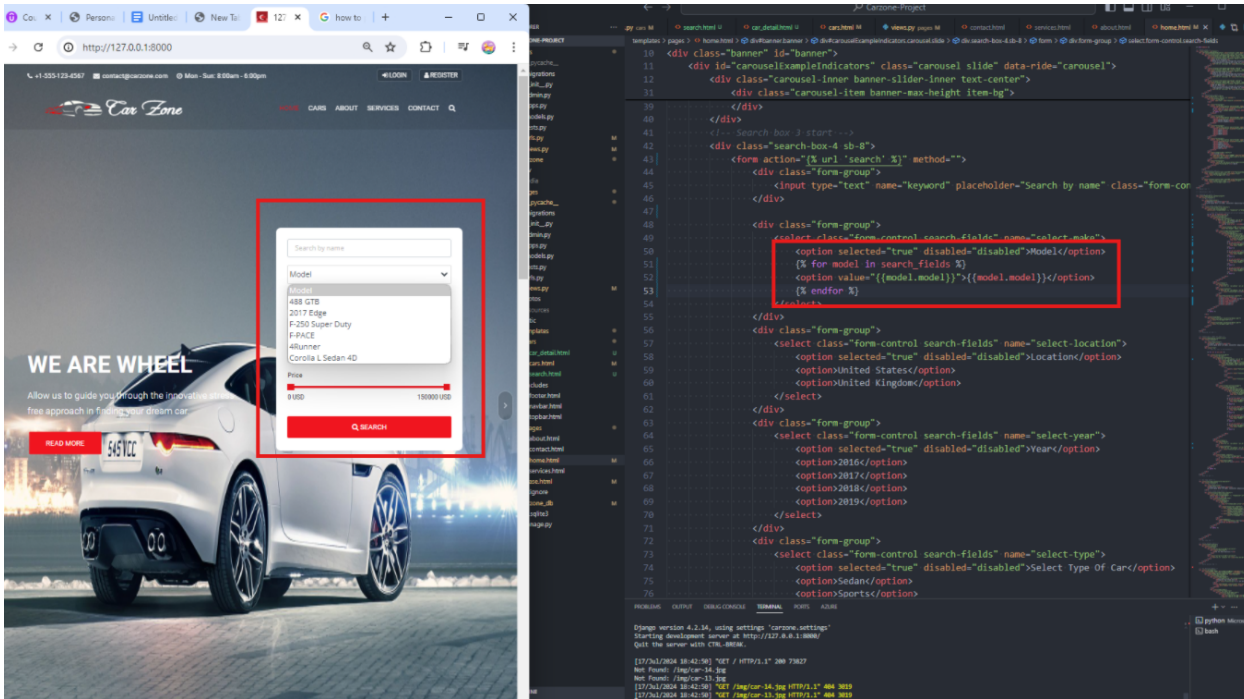
FROM:

```

<div class="form-group">
  <select class="form-control search-fields" name="select-make">
    <option selected="true" disabled="disabled">Model</option>
    <option>BMW</option>
    <option>Honda</option>
    <option>Lamborghini Huracán</option>
    <option>Sports Car</option>
  </select>
</div>

```

TO:



16. To remove the duplicates in the search drop - down box, modify the PAGES\VIEWS.PY home function,

FROM:

```

cars = car.objects.order_by(-created_date)
# extracts the values of these fields for searching
search_fields = Car.objects.values('model', 'city', 'year', 'body_style')

```

TO:

```

# Create your views here.
def home(request):
    # get all the team members
    teams = Team.objects.all()
    featured_cars = Car.objects.order_by(
        '-created_date').filter(is_featured=True)
    all_cars = Car.objects.order_by('-created_date')
    # extracts the values of these fields for searching
    # search_fields = Car.objects.values('model', 'city', 'year', 'body_style')

    # extracts only unique values on models into a list
    model_search = Car.objects.values_list('model', flat=True).distinct()
    city_search = Car.objects.values_list('city', flat=True).distinct()
    year_search = Car.objects.values_list('year', flat=True).distinct()
    body_style_search = Car.objects.values_list(
        'body_style', flat=True).distinct()
    data = {
        'teams': teams,
        'featured_cars': featured_cars,
        'all_cars': all_cars,
        # 'search_fields': search_fields,
        'model_search': model_search,
        'city_search': city_search,
        'year_search': year_search,
        'body_style_search': body_style_search,
    }

    return render(request, 'pages/home.html', data)

```

17. We make a LIST out of CARS query on extracting only UNIQUE VALUES OF MODEL, CITY, YEAR, BODY STYLE. We modify our HOME.HTML as:


```
<div class="form-group">
  <select class="form-control search-fields" name="select-make">
    <option selected="true" disabled="disabled">Model</option>
    {% for model in model_search %}
    <!-- calling the list of model values and not a dictionary -->
    <option value="{{model}}">{{model}}</option>
    {% endfor %}
  </select>
</div>
<div class="form-group">
  <select class="form-control search-fields" name="select-location">
    <option selected="true" disabled="disabled">Location</option>
    {% for city in city_search %}
    <option value="{{city}}">{{city}}</option>
    {% endfor %}
  </select>
</div>
<div class="form-group">
  <select class="form-control search-fields" name="select-year">
    <option selected="true" disabled="disabled">Year</option>
    {% for year in year_search %}
    <option value="{{year}}">{{year}}</option>
    {% endfor %}
  </select>
</div>
<div class="form-group">
  <select class="form-control search-fields" name="select-type">
    <option selected="true" disabled="disabled">Select Type Of Car</option>
    {% for type_of_car in body_style_search %}
    <option value="{{type_of_car}}">{{type_of_car}}</option>
    {% endfor %}
  </select>
</div>
```