**Search page**

**Note**: Loops are extracting the value from the dictionary

1. Create the URL in cars/urls.py

path('search',views.search, name="search")

1. Create the view

def search(request):

return render(request,'cars/search.html')

1. Create the search.html

Then search the url <http://127.0.0.1:8000/cars/search>

1. Copy the search page from the Car House project.
2. Remove all the static data except one.

**Search Functionality**

First go to the search box in home html in it is not in the home.html then go to the base.html.

**If it is in the home.html then cut this and paste in base.html**

{% endblock %}

<!-- Full Page Search -->

<div id="full-page-search">

<button type="button" class="close">×</button>

<form action="{% url 'search' %}" class="search-header" >

<input type="search" value="" placeholder="type keyword(s) here. Eg: audi, benz etc" name="keyword" />

<button type="submit" class="btn btn-sm button-theme">Search</button>

</form>

</div>

{% include 'includes/footer.html' %}

Here I gave the search URL in form action.

And I also use keyword in input field.

**Views.py**

def search(request):

cars = Car.objects.order\_by('-created\_date')

if 'keyword' in request.GET:

keyword = request.GET['keyword']

if keyword:

cars = cars.filter(description\_\_icontains=keyword)

data = {

'cars': cars,

}

return render(request,'cars/search.html',data)

cars = cars.filter(description\_\_icontains=keyword)

**here there are 2 \_\_ underscores**

**search.html**

**Copy the home page rent car**

Copy the loop data only and paste in search page.

Search form on home page

Small box search page

**Note: This code only give the all the data but not unique (distinct data)**

**WRONG QUERY**

**Pages/Views.py (this view is home view)**

search\_fields = Car.objects.values('model','city','year','body\_style')

Here we are passing the values.

and pass this search field context.

Home.html

This loop is extracting value from the dictionry

<!-- Search box 3 start -->

{% for model in search\_fields %}

<option value="{{model.model}}">{{model.model}}</option>

{% endfor %}

{% for city in search\_fields %}

<option value="{{city.city}}">{{city.city}}</option>

{% endfor %}

Do the same for year and body-style.

I want only unique values (like year, city, model etc.)

**CORRECT**

**Pages/Views.py**

Here we wrote query for every field with distinct function(value)

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,

'is\_featured': is\_featured,

'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,

}

**Home.htmls**

{% for model in model\_search %}

<option value="{{model}}">{{model}}</option>

{% endfor %}

**Here we are extracting from the list so we using only the variable name and do the same with year, city and body style**

**Search by the name is also working we already used name=keyword**

**If not then do it.**

**Now we should change the model, city, year and body\_style name in html form.**

**Here I change the name = model**

**Do it for city, year and body\_style**

**But price is already given min\_price and max\_price.**

<div class="form-group">

<select class="form-control search-fields" name="model">

<option selected="true" disabled="disabled">Model</option>

{% for model in model\_search %}

<option value="{{model}}">{{model}}</option>

{% endfor %}

</select>

</div>

**Now when we request with keyword/name so we have to check in views that we have that car or not.**

So we have to use separate conditions for:

**And This words are model.py’s name not the keyword/name.**

model

city

year

body\_style

price also (but will be different from other conditions)

**And we used iexact for exact match and here we use model.py’s variable name not keyword/name .**

def search(request):

cars = Car.objects.order\_by('-created\_date')

if 'keyword' in request.GET:

keyword = request.GET['keyword']

if keyword:

cars = cars.filter(description\_\_icontains=keyword)

data = {

'cars': cars,

}

if 'model' in request.GET:

model = request.GET['model']

if model:

cars = cars.filter(model\_\_iexact=model)

data = {

'cars': cars,

}

if 'city' in request.GET:

city = request.GET['city']

if city:

cars = cars.filter(city\_\_iexact=city)

data = {

'cars': cars,

}

if 'year' in request.GET:

year = request.GET['year']

if year:

cars = cars.filter(year\_\_iexact=year)

data = {

'cars': cars,

}

if 'body\_style' in request.GET:

body\_style = request.GET['body\_style']

if body\_style:

cars = cars.filter(body\_style\_\_iexact=body\_style)

data = {

'cars': cars,

}

**Here in inner if we use price in filter query bcz we used price name in models.py**

if 'min\_price' in request.GET:

min\_price = request.GET['min\_price']

max\_price = request.GET['max\_price']

if max\_price:

cars = cars.filter(price\_\_gte=min\_price, price\_\_lte=max\_price)

data = {

'cars': cars,

}

return render(request,'cars/search.html',data)

**Search page inside search page**

Inside the search page we have another Search page.

Do the same as search page

1. Remove the brand
2. But here copy all the distinct query inside the search view and add the transmission search and also the condtion.

def search(request):

cars = Car.objects.order\_by('-created\_date')

model\_search = Car.objects.values\_list('model',flat=True).distinct()

year\_search = Car.objects.values\_list('year',flat=True).distinct()

city\_search = Car.objects.values\_list('city',flat=True).distinct()

body\_style\_search = Car.objects.values\_list('body\_style',flat=True).distinct()

transmission\_search = Car.objects.values\_list('transmission',flat=True).distinct()

if 'transmission' in request.GET:

transmission = request.GET['transmission']

if transmission:

cars = cars.filter(transmission\_\_iexact=transmission)

data = {

'cars': cars,

}

And pass the transmission\_search inside the data the last data context.

We can delete all the data context from search view except the last one.

It will be fine.

data = {

'cars': cars,

'model\_search': model\_search,

'year\_search':year\_search,

'city\_search':city\_search,

'body\_style\_search':body\_style\_search,

'transmission\_search':transmission\_search,

}