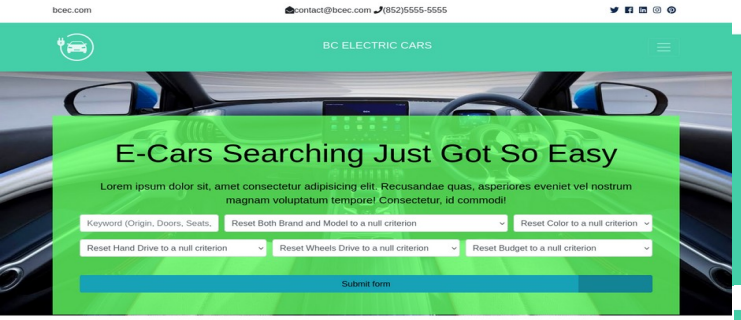
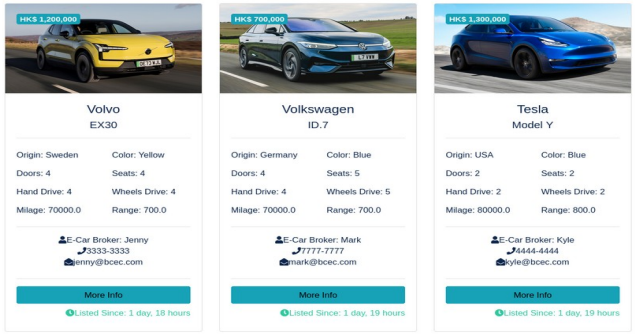
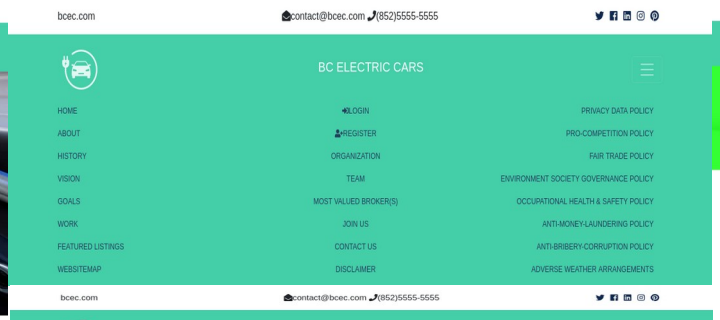
BC Electric Cars

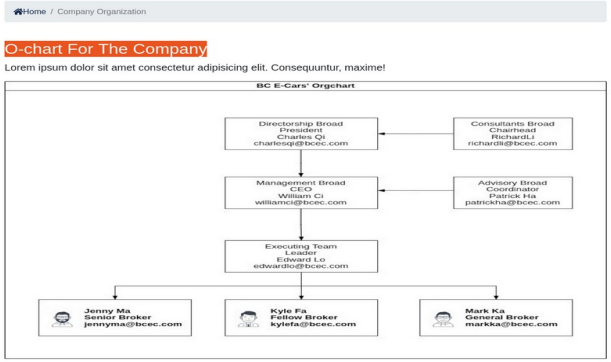
Summary: Building upon design of given bcre.com, bcec.com for web of e-cars improved searching, enchanced listing, enlightened navbar menu, incorporated a websitemap, and included some more inform pages.

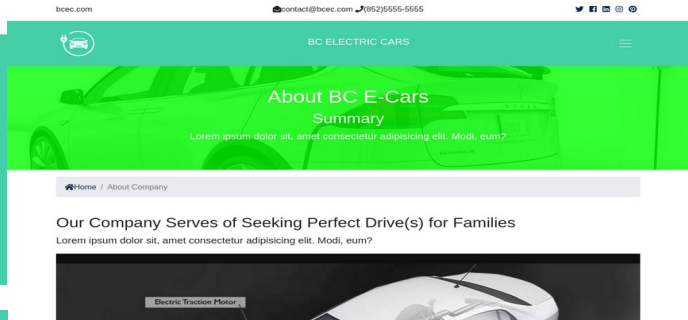
Project Creator: Frank, Mok Wai Tak Student 21 of HKCT Django Course the Fifth Class 17-Oct-2024











Limitations on Django

Web Application Framework itself not completely full-stacked with frontends of nondirect roundabout CRUDs, drillings thru' html forms' submit buttons back to an app's views functions in where they being carried out.

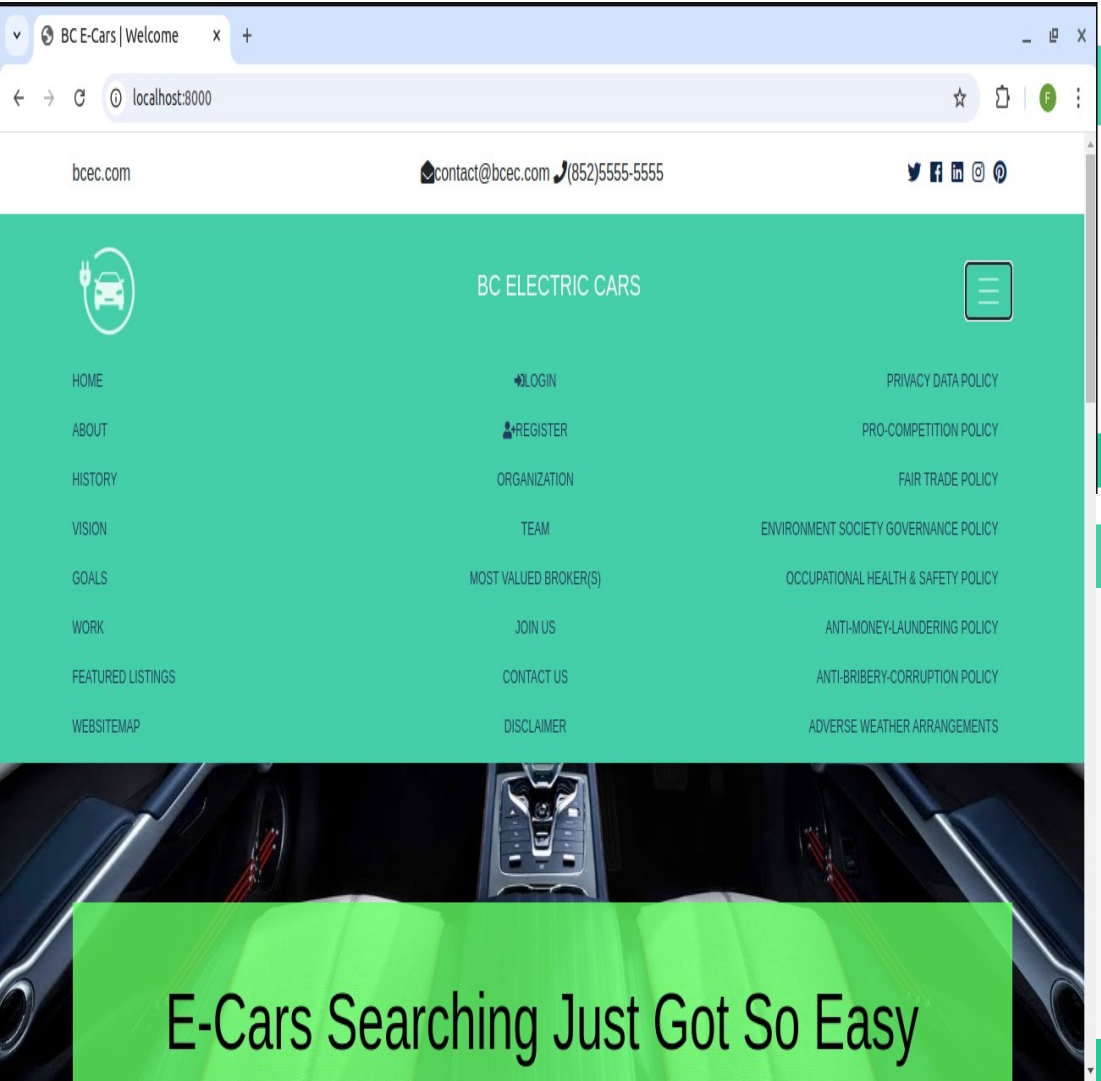
Does imply itself not strong on all kinds of applications.

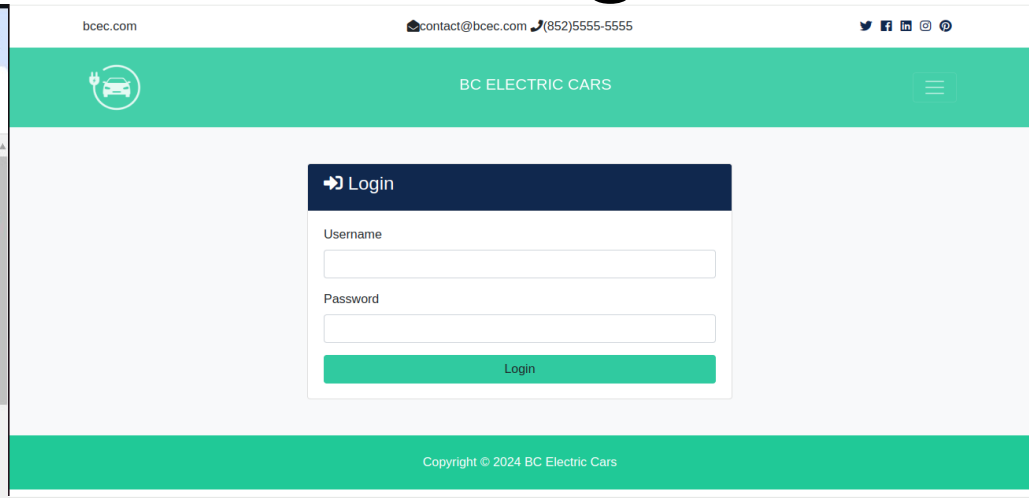
Analysis on bcre.com

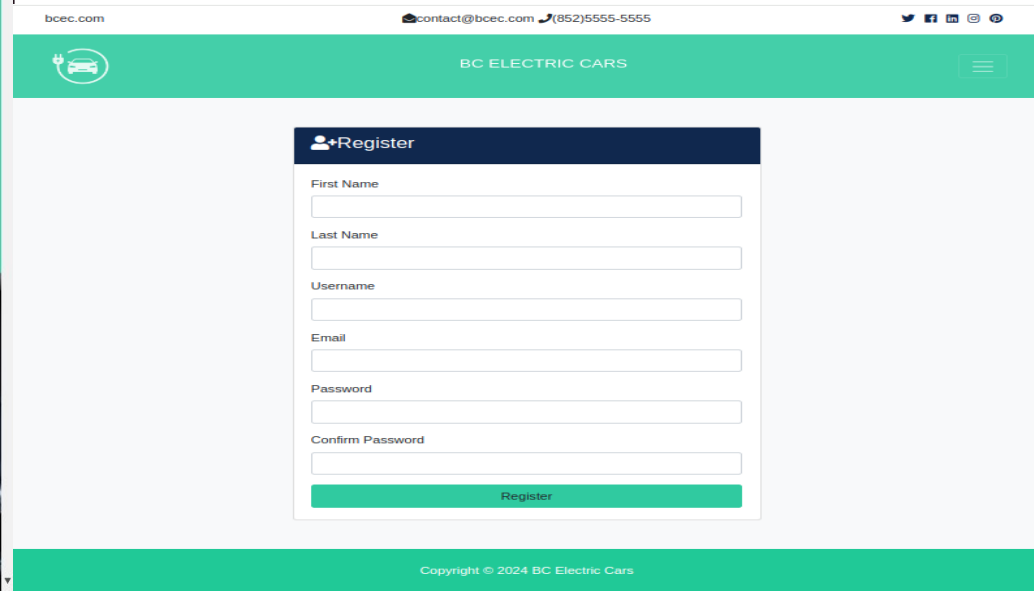
- Generic on products of very high price, very low volume mostly of uniqueness / singularity
- Instances for products like jewelry, artefacts, artworks, paintings, timepieces, real estates, vehicles (vessels, aircrafts), plant and instrumentation
- Transactions via top banks, good law firms offline c/w financial, insurantial and legal arrangements
- A promotionally informative system online

Different products, Different attributes

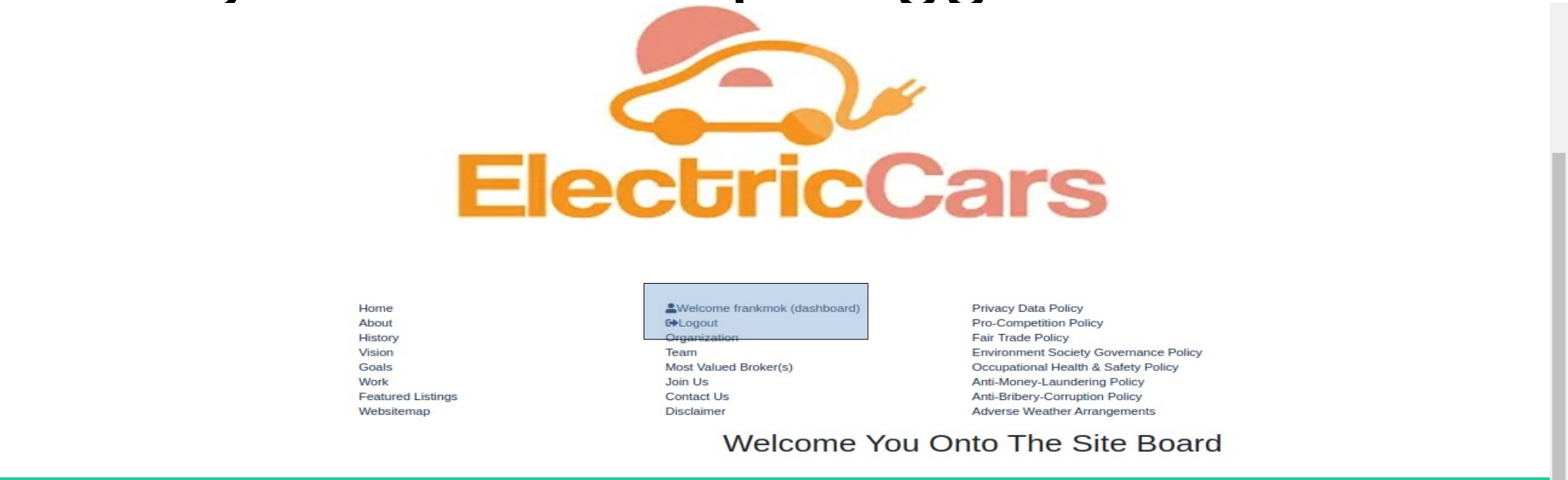
Enriched NavBar-Left/Center/Right Cols



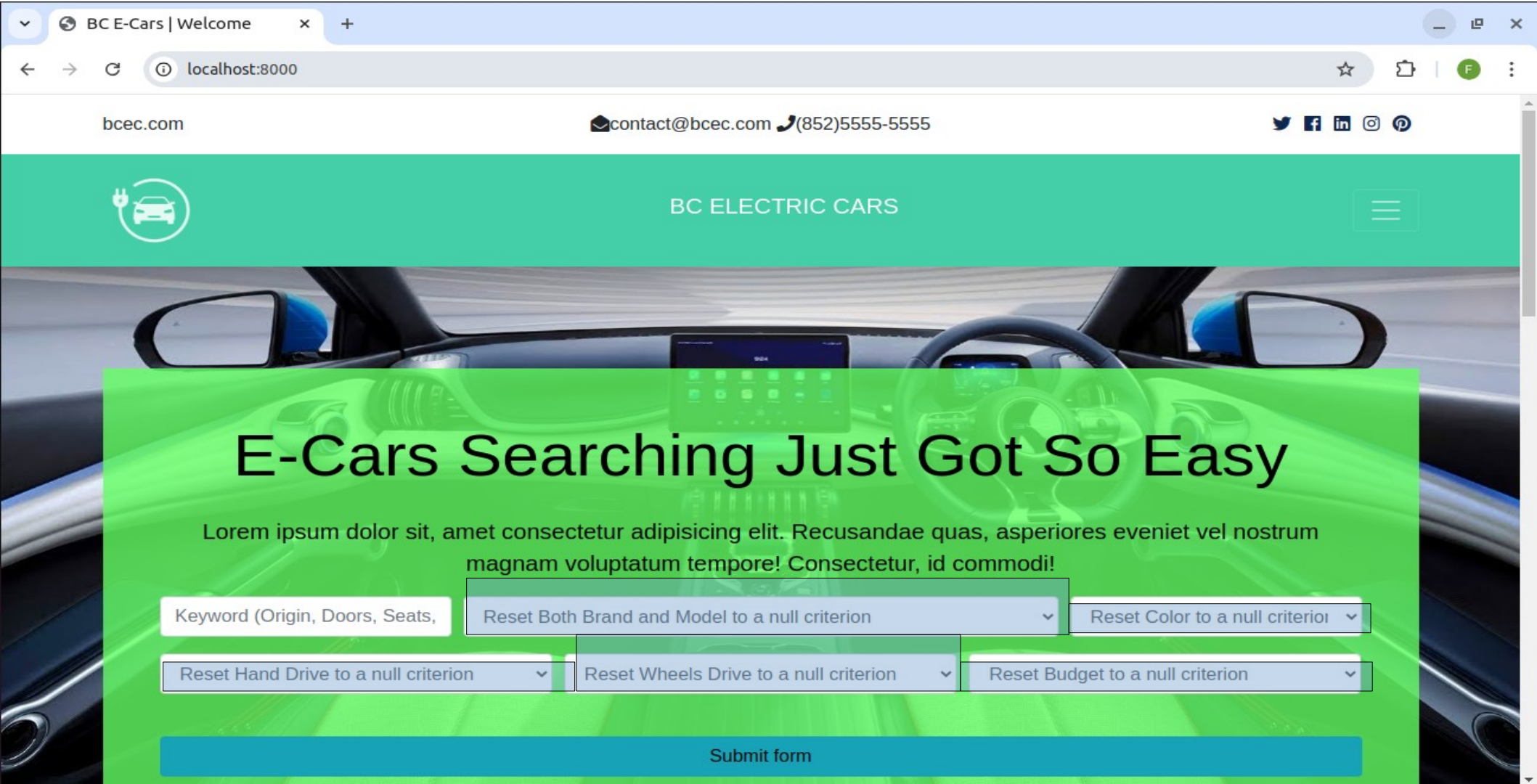




Newly WebSiteMap: logged-in/out differ



Enabled Reset Some Search Fields



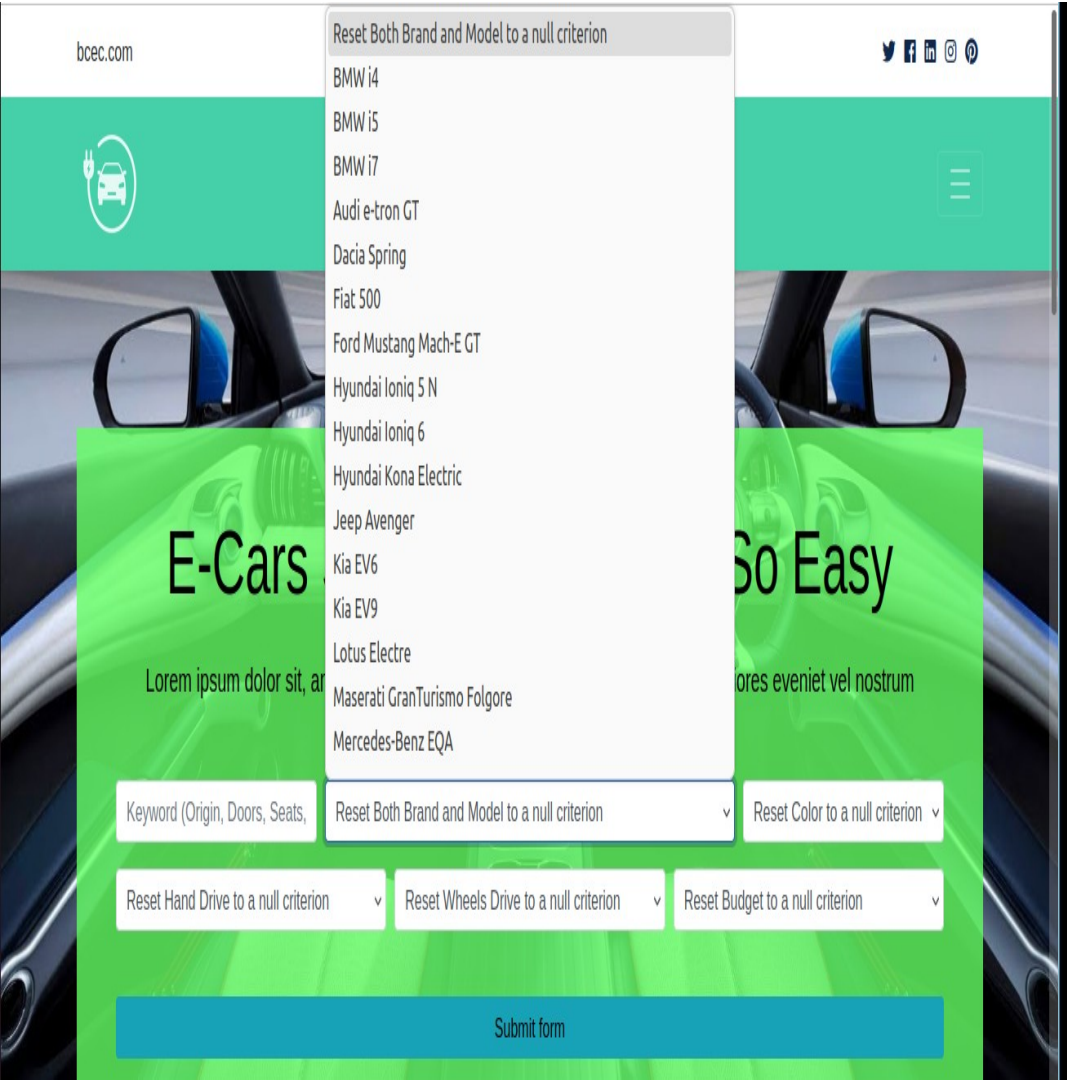
UX Design – Realisticity in Search

Some improvements on bcre website can be relied upon UX Design on the search boxes.
UX can be increased by offer of more realisticity senses to users in the options' choices.
Two things can be done on the search boxes:

1. combinations of search choices
2. real search choices softcoded

Let see the bcec website search box first: *P.T.O.*

search boxes and search fields

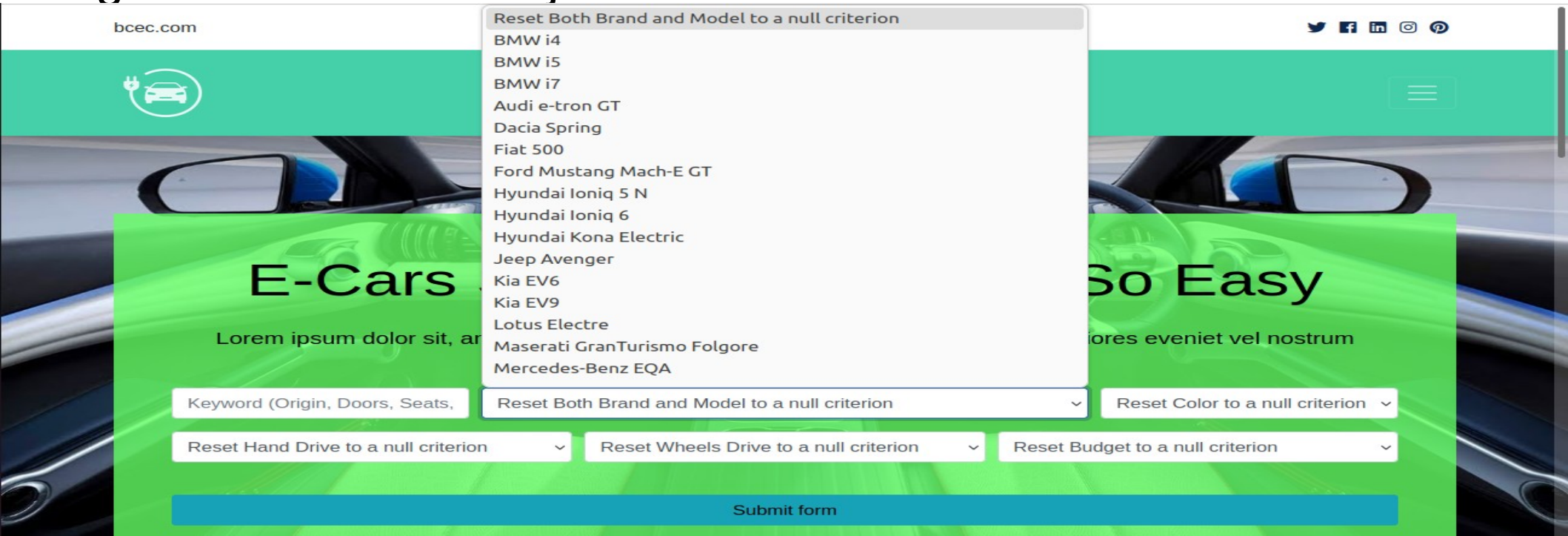



```python
class Listing(models.Model):
    broker = models.ForeignKey(Broker, on_delete=models.DO_NOTHING)
    brand = models.CharField(max_length=50)
    model = models.CharField(max_length=100)
    origin = models.CharField(max_length=50)
    color = models.CharField(max_length=50)
    doors = models.IntegerField()
    seats = models.IntegerField()
    milage = models.DecimalField(max_digits=6, decimal_places=1)
    range = models.DecimalField(max_digits=4, decimal_places=1)
    hand_drive = models.CharField(max_length=50)
    wheels_drive = models.CharField(max_length=50)
    price = models.IntegerField()
    description = models.TextField(blank=True)
    list_date = models.DateTimeField(auto_now_add=True)
    is_published = models.BooleanField(default=True)
    photo_main = models.ImageField(upload_to='photos/%Y/%m/%d/')
    photo_1 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)
    photo_2 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)
    photo_3 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)
    photo_4 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)
    photo_5 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)
    photo_6 = models.ImageField(upload_to='photos/%Y/%m/%d/', blank=True)

    def __str__(self):
        return self.brand + ' ' + self.model
```

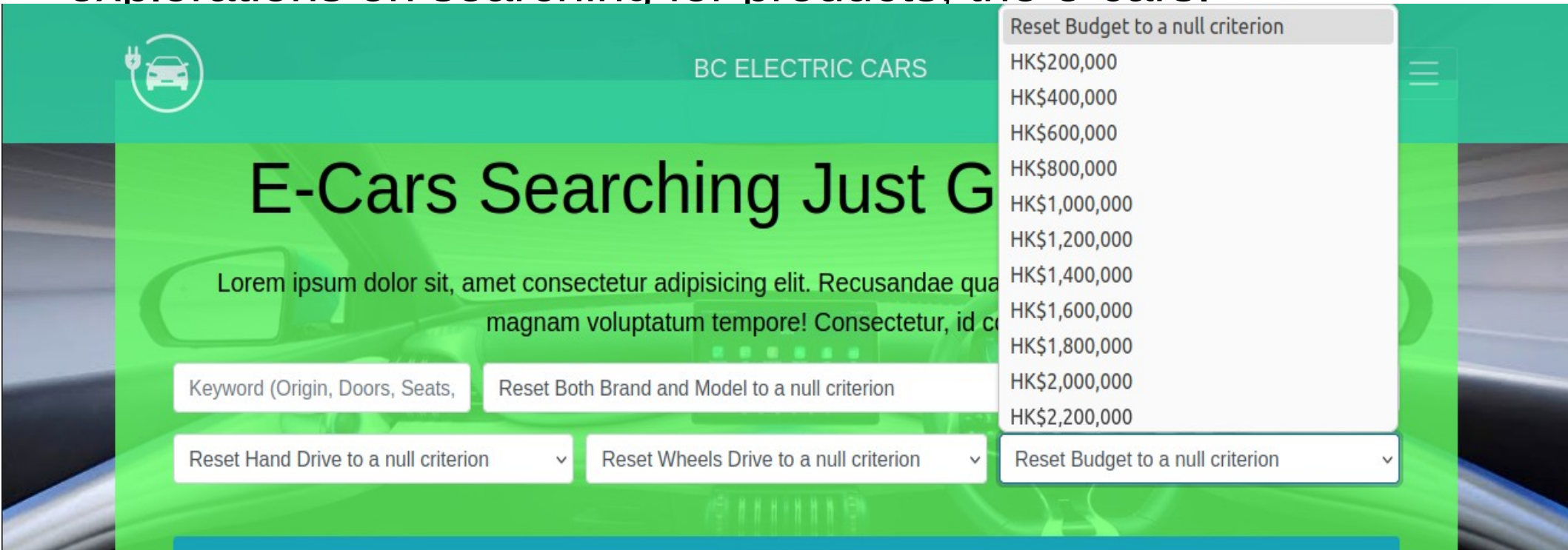
Combinations of Search Choices / Combined Fields Searches

Some relatively major search choices in this case such as brand field and model field be combined to restrict from users' unrealistic nonexistent combinations e.g. M.Benz i5 to exclude possible confusions in selecting options to keep up their own rigorous sense of majorness.



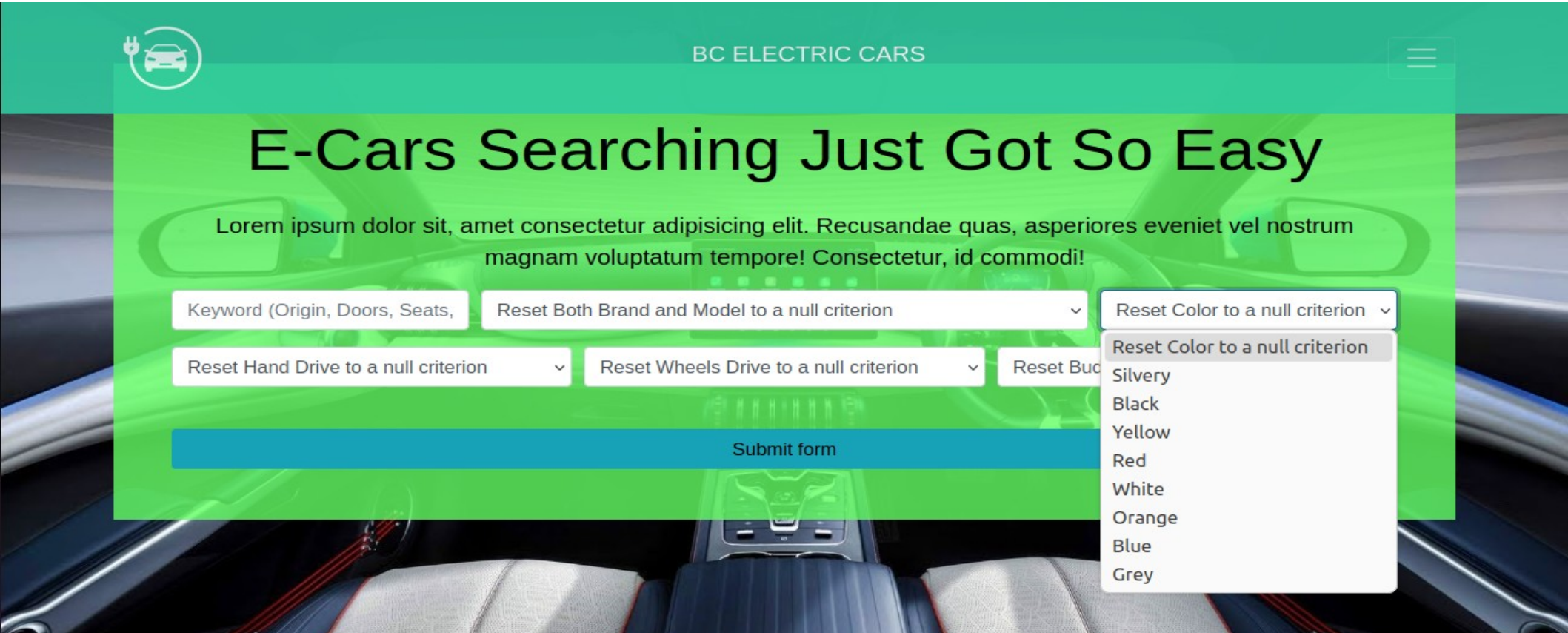
No Combinations of Search Choices / Individualized Field Searches

Other relatively minor search choices in this case such as color field and budget field be individualized as of users' preferential freedom allowing for negative unsuccessful combinations e.g. below a budget not a color offered retaining users' joy in course of explorations on searching for products, the e-cars.



Field Searches' Real Choices

Field searches' real choices can be derived from database's data existent reflecting true options for potential customers, increasing informative values for users in selections of products the e-cars.



How can we achieve?

See my share of python / django coding tips, p.t.o.

to share django / python coding tips: search field color

listings > choices.py > ...

```python
listings = Listing.objects.all().filter(is_published=True)
colors_set = {listing.color for listing in listings}
color_choices = {'Reset Color to a null criterion': ''} | {color: color for color in colors_set}
```

templates > pages > index.html

```html
<div class="col-md-3 mb-3">
  <label class="sr-only" for="color">Colour</label>
  <select name="color" class="form-control" id="color">
    {% for key, value in color_choices.items %}
    <option value="{{value}}">{{key}}</option>
    {% endfor %}
  </select>
</div>
```

1: set of colors from db 2:dictionary 3:add reset null to dict head 4:options, dynamic from db

to share django / python coding tips: search field hand_drive

listings > choices.py > ...

```python
listings = Listing.objects.all().filter(is_published=True)
hand_drives_set = {listing.hand_drive for listing in listings}
hand_drive_choices = {'Reset Hand Drive to a null criterion': ''} |
{hand_drive + '-hand-drive': hand_drive for hand_drive in hand_drives_set}
```

templates > pages > index.html

```html
<div class="form-row">
  <div class="col-md-4 mb-3">
    <label class="sr-only" for="hand_drive">Hand Drive</label>
    <select name="hand_drive" class="form-control" id="hand_drive">
      {% for key, value in hand_drive_choices.items %}
      <option value="{{value}}">{{key}}</option>
      {% endfor %}
    </select>
  </div>
```

1: set of hand_drives from db 2:dictionary 3:add reset null to dict head 4:options, dynamic from db

to share django / python coding tips: search field wheels drive

listings > choices.py > ...

```python
listings = Listing.objects.all().filter(is_published=True)
wheels_drives_set = {listing.wheels_drive for listing in listings}
wheels_drive_choices = {'Reset Wheels Drive to a null criterion': ''} |
{wheels_drive + '-wheels-drive': wheels_drive for wheels_drive in wheels_drives_set}
```

templates > pages > index.html

```html
<div class="col-md-4 mb-3">
  <label class="sr-only" for="wheels_drive">Wheels Drive</label>
  <select name="wheels_drive" class="form-control" id="wheels_drive">
    {% for key, value in wheels_drive_choices.items %}
    <option value="{{value}}">{{key}}</option>
    {% endfor %}
  </select>
</div>
```

1: set of wheels_drives from db 2:dictionary 3:add reset null to dict head 4:options, dynamic from db

to share django / python coding tips: search field budget

listings > choices.py > ...

```
3   listings = Listing.objects.all().filter(is_published=True)
10  max_price = max({listing.price for listing in listings})
11  grade = 200000
12  budget_choices = {'Reset Budget to a null criterion': '0'} |
13  {'HK$'+f'{g:,}': str(g) for g in range(grade, max_price + grade, grade)}
```

templates > pages > index.html

```
67                  <div class="col-md-4 mb-3">
68                    <label class="sr-only" for="budget">Budget</label>
69                    <select name="budget" class="form-control" id="budget">
70                      {% for key, value in budget_choices.items %}
71                      <option value="{{value}}">{{key}}</option>
72                      {% endfor %}
73                    </select>
74                  </div>
```

1: max of set of prices from db 2:grade 3: grades of range() 4: dictionary 3:add reset null to dict head 4:options, dynamic from db

django / python coding tips: combination field brand & model

listings > choices.py > ...

```python
listings = Listing.objects.all().filter(is_published=True)
brand_model_choices = {'Reset Both Brand and Model to a null criterion': ''} |
{listing.brand + '  ' + listing.model: listing.brand + '~' + listing.model for listing in listings}
```

templates > pages > index.html

```html
<div class="col-md-6 mb-3">
  <label class="sr-only" for="brand_model">Brand                Model</label>
  <select name="brand_model" class="form-control" id="brand_model">
    {% for key, value in brand_model_choices.items %}
    <option value="{{value}}">{{key}}</option>
    {% endfor %}
  </select>
</div>
```

1:dictionary of brand+’ ‘+model : brand+’~’+model 2:add reset null to dict head 3:options, dynamic from db

django / python coding tips: the search code

listings > views.py > search

```python
def search(request):
    queryset_list = Listing.objects.order_by('-list_date').filter(is_published=True)
    if 'keywords' in request.GET:
        keywords = request.GET['keywords']
        if keywords:
            queryset_list = queryset_list.filter(description__icontains=keywords)
    if 'brand_model' in request.GET:
        brand_model = request.GET['brand_model']
        if brand_model:
            i = brand_model.index('~')
            queryset_list = queryset_list.filter(brand__iexact=brand_model[:i]).filter(model__iexact=brand_model[i+1:])
    if 'color' in request.GET:
        color = request.GET['color']
        if color:
            queryset_list = queryset_list.filter(color__iexact=color)
    if 'budget' in request.GET:
        budget = int(request.GET['budget'])
        if budget:
            queryset_list = queryset_list.filter(price__lte=budget)
    if 'hand_drive' in request.GET:
        hand_drive = request.GET['hand_drive']
        if hand_drive:
            queryset_list = queryset_list.filter(hand_drive__iexact=hand_drive)
    if 'wheels_drive' in request.GET:
        wheels_drive = request.GET['wheels_drive']
        if wheels_drive:
            queryset_list = queryset_list.filter(wheels_drive__iexact=wheels_drive)
```

1.quest for search field key in json 2.read in search field value in json 3.screen out null criterion search passing wholly 4.search non-null criterion 5.pass to repeat till all search fields 6.for brand & model: double searches / filters in succession

django / python coding tips: the websitemap code

templates > pages > sitemap.html

```
<div class="row">
    <div class="col-md-4">
        <ul style="list-style-type:none;">
            <li>
                <a href={% url "index" %}>Home</a>
            </li>
            <li>
                <a href={% url "about" %}>About</a>
            </li>
            <li>
                <a href={% url "history" %}>History</a>
            </li>
            <li>
                <a href={% url "vision" %}>Vision</a>
            </li>
            <li>
                <a href={% url "goals" %}>Goals</a>
            </li>
            <li>
                <a href={% url "work" %}>Work</a>
            </li>
            <li>
                <a href={% url "listings" %}>Featured Listings</a>
            </li>
            <li>
                <a href={% url "sitemap" %}>Websitemap</a>
            </li>
        </ul>
    </div>
            <li>
                <a href={% url "organization" %}>Organization</a>
            </li>
            <li>
                <a href={% url "team" %}>Team</a>
            </li>
            <li>
                <a href={% url "mvb" %}>Most Valued Broker(s)</a>
            </li>
            <li>
                <a href={% url "joinus" %}>Join Us</a>
            </li>
            <li>
                <a href={% url "contactus" %}>Contact Us</a>
            </li>
            <li>
                <a href={% url "disclaimer" %}>Disclaimer</a>
            </li>
        </ul>
    </div>
    <div class="col-md-4">
        <ul style="list-style-type:none;">
            <li>
                <a href={% url "pdpolicy" %}>Privacy Data Policy</a>
            </li>
<div class="col-md-4">
    <ul style="list-style-type:none;">
        {% if user.is_authenticated %}
        <li>
            <a href={% url "dashboard" %}><i class="fas fa-user"></i>Welcome {{user.username}} (dashboard)</a>
        </li>
        <li>
            <a href="javascript: {document.getElementById('logout').submit()}">
                <i class="fas fa-sign-out-alt"></i>Logout</a>
            <form action="{% url 'logout' %}" method="POST" id="logout">
            {% csrf_token %}
                <input type="hidden" />
            </form>
        </li>
        {% else %}
        <li>
            <a href={% url "login" %}><i class="fas fa-sign-in-alt"></i>Login</a>
        </li>
        <li>
            <a href={% url "register" %}><i class="fas fa-user-plus"></i>Register</a>
        </li>
        {% endif %}
            <li>
                <a href={% url "pcpolicy" %}>Pro-Competition Policy</a>
            </li>
            <li>
                <a href={% url "ftpolicy" %}>Fair Trade Policy</a>
            </li>
            <li>
                <a href={% url "esgpolicy" %}>Environment Society Governance Policy</a>
            </li>
            <li>
                <a href={% url "ohspolicy" %}>Occupational Health & Safety Policy</a>
            </li>
            <li>
                <a href={% url "amlpolicy" %}>Anti-Money-Laundering Policy</a>
            </li>
            <li>
                <a href={% url "abcpolicy" %}>Anti-Bribery-Corruption Policy</a>
            </li>
            <li>
                <a href={% url "awarrange" %}>Adverse Weather Arrangements</a>
            </li>
        </ul>
    </div>
</div>
```

Conclusion

Before conclusion, demo time of bece.com .

Finally, UX is an important niche of software design as ultimate users are human. It much increases the value of use, i.e. usability, of the software.

Thank you for your participation.

Thanksgiving to Wai Lung Sir.

Chance comes, meeting again.