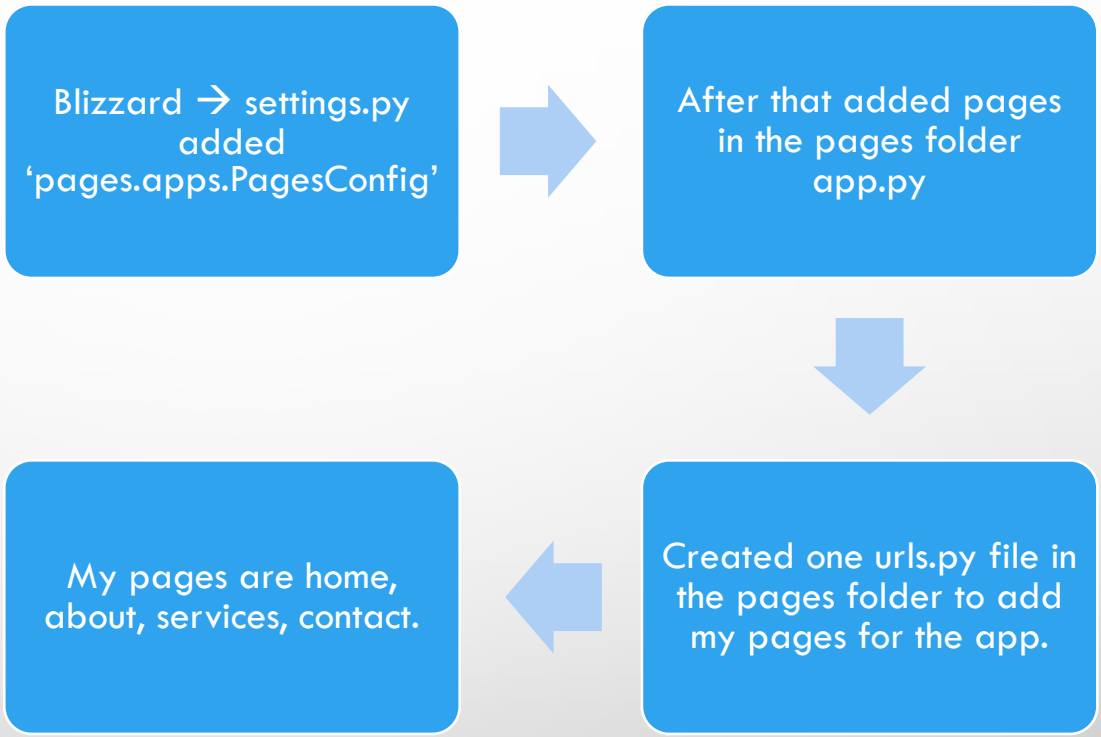
INTERNSHIP PROJECT

TECHNICAL PRESENTATION

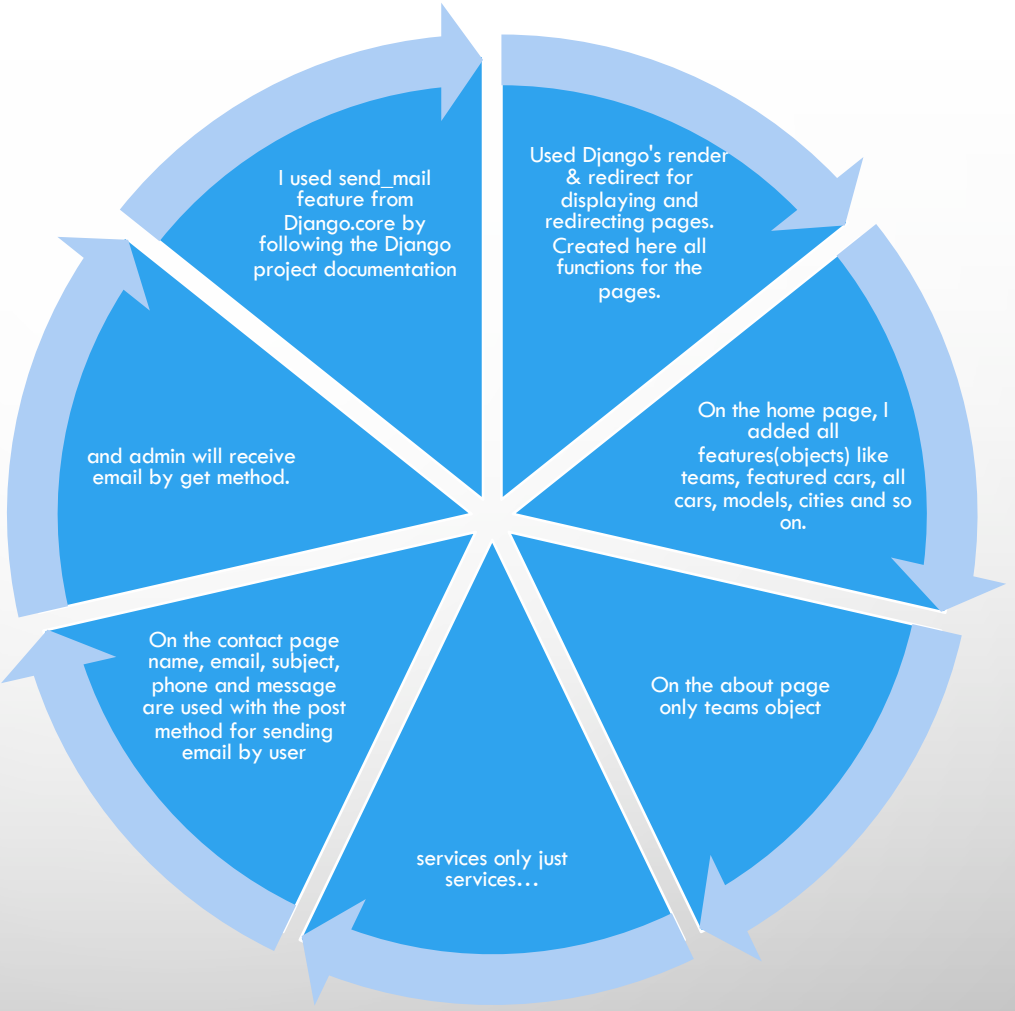
BIZZARD AUTO

Created app by running the command:

```
python manage.py startapp blizzard
```



Pages → views.py



blizzard →

settings.py

Added:

```
TEMPLATES = [
{

'DIRS': ['templates'],

}

]

>>>>>>>>>>


STATIC_ROOT = os.path.join(BASE_DIR, 'static')

STATICFILES_DIRS = [
    os.path.join(BASE_DIR, 'blizzard/static'),
]
```

root→templates

Here I created all pages and sub-pages folders

Main:	base.html
Accounts:	dashboard.html, login.html, register.html
Admin:	base_site.html
Cars:	car_detail.html, cars.html, search.html
Includes:	footer.html, messages.html, navbar.html, topbar.html
Pages:	home.html, about.html, services.html, contact.html

In this part, I installed using pip

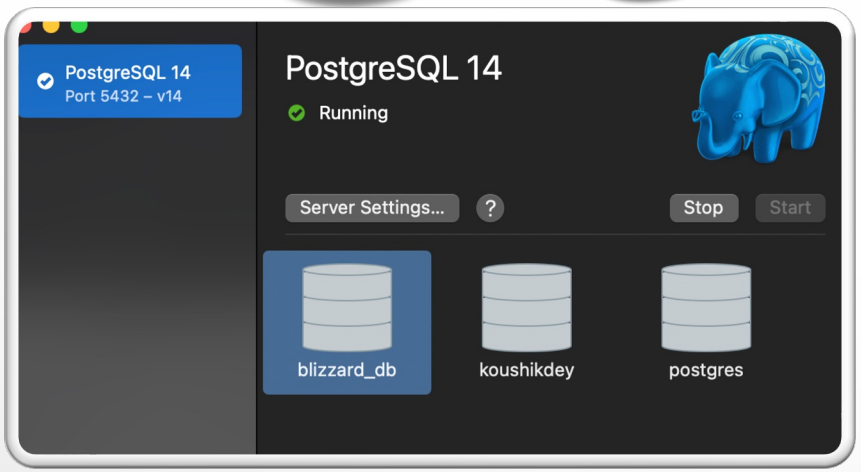
- PostgreSQL
- Psycopg2-binary (PostgreSQL DB adapter for Python)
- Pillow (For processing basic image functionality)
- After that I created admin/superuser by running the command:
  - *python manage.py createsuperuser*

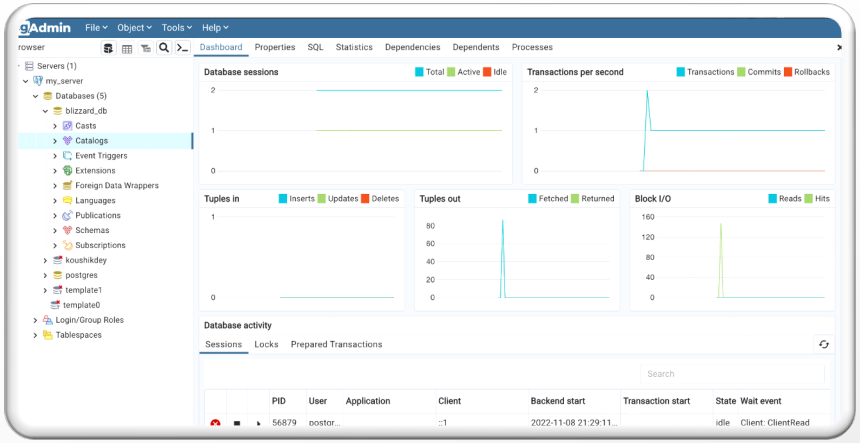
PostgreSQL

settings.py M

blizzard > settings.py > ...

```
100
101 # Database
102 # https://docs.djangoproject.com/en/3.0/ref/settings/#databases
103
104 DATABASES = {
105     'default': {
106         'ENGINE': 'django.db.backends.postgresql',
107         'NAME': 'blizzard_db',
108         'USER': 'postgres',
109         'PASSWORD': config('DB_PASSWORD'),
110         'HOST': 'localhost',
111     }
112 }
```





PostgreSQL & pgAdmin panel

```
koushik, last month | 1 author (koushik)
class Client(models.Model):
    name = models.CharField(max_length=100)
    email = models.EmailField(max_length=100)
    subject = models.CharField(max_length=250)
    phone = models.CharField(max_length=100)
    message = models.TextField(blank=True)

    def __str__(self):
        return self.name
```

```
#from distutils.command.upload import upload
from django.db import models

# Created models for Team and Clients.      Find related code in blizzard-django-project

koushik, 2 months ago | 1 author (koushik)
class Team(models.Model):
    first_name = models.CharField(max_length=255)
    last_name = models.CharField(max_length=255)
    designation = models.CharField(max_length=255)
    photo = models.ImageField(upload_to='photos/%Y/%m/%d/')
    facebook_link = models.URLField(max_length=100)
    twitter_link = models.URLField(max_length=100)
    google_plus_link = models.URLField(max_length=100)
    created_date = models.DateTimeField(auto_now_add=True)
```

```
models.py M    admin.py
pages > admin.py > ...
6   # Register your models here.
    koushik, last month | 1 author (koushik)
7   class TeamAdmin(admin.ModelAdmin):
8       def thumbnail(self, object):
9           return format_html('<img src="{}" width="40" style="border-radius: 10px;" />'.format(
10
11      thumbnail.short_description = 'Photo'
12
13      list_display = ('id', 'thumbnail', 'first_name', 'last_name', 'designation', 'created_dat
14      list_display_links = ('id', 'thumbnail','first_name',)
15      search_fields = ('first_name', 'last_name', 'designation')
16      list_filter = ('designation',)
17
18  admin.site.register(Team, TeamAdmin)
19
    koushik, last month | 1 author (koushik)
20  class ClientAdmin(admin.ModelAdmin):
21      list_display = ('name', 'email', 'subject', 'phone', 'message')
22      list_display_links = ('name', 'email', 'message')
23      search_fields = ('name', 'subject', 'email')
24      list_per_page = 10
25      list_filter = ('name', 'email', 'subject')
26
27  admin.site.register(Client, ClientAdmin)      Find related code in blizzard-django-project
28
```

CREATED DB MODEL FOR TEAM AND CLIENTS

THEN RUN COMMANDS TO SAVE MODELS IN POSTGRESQL:

- python manage.py makemigrations
- python manage.py migrate

Then registered these models to admin.py

models.py M urls.py

blizzard > urls.py > ...

```
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from django.conf.urls.static import static
from django.conf import settings


urlpatterns = [        Find related code in blizzard-django-project
    path('admin/', admin.site.urls),
    path('', include('pages.urls')),
    path('cars/', include('cars.urls')),
    path('accounts/', include('accounts.urls')),
    path('socialaccounts/', include('allauth.urls')),
    path('contacts/', include('contacts.urls')),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

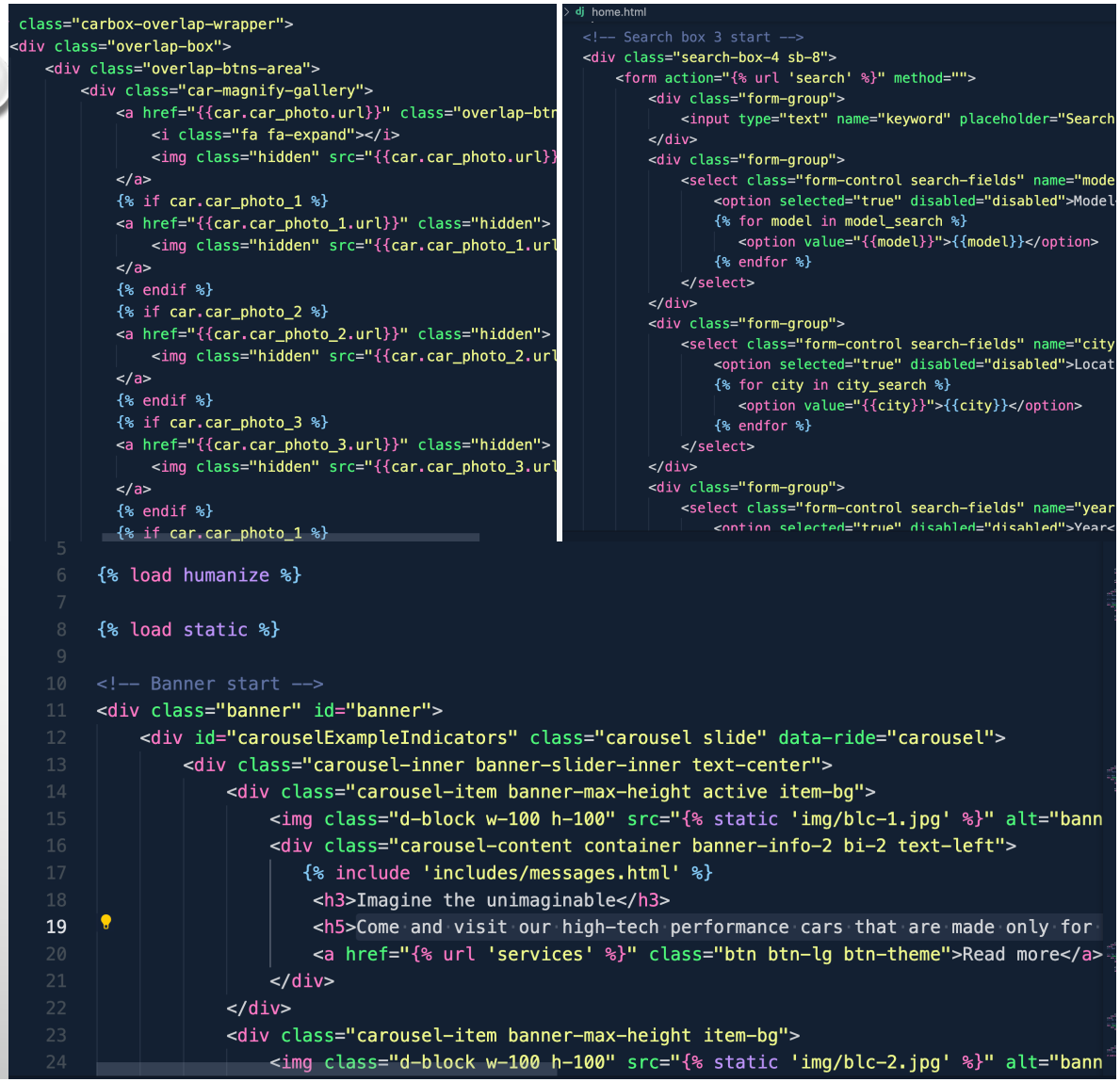
```
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
STATICFILES_DIRS = [
    os.path.join(BASE_DIR, 'blizzard/static'),
]

# Media Settings

MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'

# Messages

from django.contrib.messages import constants as messages
MESSAGE_TAGS = {
    messages.ERROR: 'danger',
}

```

MEDIA SETTINGS BLIZZARD

settings.py & urls.py

HTML PAGES

HOME, ABOUT, SERVICES & CONTACT



```
<h3>Create an account</h3>
{% include 'includes/messages.html' %}
<form action="{% url 'register' %}" method="POST">
    {% csrf_token %}
    <div class="form-group form-box">
        <input type="text" name="firstname" class="input-te
        <i class="fa fa-user"></i>
```

```
<form action="{% url 'contact' %}" method="POST" enctype="multipart/form-data">
    {% include 'includes/messages.html' %}
    {% csrf_token %}
    <div class="row">
        <div class="col-lg-7">
            <div class="row">
                <div class="col-md-6 text-left">
                    <div class="form-group name">
```

```
<li>
    <a href="javascript:{document.getElementById('logout').submit()}" class="sign-in"><i clas
    <form action="{% url 'logout'%}" id="logout" method="POST">
        {% csrf_token %}
        <input type="hidden">
    </form>
</li>
{% else %}
```

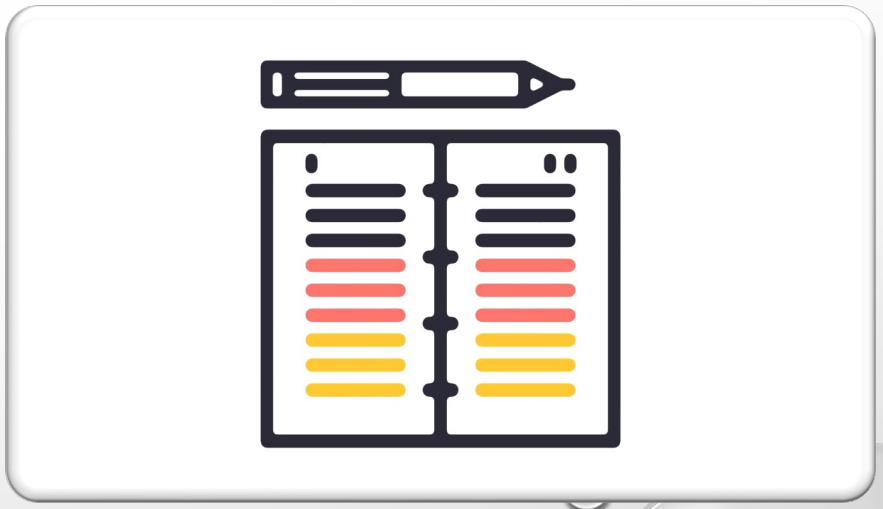
When I use the post method I need to use

{% CSRF_TOKEN %}

To prevent cross-site forgery

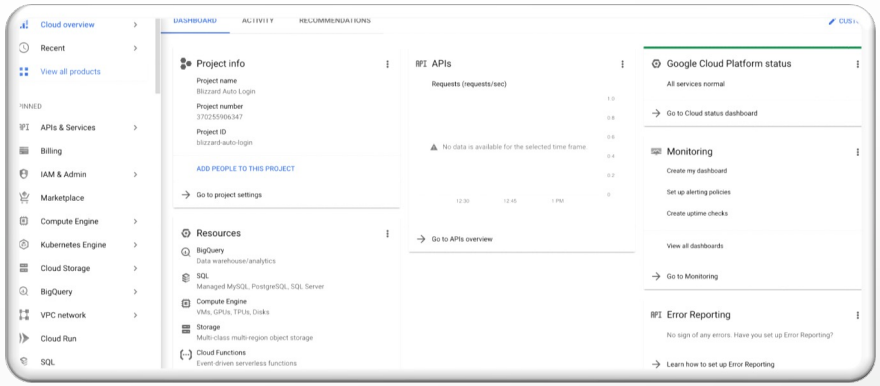
USER REGISTRATION

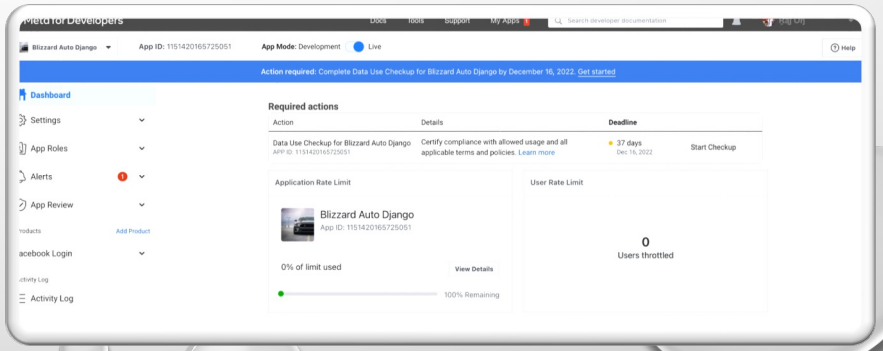
I didn’t make a model here as Django comes with a built-in user model. and I didn’t use hashing technique as Django is using sha256 hashing technique.



SOCIAL LOGIN:

GOOGLE & FB CONSOLE FOR DEVELOPERS





LOGOUT FUNCTIONALITY

User authentication is by default enabled in Django, so it wasn't to call it from anywhere.

INSTALLED DECOUPLE

pip install python-decouple

Procedures in settings.py

from decouple import config

Created an env file in the root folder.

DEBUG = config('DEBUG', cast=bool) which I mentioned as True in the .env file.

All other secret keys like APP_SECRET_KEY, DB_PASSWORD and SMTP EMAIL_HOST_PASSWORD are saved in the .env file.

DATABASE BACKUP/DUMPDATA & REQUIREMENTS

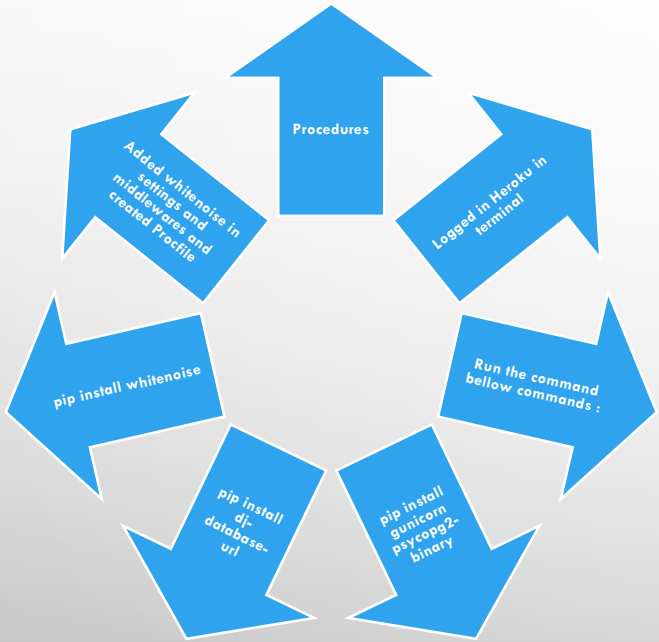
Run these commands below to create a back-up in JSON file and all requirements for blizzard app

```
python manage.py dumpdata --natural-foreign --natural-primary -e contenttypes -e auth.permission --indent 4 > project_dump.json
```

```
pip freeze > requirements.txt
```

DEPLOY TO HEROKU

https://blizzard-car.herokuapp.com/




```python
import os
import dj_database_url

from pathlib import Path
from decouple import config
```

```python
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
    'whitenoise.middleware.WhiteNoiseMiddleware',
]
```

tests.py | car_detail.html | Procfile

Procfile

koushik, last month | 1 author (koushik)

```
1  release: python manage.py migrate
2  web: gunicorn blizzard.wsgi
```

koushik, last month • procfile and requirements …

```python
# whitenoise settings

STATICFILES_STORAGE = 'whitenoise.storage.CompressedManifestStaticFilesStorage'
```

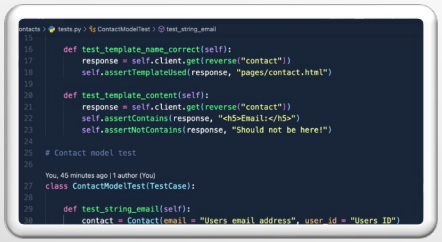
```python
#DATABASES = {'default': dj_database_url.config(default='postgres://postgres:password@lo

DEFAULT_AUTO_FIELD='django.db.models.AutoField'
```

AFTER DEPLOYMENT PROCEDURES ACTIVATED ADMIN IN THE HEROKU SERVER

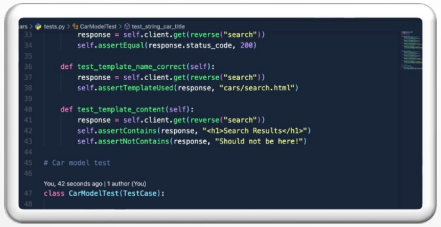
Command : heroku run python manage.py shell

```
>>> from django.contrib.sites.models import site
>>> site = site()
>>> site.domain = 'blizzard-car.herokuapp.com'
>>> site.name = 'blizzard-car.herokuapp.com'
>>> site.save()
>>> print(site.objects.get(name='blizzard-car.herokuapp.com').id)
```

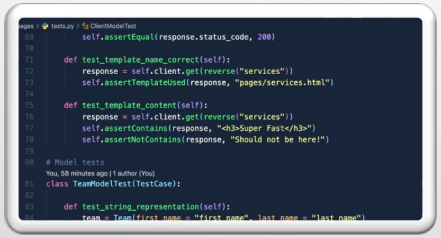
TESTING



```
def test_template_name_correct(self):
    response = self.client.get(reverse("contact"))
    self.assertTemplateUsed(response, "pages/contact.html")

def test_template_content(self):
    response = self.client.get(reverse("contact"))
    self.assertContains(response, "<h5>Email:</h5>")
    self.assertNotContains(response, "Should not be here!")

# Contact model test

You, 45 minutes ago | 1 author (You)
class ContactModelTest(TestCase):

    def test_string_email(self):
        contact = Contact(email = "Users email address", user_id = "Users ID")
```



```
        response = self.client.get(reverse("search"))
        self.assertEqual(response.status_code, 200)

    def test_template_name_correct(self):
        response = self.client.get(reverse("search"))
        self.assertTemplateUsed(response, "cars/search.html")

    def test_template_content(self):
        response = self.client.get(reverse("search"))
        self.assertContains(response, "<h1>Search Results</h1>")
        self.assertNotContains(response, "Should not be here!")

# Car model test

You, 42 seconds ago | 1 author (You)
class CarModelTest(TestCase):
```



```
        self.assertEqual(response.status_code, 200)

    def test_template_name_correct(self):
        response = self.client.get(reverse("services"))
        self.assertTemplateUsed(response, "pages/services.html")

    def test_template_content(self):
        response = self.client.get(reverse("services"))
        self.assertContains(response, "<h3>Super Fast</h3>")
        self.assertNotContains(response, "Should not be here!")

# Model tests
You, 58 minutes ago | 1 author (You)
class TeamModelTest(TestCase):

    def test_string_representation(self):
        team = Team(first_name = "first name", last_name = "last name")
```

- python manage.py test

BLIZZARD AUTO