All right

Save ▾ Share

Your code

```python
from django.apps import AppConfig


class HomeConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'home'

```

All right

Save ▾ Share

Your code

```python
from django.contrib import admin
from .models import Contact
from django_summernote.admin import SummernoteModelAdmin


@admin.register(Contact)
class ContactAdmin(SummernoteModelAdmin):
    list_display = ('name', 'email', 'subject', 'message')
    summernote_fields = ('message',)

```

All right

Save ▾ Share

Your code

```
1  from .models import Contact
2  from django import forms
3
4
5  class ContactForm(forms.ModelForm):
6      """Form for users to contact quteba"""
7      class Meta:
8          """Meta class"""
9          model = Contact
10         fields = [
11             'name',
12             'email',
13             'subject',
14             'message',
15             ]
```

All right

Save ▾ Share

Your code

```python
from django.db import models


class Contact(models.Model):
    name = models.CharField(blank=False, max_length=200)
    email = models.EmailField()
    subject = models.CharField(blank=False, max_length=200)
    message = models.TextField(blank=False)

    def __str__(self):
        return self.name

```

All right

Save ▾ Share

Your code

```python
from . import views
from django.urls import path


urlpatterns = [
    path('', views.index, name='home'),
    path('about/', views.about, name='about'),
    path('search/', views.SearchView.as_view(), name='search_results'),
    path('contact/', views.ContactView.as_view(), name='contact'),
]

```

All right

Save ▾ Share

Your code

```
1  from django.shortcuts import render
2  from django.views import View
3  from django.http import HttpResponseRedirect
4  from django.contrib import messages
5  from django.db.models import Q
6  from qblog.models import Post
7  from qforum.models import Thread
8  from .forms import ContactForm
9
10
11 def index(request):
12     """ Quteba home page view
13      showing most recent three posts and three threads"""
14     recent_posts = Post.objects.filter(status=1).order_by('-created_on')
15     active_topics = Thread.objects.all().order_by('-created_on')
```

All right

Save ▾ Share

Your code

```python
from django.contrib import admin
from .models import Post, Comment
from django_summernote.admin import SummernoteModelAdmin


@admin.register(Post)
class PostAdmin(SummernoteModelAdmin):
    """ Register Post model to admin """
    list_display = ('title', 'slug', 'status', 'created_on')
    search_fields = ['title', 'content']
    list_filter = ('status', 'created_on')
    prepopulated_fields = {'slug': ('title',)}
    summernote_fields = ('content',)


```

All right

Save ▾ Share

Your code

```python
from django.apps import AppConfig


class QblogConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'qblog'

```

All right

Save ▾ Share

Your code

```python
from django import forms
from .models import Comment, Post


class PostForm(forms.ModelForm):
    """ Form to create blog posts """
    class Meta:
        model = Post
        fields = ('author', 'title', 'content', 'excerpt', 'status')

    def __init__(self, *args, **kwargs):
        super(PostForm, self).__init__(*args, **kwargs)
        self.fields['title'].widget.attrs = {'placeholder': 'Title...',
                                             'class': 'form-control',
                                             'rows': '5'}
```

All right

Save ▾ Share

Your code

```python
from django.db import models
from django.utils.text import slugify
from django.contrib.auth.models import User
from cloudinary.models import CloudinaryField


STATUS = ((0, 'Draft'), (1, 'Published'))


class Post(models.Model):
    """ Model for blog posts """
    title = models.CharField(max_length=200, unique=True)
    slug = models.SlugField(max_length=200, unique=True)
    author = models.ForeignKey(
        User, on_delete=models.CASCADE, related_name="blog_posts")
```

All right

Save ▾ Share

Your code

```
1  from . import views
2  from django.urls import path
3
4  app_name = 'qblog'
5
6  urlpatterns = [
7      path('', views.PostList.as_view(), name='blog'),
8      path('create/', views.PostCreateView.as_view(), name='post_create'),
9      path('<slug:slug>/', views.PostDetail.as_view(), name='post_detail'),
10     path('edit/<slug:slug>/', views.PostEditView.as_view(), name='edit_post'),
11     path('like/<slug:slug>', views.PostLike.as_view(), name='post_like'),
12 ]
13
```

All right

Save ▾ Share

Your code

```
1  from django.shortcuts import render, get_object_or_404, reverse
2  from django.urls import reverse_lazy
3  from django.views.generic import ListView, View
4  from django.views.generic.edit import CreateView, UpdateView
5  from django.contrib.auth.mixins import LoginRequiredMixin
6  from django.http import HttpResponseRedirect
7  from .models import Post
8  from .forms import CommentForm, PostForm
9
10
11 class PostCreateView(CreateView):
12     """
13     View to create blog posts
14     """
15     template_name = 'qblog/create_post.html'
```

All right

Save ▾ Share

Your code

```python
from django.contrib import admin
from django_summernote.admin import SummernoteModelAdmin
from .models import Thread, Comment, Category



@admin.register(Thread)
class ThreadAdmin(SummernoteModelAdmin):
    """
    Register Thread model to admin
    """
    list_display = ('name', 'topic', 'slug', 'description', 'category',
                    'created_on')
    search_fields = ['topic', 'description']
    list_filter = ('status', 'created_on')
    prepopulated_fields = {'slug': ('topic',)}
```

All right

Save ▾ Share

Your code

```
1  from django.apps import AppConfig
2
3
4  class QforumConfig(AppConfig):
5      default_auto_field = 'django.db.models.BigAutoField'
6      name = 'qforum'
7
```

All right

Save ▾ Share

Your code

```
1  from django.forms import ModelForm
2  from .models import Thread, Comment
3
4
5  class ThreadForm(ModelForm):
6      """
7      Form for a user to create a discussion forum
8      """
9      class Meta:
10         model = Thread
11         fields = ('category', 'topic', 'description')
12
13     def __init__(self, *args, **kwargs):
14         super(ThreadForm, self).__init__(*args, **kwargs)
15         self.fields['topic'].widget.attrs = {'placeholder': 'Enter a topic...',
```

All right

Save ▾ Share

Your code

```python
from django.db import models
from django.urls import reverse
from django.contrib.auth.models import User
from django.utils.text import slugify


STATUS = ((0, 'Waiting'), (1, 'Approved'))


class Category(models.Model):
    """
    A class to create categories of forums.
    """
    subject = models.CharField(default='Uncategorized', max_length=50)
    description = models.TextField(max_length=255)
```

All right

Save ▾ Share

Your code

```
1  from . import views
2  from django.urls import path
3
4  app_name = 'qforum'
5
6  urlpatterns = [
7      path('', views.ThreadList.as_view(), name='threads'),
8      path('upvote/', views.vote_up, name='upvotes'),
9      path('downvote/', views.vote_down, name='downvotes'),
10     path('like/', views.like_dislike_view, name='comment-like'),
11     path('<slug:slug>/', views.ThreadDetailView.as_view(),
12          name='thread_detail'),
13     path('edit/<slug:slug>/', views.ThreadEditView.as_view(),
14          name='edit-thread'),
15     path('delete/<slug:slug>/', views.ThreadDeleteView.as_view(),
```

All right

Save ▾ Share

Your code

```python
from django.shortcuts import render, get_object_or_404
from django.views import generic, View
from django.views.generic.edit import UpdateView, DeleteView
from django.urls import reverse_lazy
from django.contrib.auth.decorators import login_required
from django.http import JsonResponse
from django.contrib.auth.mixins import UserPassesTestMixin, LoginRequiredMixin
from .models import Thread, Comment, Category
from .forms import ThreadForm, CommentForm


class ThreadList(View):
    """
    A class based view for a list of threads
    """
```

All right

Save ▾ Share

Your code

```
"""
ASGI config for quteba project.

It exposes the ASGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.0/howto/deployment/asgi/
"""

import os

from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'quteba.settings')

```

All right

Save ▾ Share

Your code

```
"""
Django settings for quteba project.

Generated by 'django-admin startproject' using Django 4.0.3.

For more information on this file, see
https://docs.djangoproject.com/en/4.0/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/4.0/ref/settings/
"""
from django.contrib.messages import constants as messages
from pathlib import Path
import os
import sys
```

All right

Save ▾ Share

Your code

```
"""quteba URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/4.0/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
```

All right

Save ▾ Share

Your code

```python
from django.shortcuts import render


def handler404(request, exception):
    """
    Error Handler 404 - Page Not Found
    """
    return render(request, "errors/404.html", status=404)


def handler500(request):
    """
    Error Handler 500 - Internal Server Error
    """
    return render(request, "errors/500.html", status=500)
```

All right

Save ▾ Share

Your code

```
"""
WSGI config for quteba project.

It exposes the WSGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.0/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'quteba.settings')

```

All right

Save ▾ Share

Your code

```python
from django.test import TestCase
from qforum.forms import ThreadForm, CommentForm
from qblog.forms import CommentForm as PostCommentForm
from users.forms import UserRegisterForm, UserUpdateForm, ProfileUpdateForm
from users.models import Profile


class TestThreadForm(TestCase):
    """
    Unittest for thread form
    """
    def test_empty_form(self):
        form = ThreadForm()
        self.assertIn('topic', form.fields)
        self.assertIn('category', form.fields)
```

All right

Save ▾ Share

Your code

```
1  import datetime
2  from django.test import TestCase
3  from django.utils import timezone
4  from django.contrib.auth.models import User
5  from qforum.models import Category, Thread, Comment
6  from qblog.models import Comment as PostComment
7  from qblog.models import Post
8
9
10 class CategoryModelTest(TestCase):
11     """
12     Testing the category model in qforum
13     """
14     @classmethod
15     def setUpTestData(cls):
```

All right

Save ▾ Share

Your code

```
1  import datetime
2  from django.test import TestCase
3  from django.contrib.auth.models import User
4  from django.urls import reverse
5  from django.test import Client
6  from qblog.models import Post
7  from qblog.models import Comment as PostComment
8  from qforum.models import Thread, Category, Comment
9  from qblog.views import Post, PostDetail
10
11
12 class HomePageViewTests(TestCase):
13     """
14     Testing home page view
15     """
```

All right

Save ▾ Share

Your code

```
1  from django.contrib import admin
2  from .models import Profile
3
4  admin.site.register(Profile)
5
```

All right

Save ▾ Share

Your code

```python
from django.apps import AppConfig


class UsersConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'users'

    def ready(self):
        import users.signals

```

All right

Save ▾ Share

Your code

```python
from django import forms
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm
from .models import Profile


class UserRegisterForm(UserCreationForm):
    """
    Form for a new user to sign up
    """
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']
```

All right

Save ▾ Share

Your code

```python
from django.db import models
from django.contrib.auth.models import User
from PIL import Image


class Profile(models.Model):
    """
    User profile model
    """
    user = models.OneToOneField(User,
                                on_delete=models.CASCADE,
                                related_name='user_profile')
    image = models.ImageField(default='default.jpg',
                              upload_to='profile_pics')
    bio = models.TextField(null=True, blank=True)
```

All right

Save ▾ Share

Your code

```python
from django.db.models.signals import post_save
from django.contrib.auth.models import User
from django.dispatch import receiver
from .models import Profile


@receiver(post_save, sender=User)
def create_profile(sender, instance, created, **kwargs):
    if created:
        Profile.objects.create(user=instance)


@receiver(post_save, sender=User)
def save_profile(sender, instance, **kwargs):
    instance.user_profile.save()
```

All right

Save ▾ Share

Your code

```python
from django.contrib.auth import views as auth_views
from django.urls import path
from . import views


urlpatterns = [
    path('', views.profile, name='profile'),
    path('signup/', views.register, name='signup'),
    path('login/', auth_views.LoginView.as_view(
        template_name='account/login.html'),
        name='login'),
    path('logout/', auth_views.LogoutView.as_view(
        template_name='account/logout.html'),
        name='logout'
        ),
```

All right

Save ▾ Share

Your code

```python
from django.shortcuts import render, redirect, reverse, get_object_or_404
from django.http import HttpResponseRedirect
from django.contrib import messages
from django.contrib.auth.decorators import login_required
from django.contrib.auth.models import User
from .models import Profile
from .forms import UserRegisterForm, UserUpdateForm, ProfileUpdateForm


def register(request):
    """ User registration to create account """
    if request.method == 'POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            username = form.cleaned_data.get('username')
```