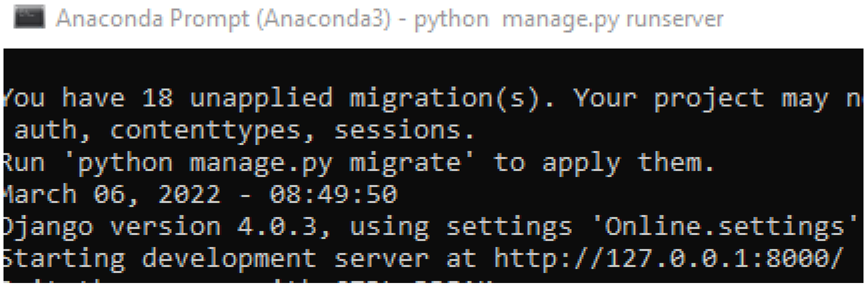
Each time log in Step by Step to code:

1. Start env
   cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts
   activate
2. Display the packages install in the env
   pip freeze
3. Run server to start coding
   cd C:\Users\angel\OneDrive\Desktop\Python Django
   python manage.py runserver
   http://127.0.0.1:8000/



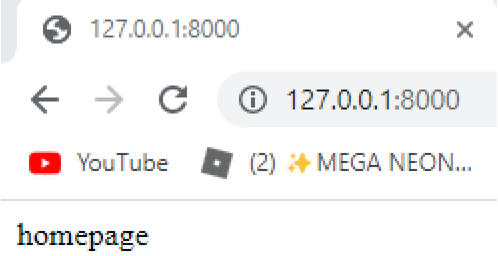
Anaconda Prompt (Anaconda3) - python manage.py runserver

```
You have 18 unapplied migration(s). Your project may n
 auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
March 06, 2022 - 08:49:50
Django version 4.0.3, using settings 'Online.settings'
Starting development server at http://127.0.0.1:8000/
```

python manage.py makemigrations
python manage.py migrate

Website development Step by Step:

1. Find out the path name of the web development folder
   ```
   cd C:\Users\angel\OneDrive\Desktop\Python Django
   ```
2. Installing virtualenv
   ```
   py –m pip install –-user virtualenv
   ```
3. Creating a virtual environment folder
   ```
   py –m venv env
   ```
4. Pointing to the activate the env folder
   ```
   cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts
   activate
   ```
5. Install Django package
   ```
   pip install django
   ```
6. Display the packages install in the env
   ```
   pip freeze
   ```
7. create project folder
   ```
   cd C:\Users\angel\OneDrive\Desktop\Python Django
   django-admin startproject Online .
   ```
8. Python start web site server
   ```
   python manage.py runserver
   http://127.0.0.1:8000/
   ```
9. Create a views.py in the online folder together with urls.py
10. Web Testing thru HTTP Response

    urls.py
    ```
    from django.contrib import admin
    from django.urls import path
    from . import views

    urlpatterns = [
       path('admin/', admin.site.urls),
       path('',views.home, name='home'),
    ]
    ```
    Views.py
    ```
    from django.http import HttpResponse

    def home(request):
       return HttpResponse('homepage')
    ```
    

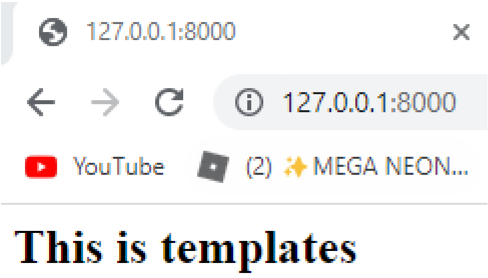
    homepage
11. Create templates folder in the project folder
12. Create home.html in the templates folder
    ```
    <h2>This is templates</h2>
    ```

13. Add templates folder name in the setting.py file in the project folder

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': ['templates'],
```

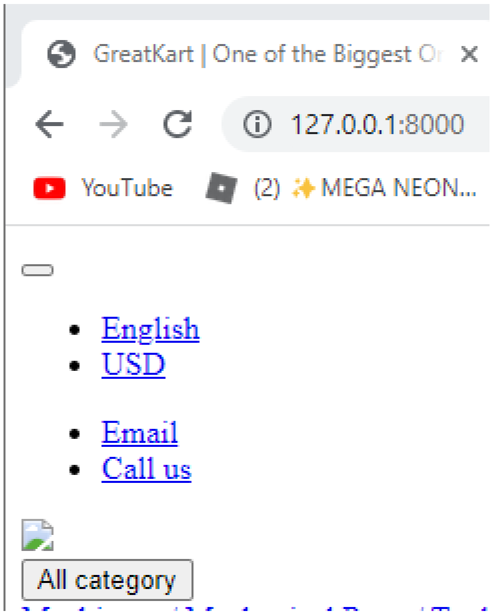
14. Adjust the view filer from HttpResponse to Render

```
#from django.http import HttpResponse
from django.shortcuts import render

def home(request):
    #return HttpResponse('homepage')
    return render(request, 'home.html')
```



This is templates

15. Copy and paste the index.html codes into home.html in the templates folder

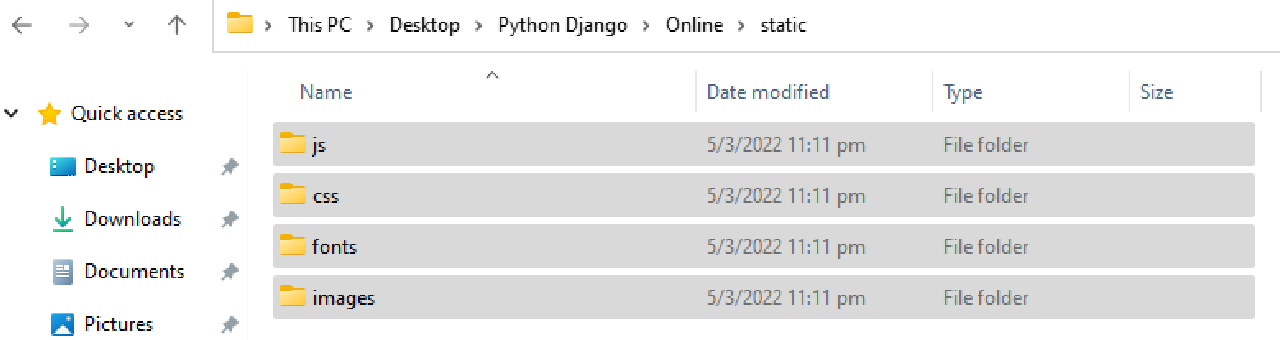


- English
- USD

- Email
- Call us

All category

16. Create static folder in the Online folder and copy and paste the 4 folders into the static folder



Name	Date modified	Type	Size
js	5/3/2022 11:11 pm	File folder	
css	5/3/2022 11:11 pm	File folder	
fonts	5/3/2022 11:11 pm	File folder	
images	5/3/2022 11:11 pm	File folder	

17. Adjust settings/py to prepare the linkage with pictures with the website from images folder

```
STATIC_URL = '/static/'
```

```
STATIC_ROOT = BASE_DIR /'static'
STATICFILES_DIRS =[
    'online/static',
]
```

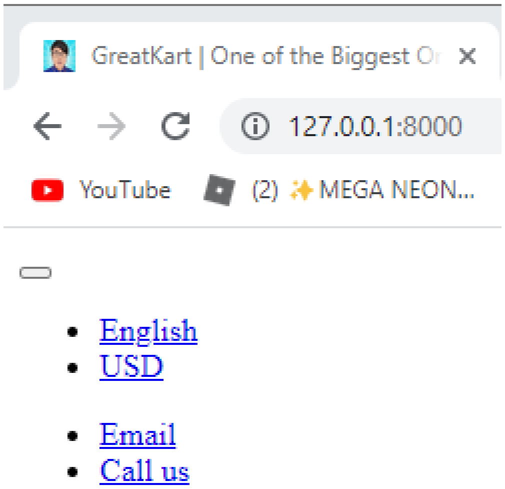
18. Change website logo with codes adjustment on home.html

    Insert 1 sentence in the 1st line in home.html

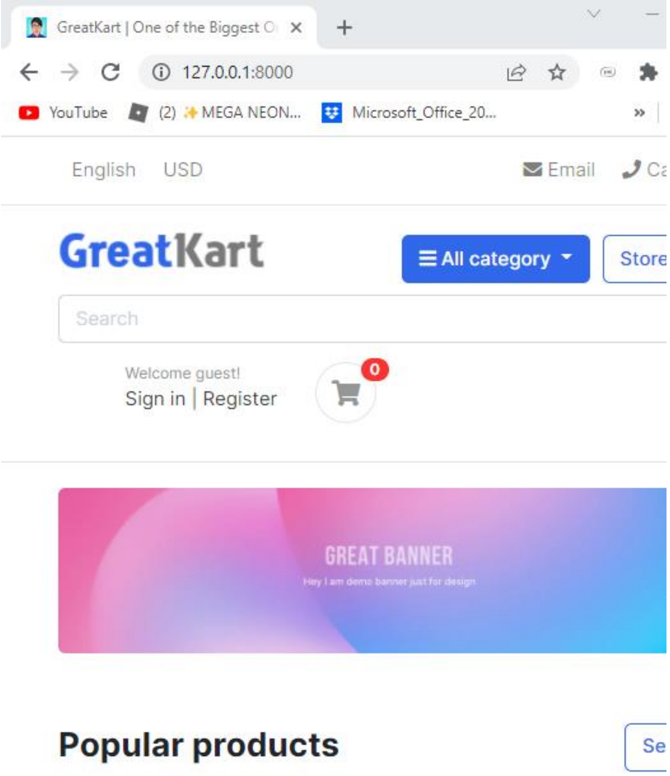
    ```
    {% load static %}
    <!DOCTYPE HTML>
    <html lang="en">
    ```

    Replace old code with new codes

    ```
    <link href="images/favicon.ico" rel="shortcut icon" type="image/x-icon">

    <link href= {% static 'images/favicon.ico'%} rel="shortcut icon" type="image/x-icon">
    ```



19. Repeat the replacement process on js, css png, jpg



20. Create base.html in the templates folder
21. Copy the 1st line to </header> <!-- section-header.// --> from home.html to base.html
22. Add lines in base.html

    1st line

    ```
    {% load static %}
    <!DOCTYPE HTML>
    ```

    Last few lines

    ```
    </header> <!-- section-header.// -->
    {% block contents %}
    <!-- content -->
    {% endblock %}
    ```

23. Add lines in the homt.html

    1st 3 lines

    ```
    {% extends 'base.html' %}
    {% load static %}}
    {% block contents %}
    <!-- ========================= SECTION MAIN ========================= -->
    <section class="section-intro padding-y-sm">
    ```

    Last 1 line

    ```
    </html>
    {% endblock %}
    ```

24. Create includes folder in the templates folder
25. Create footer.html & navbar.html in the includes folder
26. Copy the footer codes from home.html to footer.html

    ```
    <!-- ========================= FOOTER ========================= -->
    <footer class="section-footer border-top">
        <div class="container">
                <section class="footer-bottom border-top row">
                        <div class="col-md-2">
                                <p class="text-muted"> &copy 2019 Company name </p>
                        </div>
                        <div class="col-md-8 text-md-center">
                                <span  class="px-2">info@pixsellz.io</span>
                                <span  class="px-2">+879-332-9375</span>
                                <span  class="px-2">Street name 123, Avanue abc</span>
                        </div>
                        <div class="col-md-2 text-md-right text-muted">
                                <i class="fab fa-lg fa-cc-visa"></i>
                                <i class="fab fa-lg fa-cc-paypal"></i>
                                <i class="fab fa-lg fa-cc-mastercard"></i>
                        </div>
                </section>
        </div><!-- //container -->
    </footer>
    <!-- ========================= FOOTER END // ========================= -->
    ```

27. Copy header sets of codes from base.html to navbar.html

28. Add lines in the base.html at bottom of the codes

```
<body>
<!-- navbar -->
{% include 'includes/navbar.html' %}
{% block contents %}
<!-- content -->
{% endblock %}
<!-- footer -->
{% include 'includes/footer.html' %}
```

29. Add 1 line at navbar.html

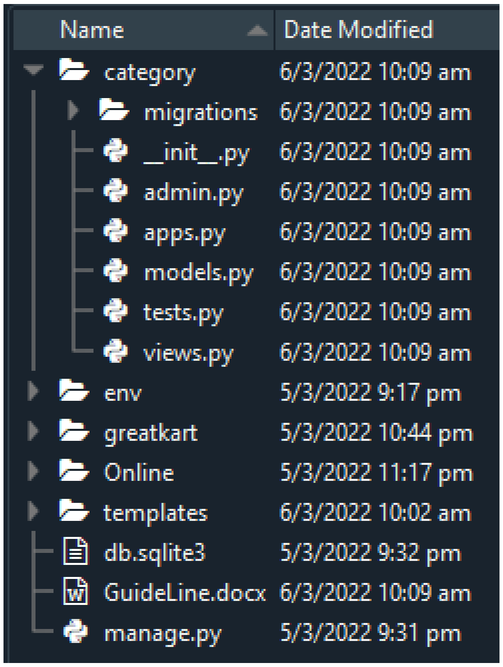
```
{% load static %}
```

30. Add 1 line in the foorer.html

```
{% load static %}
```

31. Create an category folder in the online folder

```
python manage.py startapp category
```



Name	Date Modified
category	6/3/2022 10:09 am
migrations	6/3/2022 10:09 am
__init__.py	6/3/2022 10:09 am
admin.py	6/3/2022 10:09 am
apps.py	6/3/2022 10:09 am
models.py	6/3/2022 10:09 am
tests.py	6/3/2022 10:09 am
views.py	6/3/2022 10:09 am
env	5/3/2022 9:17 pm
greatkart	5/3/2022 10:44 pm
Online	5/3/2022 11:17 pm
templates	6/3/2022 10:02 am
db.sqlite3	5/3/2022 9:32 pm
GuideLine.docx	6/3/2022 10:09 am
manage.py	5/3/2022 9:31 pm

32. Register the category folder in the settings.py under online folder

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'category',
]
```

33. Create Models and Admin code for migration preparation

models.py

```
from django.db import models
# Create your models here.
class Category(models.Model):
    category_name = models.CharField(max_length=50, unique = True)
    slug = models.CharField(max_length=100, unique=True)
    descprtion = models.TextField(max_length=255, blank=True)
    cat_image = models.ImageField(upload_to='photos/categories', blank=True)
    def __str__(self):
        return self.category_name
```

admin.py

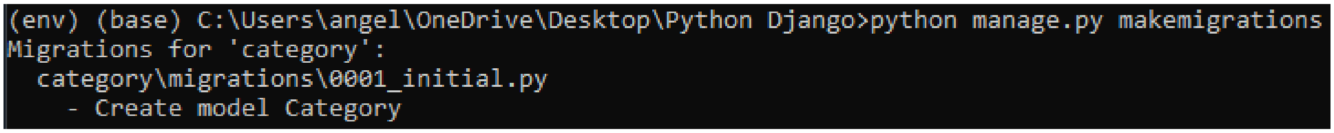
```
from django.contrib import admin
from .models import Category
# Register your models here.
admin.site.register(Category)
```

34. Install pillow package to use ImageField (Cannot use ImageField because Pillow is not installed.)

    Pip install pillow

35. Type command in the Anaconda Prompt (Anaconda3) to create migrations folder

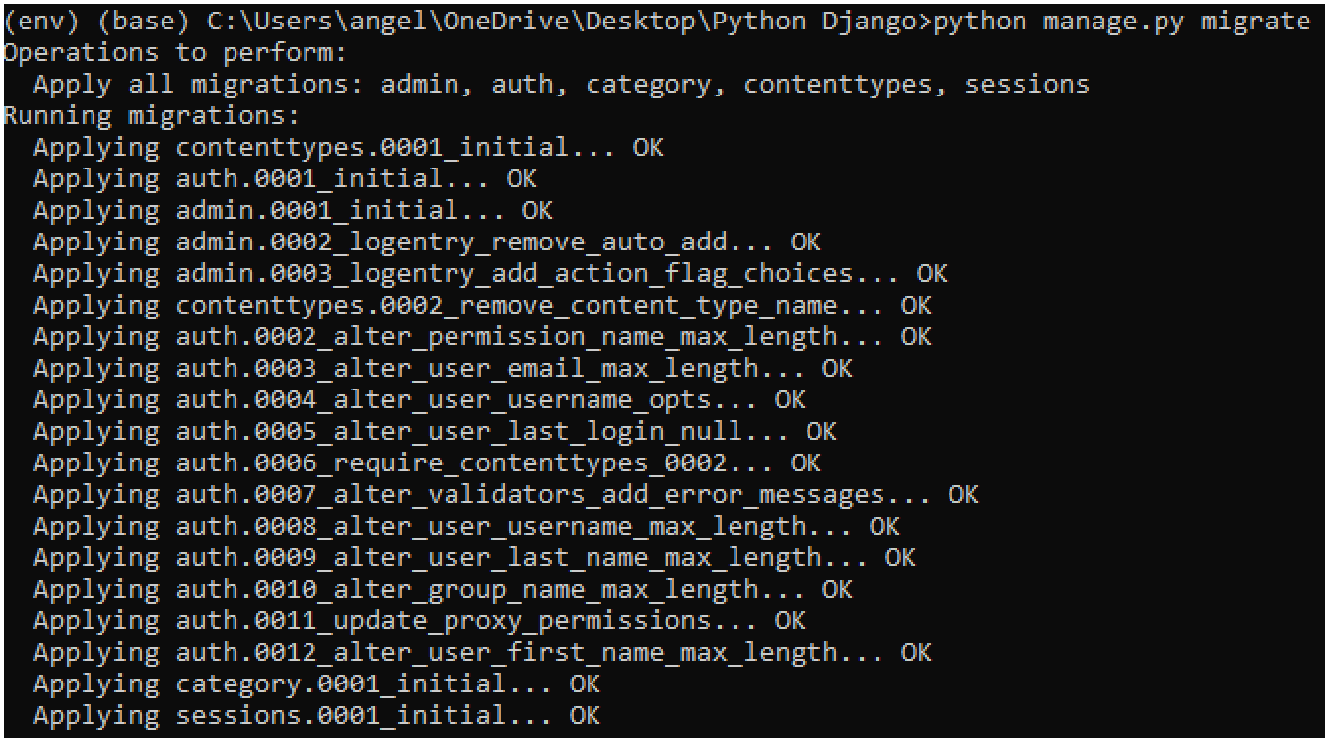
    python manage.py makemigrations



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py makemigrations
Migrations for 'category':
  category\migrations\0001_initial.py
    - Create model Category
```

36. Python run the migrate files 0001_initial.py

    python manage.py migrate



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, category, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying category.0001_initial... OK
  Applying sessions.0001_initial... OK
```

37. Create a super user

    python manage.py createsuperuser

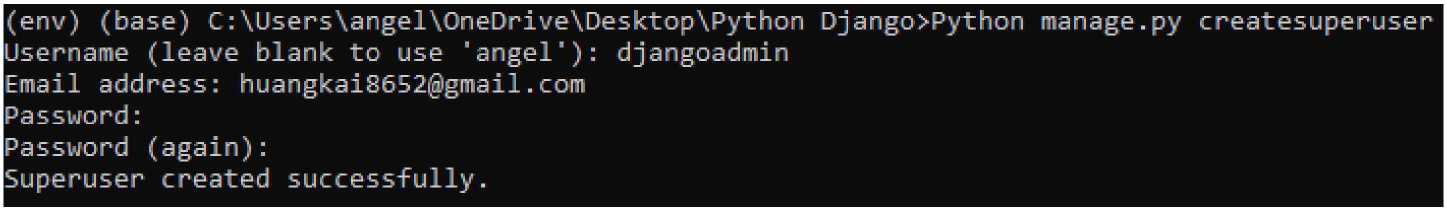
38. Create ID and Password for super user

    Username: djangoadmin

    Email: huangkai8652@gamil.com

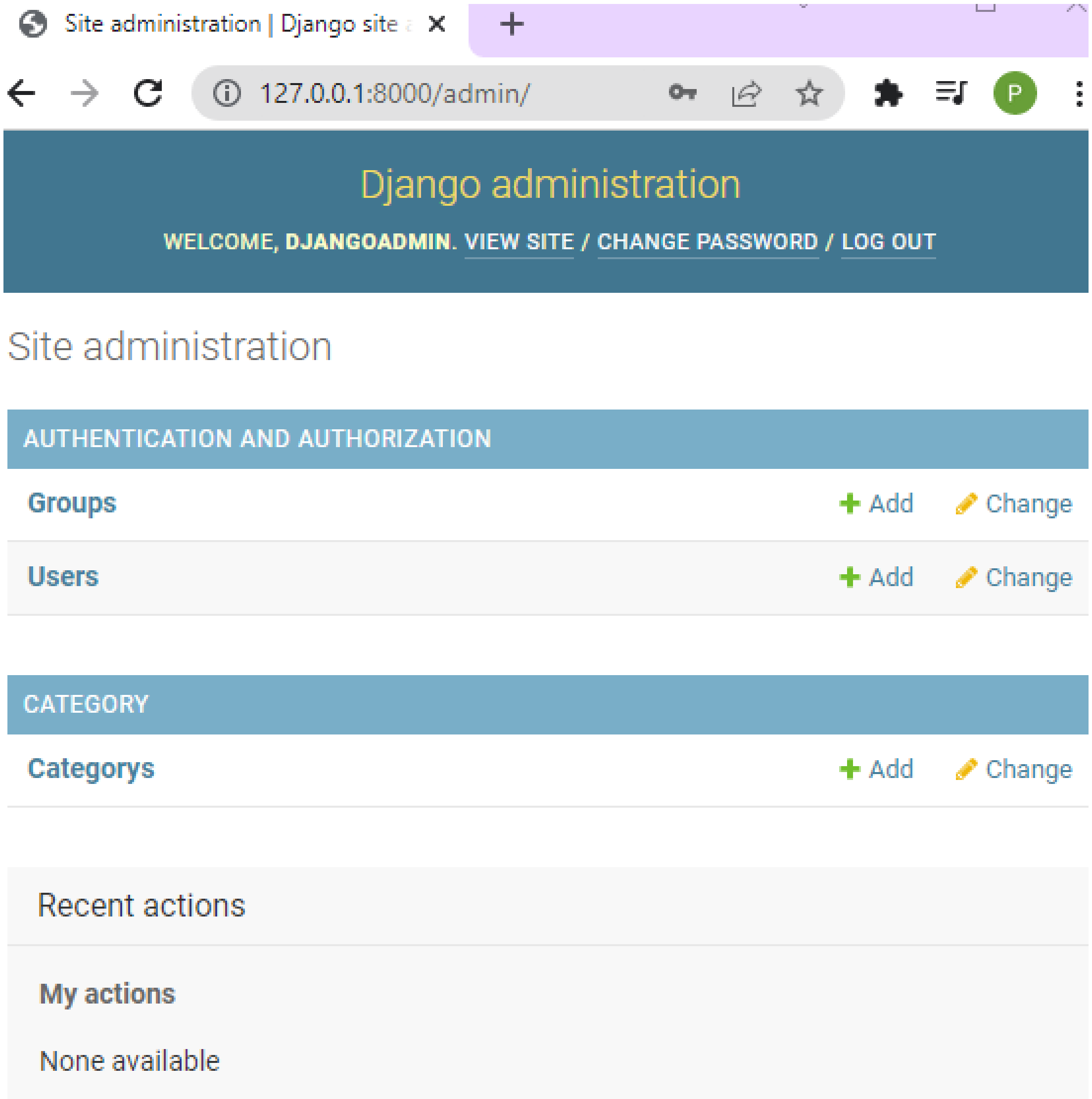
    Password: !q2w3e4R

    Password: !q2w3e4R



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>Python manage.py createsuperuser
Username (leave blank to use 'angel'): djangoadmin
Email address: huangkai8652@gmail.com
Password:
Password (again):
Superuser created successfully.
```

Django Admin Page (Logged In)



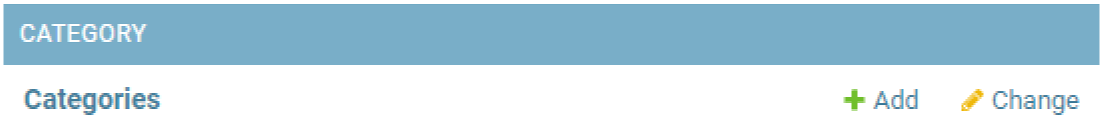
39. Edit the Categorys as above picture

    Add lines in the models.py

```
  cat_image = models.ImageField(upload_to='photos/categories', blank=True)
  class Meta:
      verbose_name = 'Category'
      verbose_name_plural = 'Categories'
  def __str__(self):
```

40. Run migration command again

    python manage.py makemigrations



41. Create account folder for email log in

```
python manage.py startapp accounts
```

42. Create linkage with settings.py in the online folder

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'category',
    'accounts',
]
```

43. Add lines of code in the models.py in the accounts folder

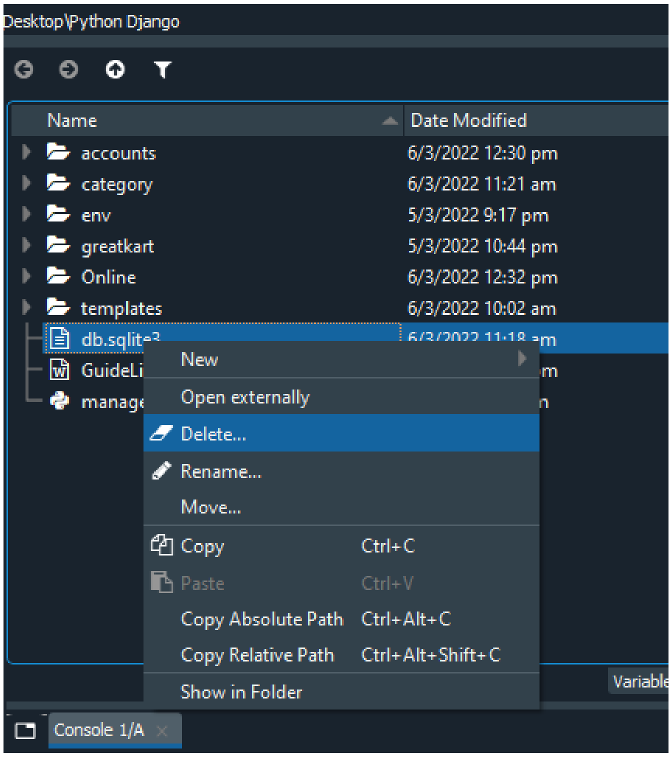
```
from django.db import models
from django.contrib.auth.models import AbstractBaseUser, BaseUserManager
# Create your models here.
class MyAccountManager(BaseUserManager):
    def create_user(self, first_name, last_name, username, email, password=None):
        if not email:
            raise ValueError('User must have an email address')
        if not username:
            raise ValueError('User must have an username')
        user = self.model(
            email = self.normalize_email(email),
            username = username,
            first_name = first_name,
            last_name = last_name,
        )
        user.set_password(password)
        user.save(using=self._db)
        return user
    def create_superuser(self, first_name, last_name, email, username, password):
        user = self.create_user(
            email = self.normalize_email(email),
            username = username,
            password = password,
            first_name = first_name,
            last_name = last_name,
        )
        user.is_admin = True
        user.is_active = True
        user.is_staff = True
        user.is_superadmin = True
        user.save(using=self._db)
        return user
class Account(AbstractBaseUser):
```

```
first_name      = models.CharField(max_length=50)
last_name       = models.CharField(max_length=50)
username        = models.CharField(max_length=50, unique=True)
email           = models.EmailField(max_length=100, unique=True)
phone_number    = models.CharField(max_length=50)
# required
date_joined     = models.DateTimeField(auto_now_add=True)
last_login      = models.DateTimeField(auto_now_add=True)
is_admin        = models.BooleanField(default=False)
is_staff        = models.BooleanField(default=False)
is_active       = models.BooleanField(default=False)
is_superadmin       = models.BooleanField(default=False)
USERNAME_FIELD = 'email'
REQUIRED_FIELDS = ['username', 'first_name', 'last_name']
objects = MyAccountManager()
def full_name(self):
   return f'{self.first_name} {self.last_name}'
def __str__(self):
   return self.email
def has_perm(self, perm, obj=None):
   return self.is_admin
def has_module_perms(self, add_label):
   return True
```

44. Add links in the settings.py file

```
WSGI_APPLICATION = 'Online.wsgi.application'
AUTH_USER_MODEL = 'accounts.Account'
# Database
```

45. Delete the database db.sqlite3 in the online folder



Desktop\Python Django

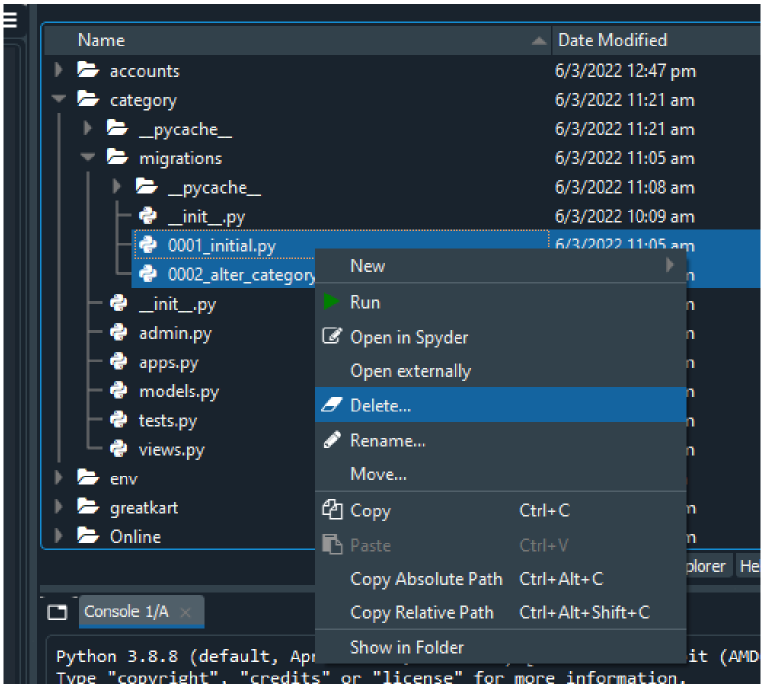
Name	Date Modified
accounts	6/3/2022 12:30 pm
category	6/3/2022 11:21 am
env	5/3/2022 9:17 pm
greatkart	5/3/2022 10:44 pm
Online	6/3/2022 12:32 pm
templates	6/3/2022 10:02 am
db.sqlite3	6/3/2022 11:18 am
GuideLi	m
manage	n

46. Add lines in the admin.py in the accoutns folder

    from django.contrib import admin
    from .models import Account

    # Register your models here.
    admin.site.register(Account)

47. Delete the 0001_initial.py & 0002_alter_category_options.py in the migrations folder.



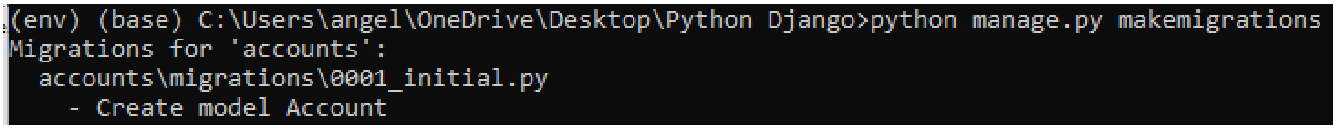
```
Python 3.8.8 (default, Apr                              it (AMD
Type "copyright", "credits" or "license" for more information.
```

48. Create a db.sqlite3 in the online folder

    python manage.py runserver

49. Migration folder again for the 0001_inital.py

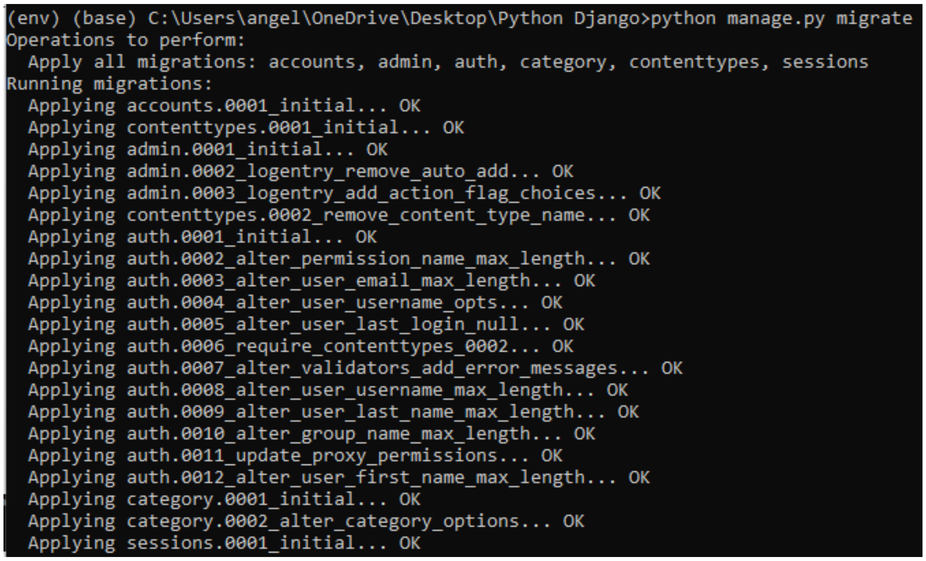
    python manage.py makemigrations



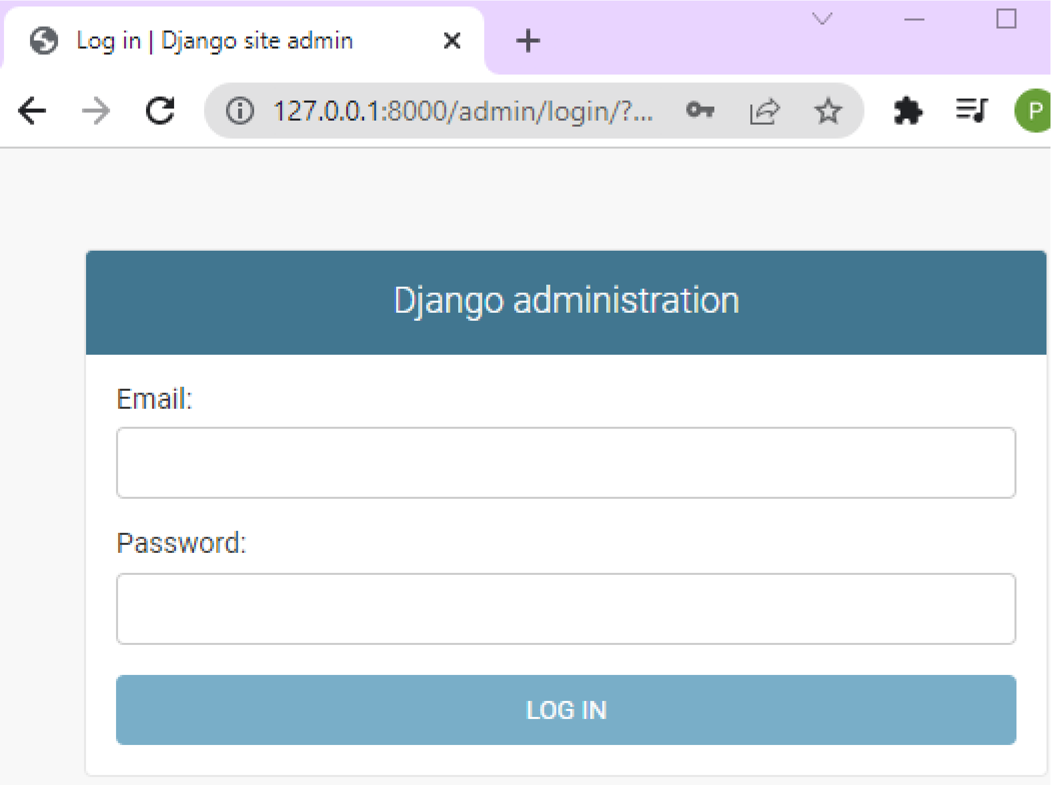
```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py makemigrations
Migrations for 'accounts':
  accounts\migrations\0001_initial.py
    - Create model Account
```

50. Migrate the folder

    python manage.py migrate



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py migrate
Operations to perform:
  Apply all migrations: accounts, admin, auth, category, contenttypes, sessions
Running migrations:
  Applying accounts.0001_initial... OK
  Applying contenttypes.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying category.0001_initial... OK
  Applying category.0002_alter_category_options... OK
  Applying sessions.0001_initial... OK
```



51. Create super user

    Email: huangkai8652@gmail.com

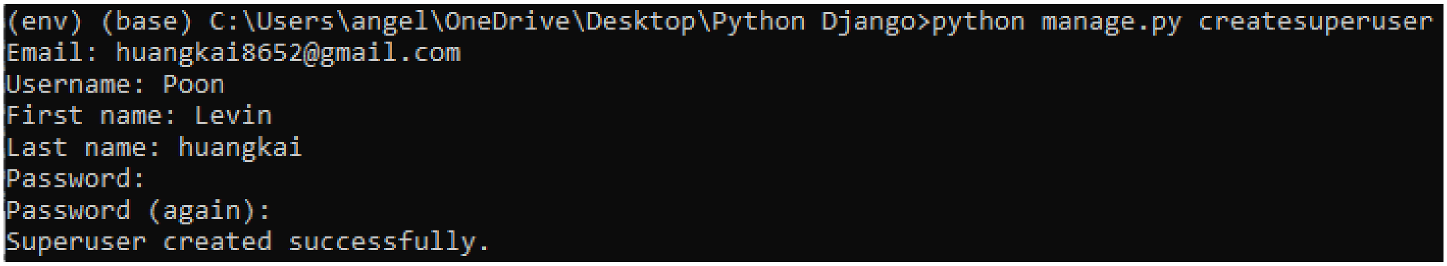
    Username: Poon

    First name: Levin

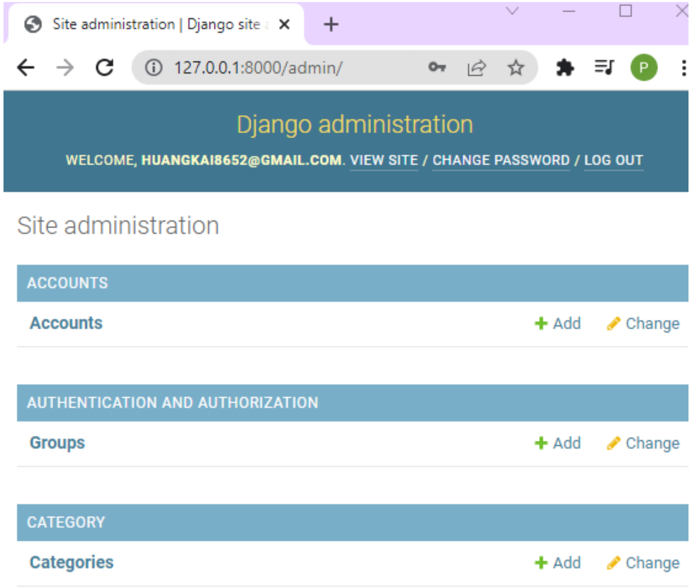
    Last name: huangkai

    Password: !q2w3e4R

    Password (again): !q2w3e4R



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py createsuperuser
Email: huangkai8652@gmail.com
Username: Poon
First name: Levin
Last name: huangkai
Password:
Password (again):
Superuser created successfully.
```

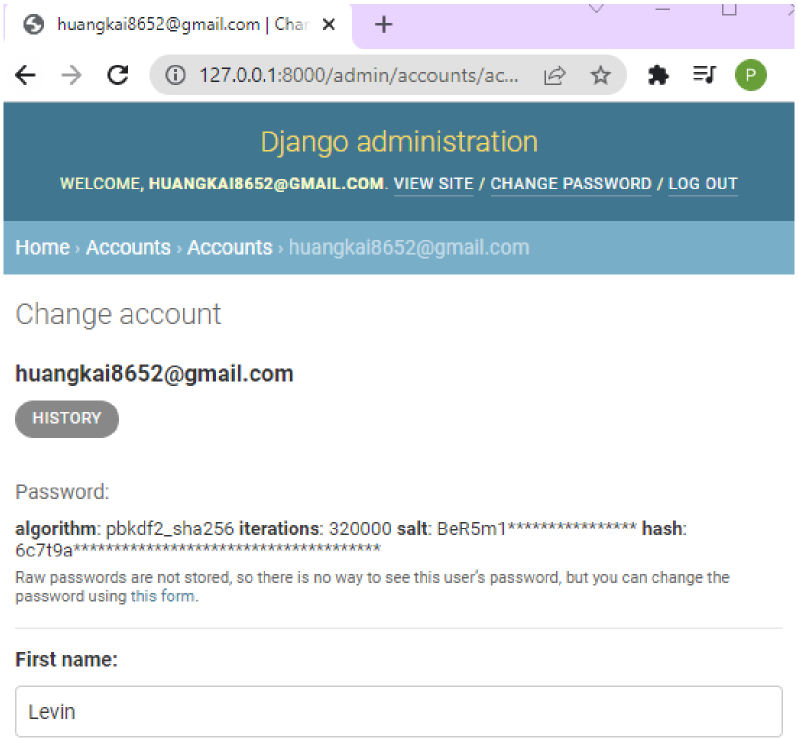

Site administration | Django site
127.0.0.1:8000/admin/
P
Django administration
WELCOME, HUANGKAI8652@GMAIL.COM. VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
ACCOUNTS
Accounts
Add
Change
AUTHENTICATION AND AUTHORIZATION
Groups
Add
Change
CATEGORY
Categories
Add
Change

User able to log in thru email instead of username

52. Change password view to read only, create linkage between first name and last name to profile and seem last login timing

    Add lines in admin.py under accounts

```
from django.contrib import admin
from django.contrib.auth.admin import UserAdmin
from .models import Account
# Register your models here.
class AccountAdmin(UserAdmin):
    list_display = ('email', 'first_name', 'last_name', 'username', 'last_login', 'date_joined', 'is_active')
    list_display_links = ('email', 'first_name', 'last_name')
    readonly_fields = ('last_login', 'date_joined')
    ordering = ('-date_joined',)
    filter_horizontal = ()
    list_filter = ()
    fieldsets = ()
admin.site.register(Account, AccountAdmin)
```


huangkai8652@gmail.com | Char
127.0.0.1:8000/admin/accounts/ac...
P

Django administration

WELCOME, **HUANGKAI8652@GMAIL.COM**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Accounts › Accounts › huangkai8652@gmail.com

Change account

huangkai8652@gmail.com

HISTORY

Password:

algorithm: pbkdf2_sha256 **iterations**: 320000 **salt**: BeR5m1**************** **hash**: 6c7t9a**************************************

Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form.

First name:

Levin

53. Add lines in the settings.py in the online folder (Configuring Media Files)

```
STATICFILES_DIRS =[
   'online/static',
]
# media files configuration
MEDIA_URL = '/media/'
MEDIA_ROOT = BASE_DIR /'media'
```

54. Add lines in the urls.py in the online folder

```
from . import views
from django.conf.urls.static import static
from django.conf import settings
urlpatterns = [
   path('admin/', admin.site.urls),
   path('',views.home, name='home'),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

55. Save information on the django admin website under category

    Category name: Shirts

    Slug: shirts

    Descriptions: This is the demo shirt category

    Cat Image: shirts.jpg

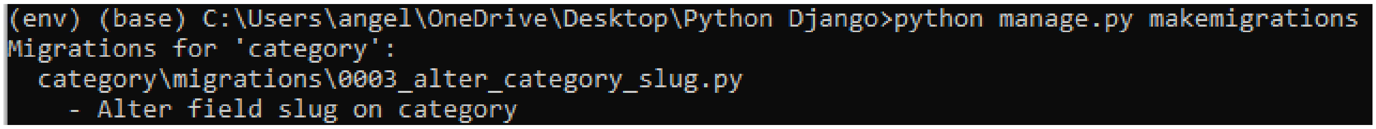
56. Slug name will auto copy from Category name

    slug = models.CharField(max_length=100, unique=True) change to =>

    slug = models.SlugField(max_length=100, unique=True)

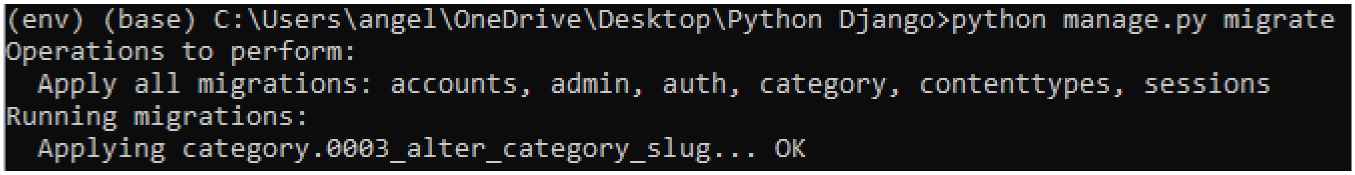
57. Once the model in the category completed, user have to make migrate and migrate in CMD

    python manage.py makemigrations



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py makemigrations
Migrations for 'category':
  category\migrations\0003_alter_category_slug.py
    - Alter field slug on category
```

python manage.py migrate



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py migrate
Operations to perform:
  Apply all migrations: accounts, admin, auth, category, contenttypes, sessions
Running migrations:
  Applying category.0003_alter_category_slug... OK
```

58. Add lines in the admin.py under the category folder

```
# Register your models here.
class CategoryAdmin(admin.ModelAdmin):
    prepopulated_fields = {'slug':('category_name',)}
    list_display = ('category_name','slug')
admin.site.register(Category,CategoryAdmin)
```

59. Add new item of t shirt and slug will auto populate

    Category name: T shirt

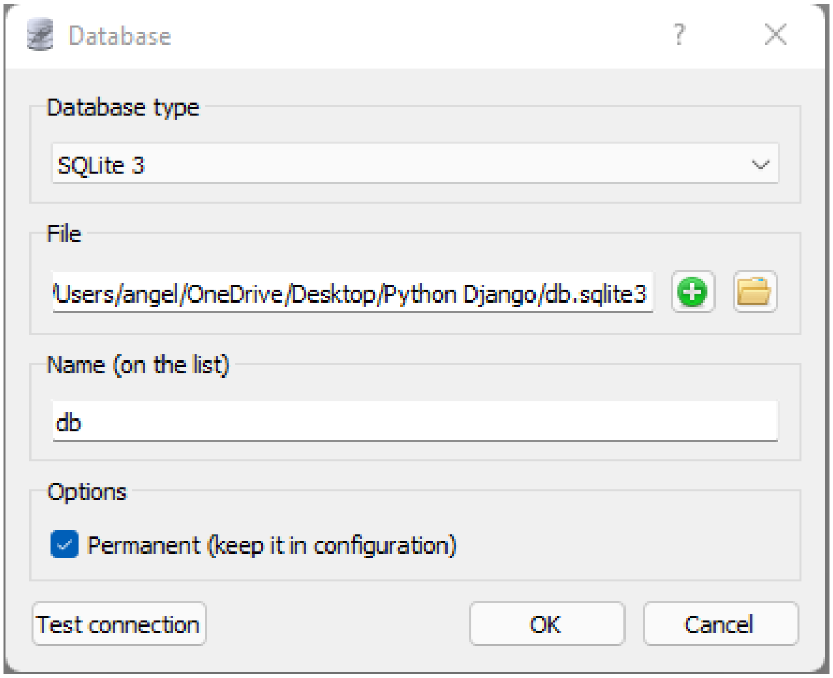
    Slug: t-shirt

    Descriptions: This is the test description

    Cat Image: tshirts.jpg

    Add the rest 4 category into the Django admin

60. The database is in the db.sqlite3 and download sqlitestudio from Goggle to view in human readable format

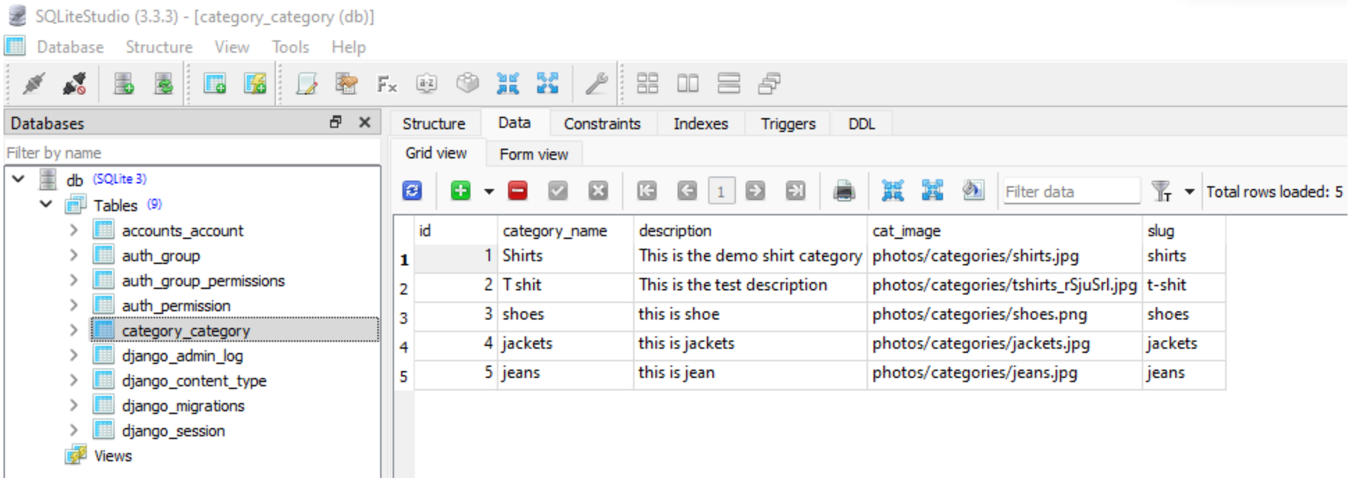


Database type: SQLite 3

File: Users/angel/OneDrive/Desktop/Python Django/db.sqlite3

Name (on the list): db

Options: ☑ Permanent (keep it in configuration)



	id	category_name	description	cat_image	slug
1	1	Shirts	This is the demo shirt category	photos/categories/shirts.jpg	shirts
2	2	T shit	This is the test description	photos/categories/tshirts_rSjuSrl.jpg	t-shit
3	3	shoes	this is shoe	photos/categories/shoes.png	shoes
4	4	jackets	this is jackets	photos/categories/jackets.jpg	jackets
5	5	jeans	this is jean	photos/categories/jeans.jpg	jeans

61. Crate store folder in the online folder

```
Python manage.py startapp store
```

62. Add store in the setting.py under online folder

```
    'category',
    'accounts',
    'store',
]
```

63. Add line in models.py under store folder

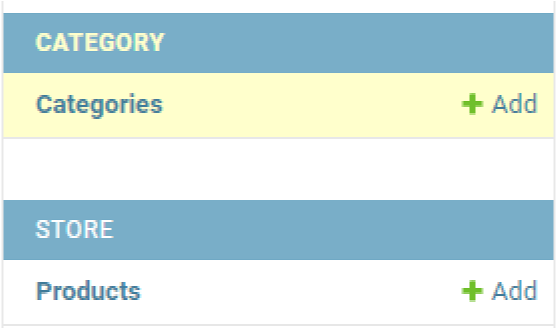
```
from django.db import models
from category.models import Category
# Create your models here.
class Product(models.Model):
    product_name    = models.CharField(max_length=200, unique=True)
    slug            = models.SlugField(max_length=200, unique=True)
    description     = models.TextField(max_length=500, blank=True)
    price           = models.IntegerField()
    images          = models.ImageField(upload_to='photos/products')
    stock           = models.IntegerField()
    is_available    = models.BooleanField(default=True)
    category        = models.ForeignKey(Category, on_delete=models.CASCADE)
    created_date    = models.DateTimeField(auto_now_add=True)
    modified_date   = models.DateTimeField(auto_now=True)

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

64. Add lines in the admin.py under the store folder

```
from django.contrib import admin
from .models import Product
# Register your models here.
class ProductAdmin(admin.ModelAdmin):
    list__display = ('product_name','price','stock','category','modified_date','is_available')
    prepopluated_fields = {'slug': ('product_name',)}
admin.site.register(Product, ProductAdmin)
```

```
python manage.py makemigrations
python manage.py migrate
```

User able to see product appear on the admin page



65. Add all products under each category

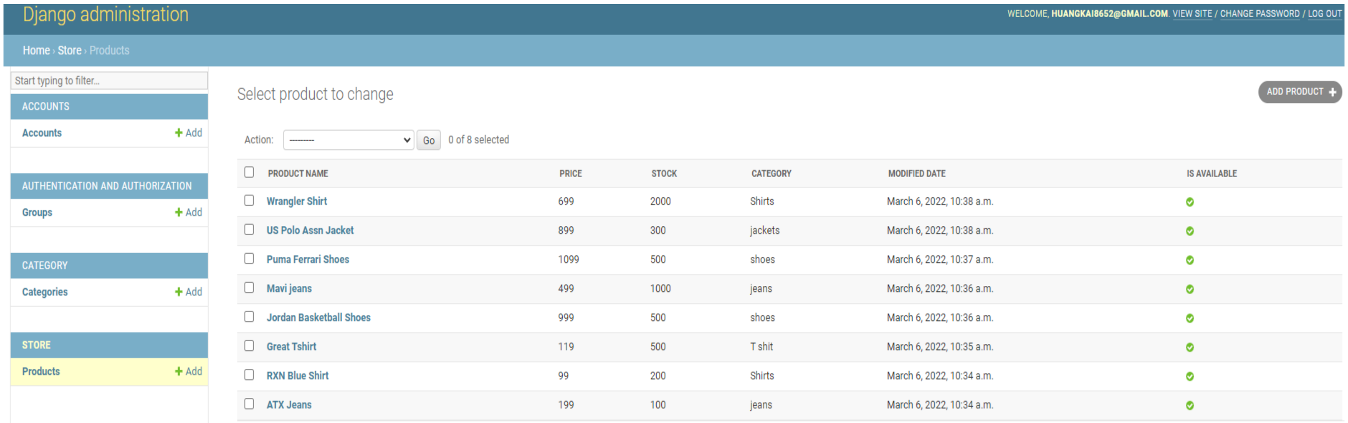
Product name:

Slug:

Description:

Price:

Images:



Select product to change

0 of 8 selected

☐	PRODUCT NAME	PRICE	STOCK	CATEGORY	MODIFIED DATE	IS AVAILABLE
☐	Wrangler Shirt	699	2000	Shirts	March 6, 2022, 10:38 a.m.	✓
☐	US Polo Assn Jacket	899	300	jackets	March 6, 2022, 10:38 a.m.	✓
☐	Puma Ferrari Shoes	1099	500	shoes	March 6, 2022, 10:37 a.m.	✓
☐	Mavi jeans	499	1000	jeans	March 6, 2022, 10:36 a.m.	✓
☐	Jordan Basketball Shoes	999	500	shoes	March 6, 2022, 10:36 a.m.	✓
☐	Great Tshirt	119	500	T shit	March 6, 2022, 10:35 a.m.	✓
☐	RXN Blue Shirt	99	200	Shirts	March 6, 2022, 10:34 a.m.	✓
☐	ATX Jeans	199	100	jeans	March 6, 2022, 10:34 a.m.	✓

66. Add lines in the view.py in the online folder to display the products from database admin page

```
#from django.http import HttpResponse
from django.shortcuts import render
from store.models import Product
def home(request):
    #return HttpResponse('homepage')
    products = Product.objects.all().filter(is_available = True)
    context = {
        'products': products,
    }
    return render(request, 'home.html', context)
```

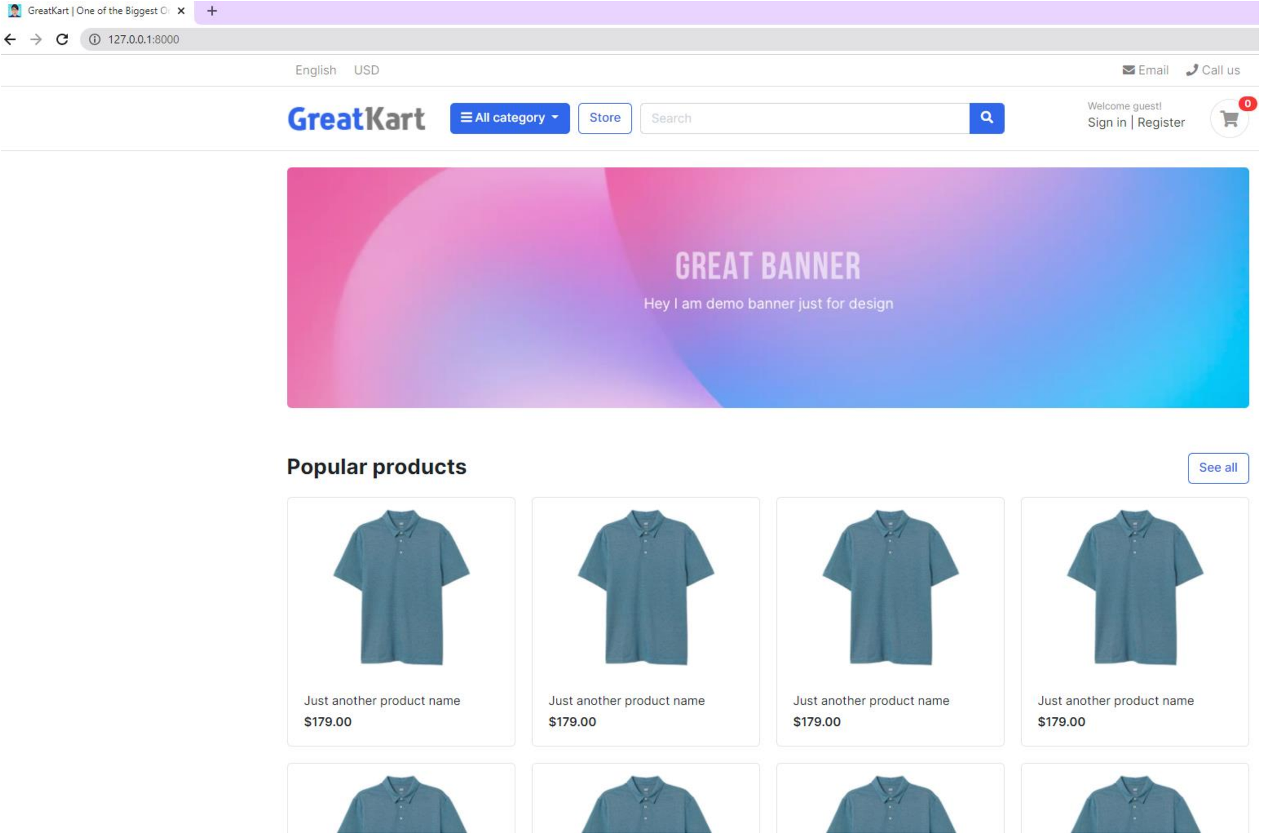
67. Delete the static image from 2 to 9 in the home.html from templates folder

{% static 'images/items/2.jpg' %} to {% static 'images/items/9.jpg' %}

```
<!-- ========================= SECTION  ========================= -->
<section class="section-name padding-y-sm">
<div class="container">
<header class="section-heading">
    <a href="./store.html" class="btn btn-outline-primary float-right">See all</a>
    <h3 class="section-title">Popular products</h3>
</header><!-- sect-heading -->
<div class="row">
    <div class="col-md-3">
            <div class="card card-product-grid">
                    <a href="./product-detail.html" class="img-wrap"> <img src={% static 'images/items/1.jpg' %}> </a>
                    <figcaption class="info-wrap">
                            <a href="./product-detail.html" class="title">Just another product name</a>
                            <div class="price mt-1">$179.00</div> <!-- price-wrap.// -->
                    </figcaption>
            </div>
    </div> <!-- col.// -->
</div> <!-- row.// -->
</div><!-- container // -->
</section>
<!-- ========================= SECTION  END// ========================= -->
```

68. Add for loop to display the 8 products (we have insert manually at step 65)

```
<div class="row">
  {% for product in products %}
    <div class="col-md-3">
            <div class="card card-product-grid">
                    <a href="./product-detail.html" class="img-wrap"> <img src={% static 'images/items/1.jpg' %}> </a>
                    <figcaption class="info-wrap">
                            <a href="./product-detail.html" class="title">Just another product name</a>
                            <div class="price mt-1">$179.00</div> <!-- price-wrap.// -->
                    </figcaption>
            </div>
    </div> <!-- col.// -->
    {% endfor %}
</div> <!-- row.// -->
```


GreatKart | One of the Biggest O
127.0.0.1:8000
English
USD
Email
Call us
GreatKart
All category
Store
Search
Welcome guest!
Sign in | Register
0
GREAT BANNER
Hey I am demo banner just for design
Popular products
See all
Just another product name
$179.00
Just another product name
$179.00
Just another product name
$179.00
Just another product name
$179.00

69. Adjust the link to display the correct images, price and product name

```
<div class="row">
  {% for product in products %}
    <div class="col-md-3">
            <div class="card card-product-grid">
                    <a href="./product-detail.html" class="img-wrap"> <img src="{{ product.images.url }}"> </a>
                    <figcaption class="info-wrap">
                            <a href="./product-detail.html" class="title">{{product.product_name}}</a>
                            <div class="price mt-1">${{product.price}}</div> <!-- price-wrap.// -->
                    </figcaption>
            </div>
    </div> <!-- col.// -->
    {% endfor %}
</div> <!-- row.// -->
</div><!-- container // -->
</section>
<!-- ========================= SECTION  END// ========================= -->
```

70. Copy urls.py from online folder to store folder (making store button working)

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.store, name='store'),
]
```

71. Add lines in urls.py in the online folder

```
from django.contrib import admin
from django.urls import path, include
from . import views
from django.conf.urls.static import static
from django.conf import settings
urlpatterns = [
    path('admin/', admin.site.urls),
    path('',views.home, name='home'),
    path('store/', include('store.urls')),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

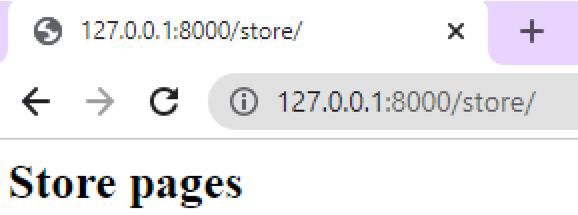
72. Add lines in views.py under store folder

```
from django.shortcuts import render
# Create your views here.
def store(request):
    return render(request, 'store/store.html')
```

73. Chang the link at home.html in the template folder

```
<!-- ========================= SECTION  ========================= -->
<section class="section-name padding-y-sm">
<div class="container">
<header class="section-heading">
     <a href="{% url 'store' %}" class="btn btn-outline-primary float-right">See all</a>
     <h3 class="section-title">Popular products</h3>
</header><!-- sect-heading -->
```

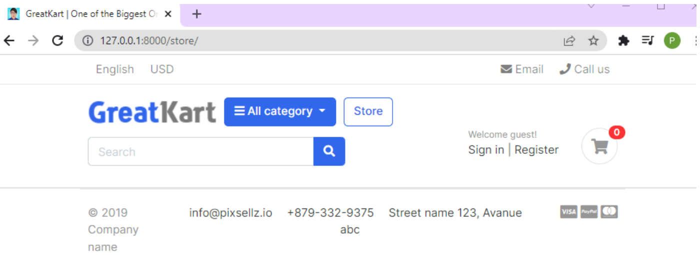
74. Create store.html in the store folder in the template folder and refresh the website



Store pages

75. Replace codes in the store.html under the store folder under templates folder

```
{% extends 'base.html' %}
{% block content %}
{% endblock %}
```



© 2019 Company name

info@pixsellz.io

+879-332-9375

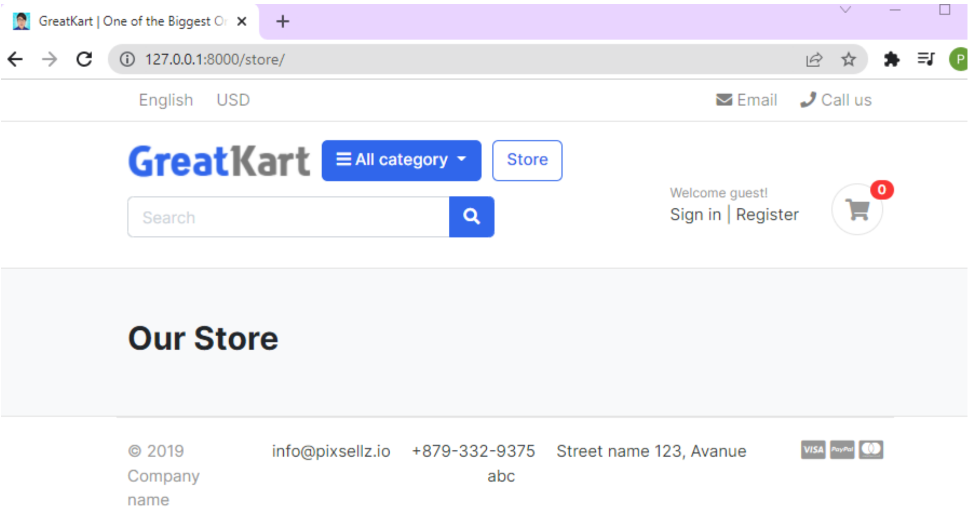
Street name 123, Avanue abc

76. Copy the codes store.html under sample project file and paste in store.html under store folder under templates folder

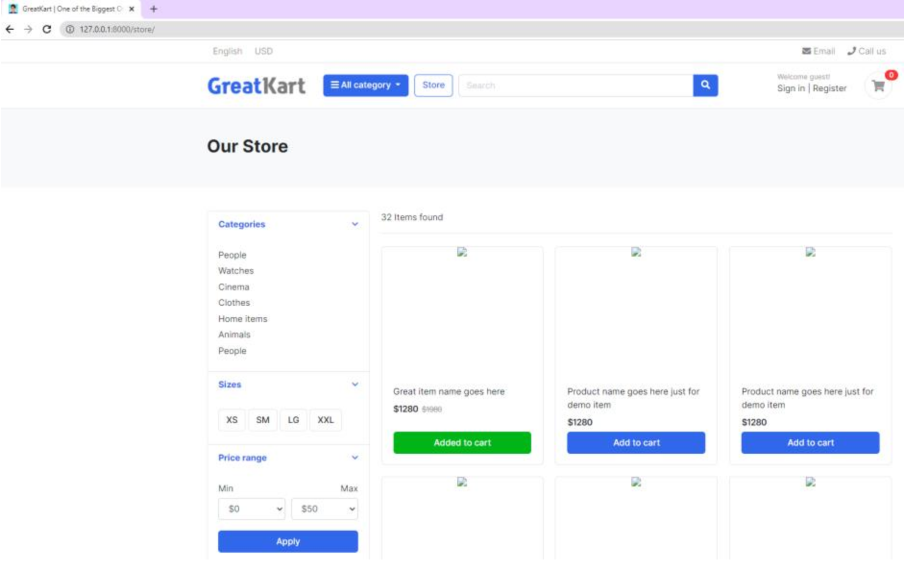
```
{% block content %}
<!-- ========================= SECTION PAGETOP ========================= -->
<section class="section-pagetop bg">
<div class="container">
    <h2 class="title-page">Our Store</h2>

</div> <!-- container //  -->
</section>
<!-- ========================= SECTION INTRO END// ========================= -->
```

77. Adjust line in the base.html under templates folder

```
<!-- navbar -->
{% include 'includes/navbar.html' %}
{% block content %}
<!-- content -->
{% endblock %}
<!-- footer -->
{% include 'includes/footer.html' %}
```



78. Copy SECTION CONTENT to SECTION CONTENT END from store.html under sample project folder



79. Load pictures into the store.html under store at 2nd line

```
{% extends 'base.html' %}
{% load static %}
{% block content %}{% load static %}
```

80. Replace image code

```
                <img src="images/items/1.jpg">
                <img src="{% static 'images/items/1.jpg' %}">
```

81. Add lines in the views.py in the store folder

```
from django.shortcuts import render
from .models import Product
# Create your views here.
def store(request):
    products = Product.objects.all().filter(is_available = True)
    context = {
        'products': products,
    }
    return render(request, 'store/store.html', context)
```

82. Delete the image 2 to last image in the store.html under store folder under templates folder

```
                <img src="{% static 'images/items/1.jpg' %}">

            </div> <!-- img-wrap.// -->
            <figcaption class="info-wrap">
                <div class="fix-height">
                    <a href="./product-detail.html" class="title">Great item name goes here</a>
                    <div class="price-wrap mt-2">
                        <span class="price">$1280</span>
                        <del class="price-old">$1980</del>
                    </div> <!-- price-wrap.// -->
                </div>
                <a href="#" class="btn btn-block btn-success">Added to cart </a>
            </figcaption>
        </figure>
    </div> <!-- col.// -->

</div> <!-- row end.// -->
```

83. Add loop to display the pictures

```
<div class="row">
  {% for product in products %}
    <div class="col-md-4">
        <figure class="card card-product-grid">
            <div class="img-wrap">
    …
            </figcaption>
        </figure>
    </div> <!-- col.// -->
  {% endfor %}
</div> <!-- row end.// -->
```

84. User able to see the same short appear on web screen with blue button of add to cart

```
<a href="#" class="btn btn-block btn-primary">Add to cart </a>
```

85. Add line to adjust the prices

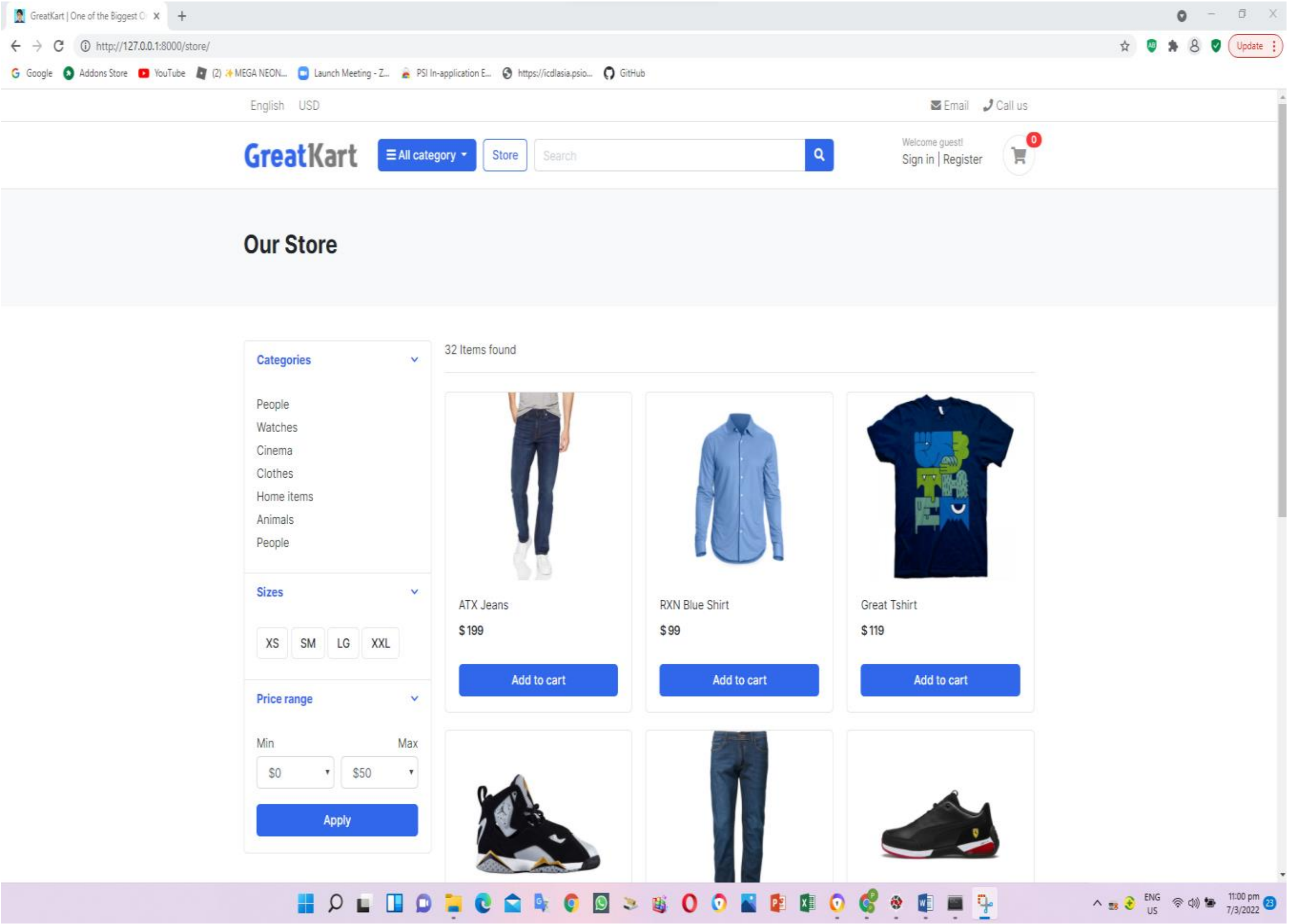
```
<span class="price">$ {{ product.price }}</span>
```

86. Delete 1 line

```
                    <del class="price-old">$1980</del>
```

87. Final adjustment to diplay the photo, product name and product price

```
<div class="row">
  {% for product in products %}
    <div class="col-md-4">
        <figure class="card card-product-grid">
            <div class="img-wrap">
                <img src="{{ product.images.url }}">
            </div> <!-- img-wrap.// -->
            <figcaption class="info-wrap">
                <div class="fix-height">
                    <a href="./product-detail.html" class="title">{{ product.product_name }}</a>
                    <div class="price-wrap mt-2">
                        <span class="price">$ {{ product.price }}</span>
                    </div> <!-- price-wrap.// -->
                </div>
                <a href="#" class="btn btn-block btn-primary">Add to cart </a>
            </figcaption>
        </figure>
    </div> <!-- col.// -->
  {% endfor %}
</div> <!-- row end.// -->
```

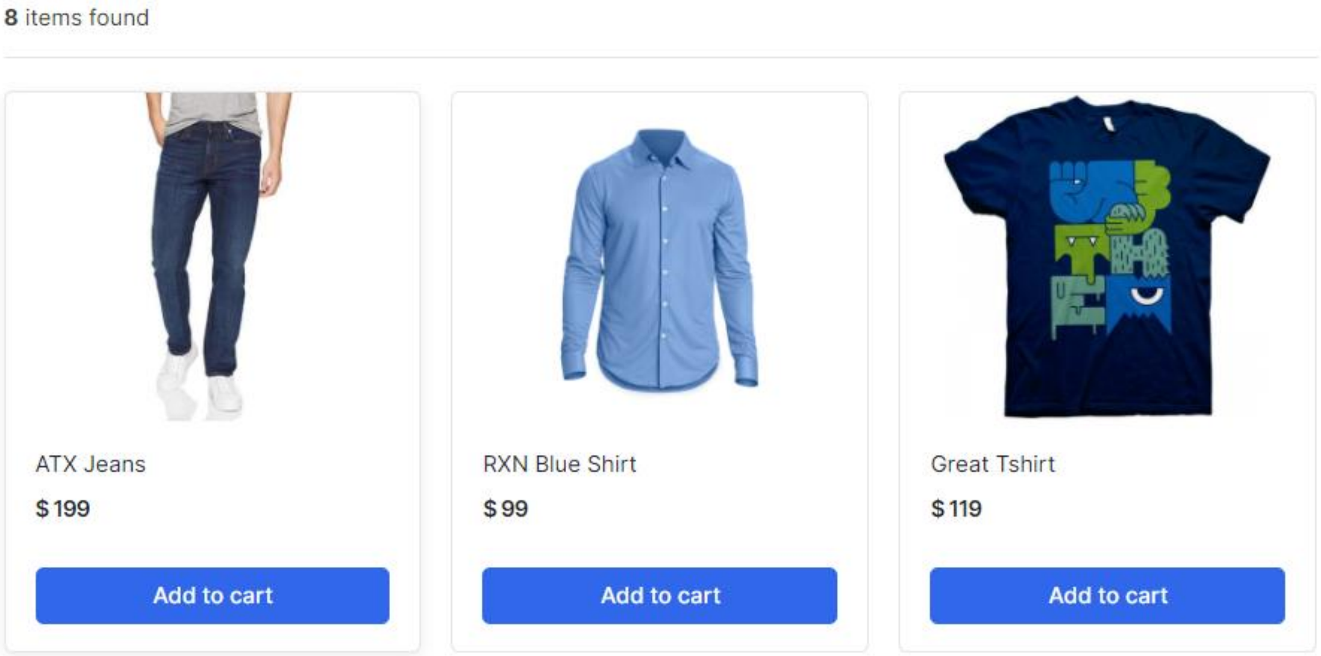


88. Add 2 lines in the view.py under store folder

```
from django.shortcuts import render
from .models import Product
# Create your views here.
def store(request):
    products = Product.objects.all().filter(is_available = True)
    product_count = products.count()
    context = {
        'products': products,
        'product_count': product_count,
    }
    return render(request, 'store/store.html', context)
```

89. Adjust the found item from static to link to database items under store.html under store folder under template folder

```
                  <span class="mr-md-auto">32 Items found </span>
                  <span class="mr-md-auto"><b>{{ product_count }}</b> items found </span>
```



8 items found

ATX Jeans

$ 199

RXN Blue Shirt

$ 99

Great Tshirt

$ 119

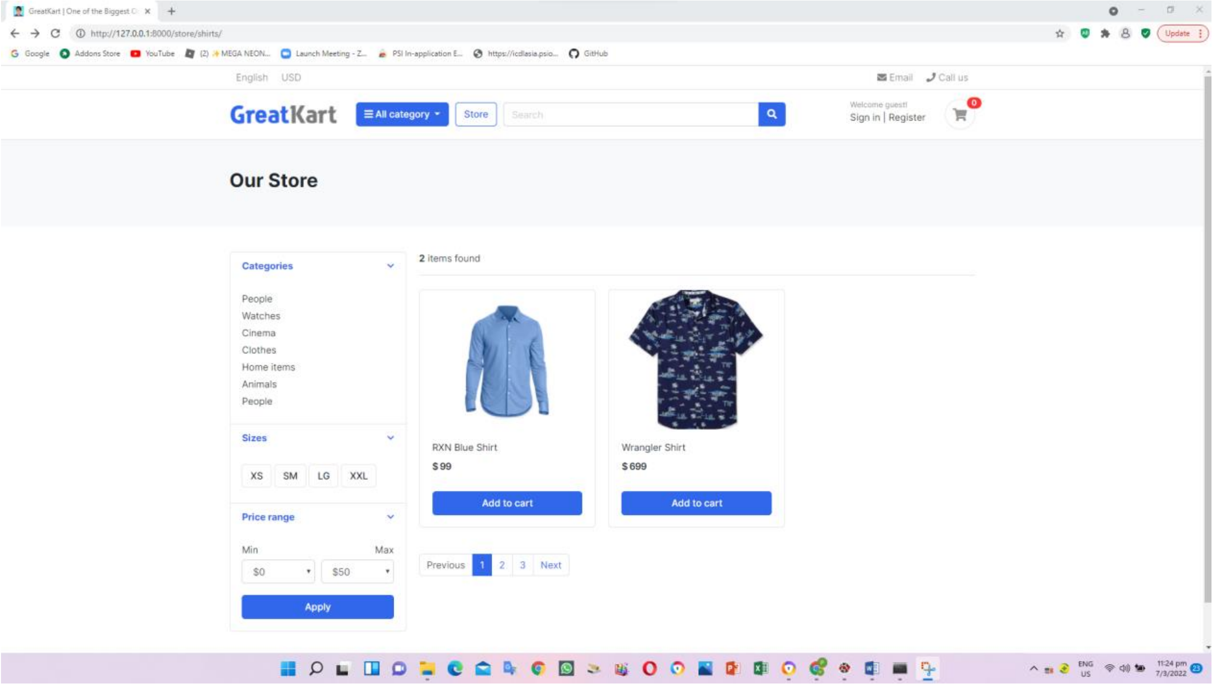
90. Adjust the lines in the urls.py under store folder (fetch the items by categories)

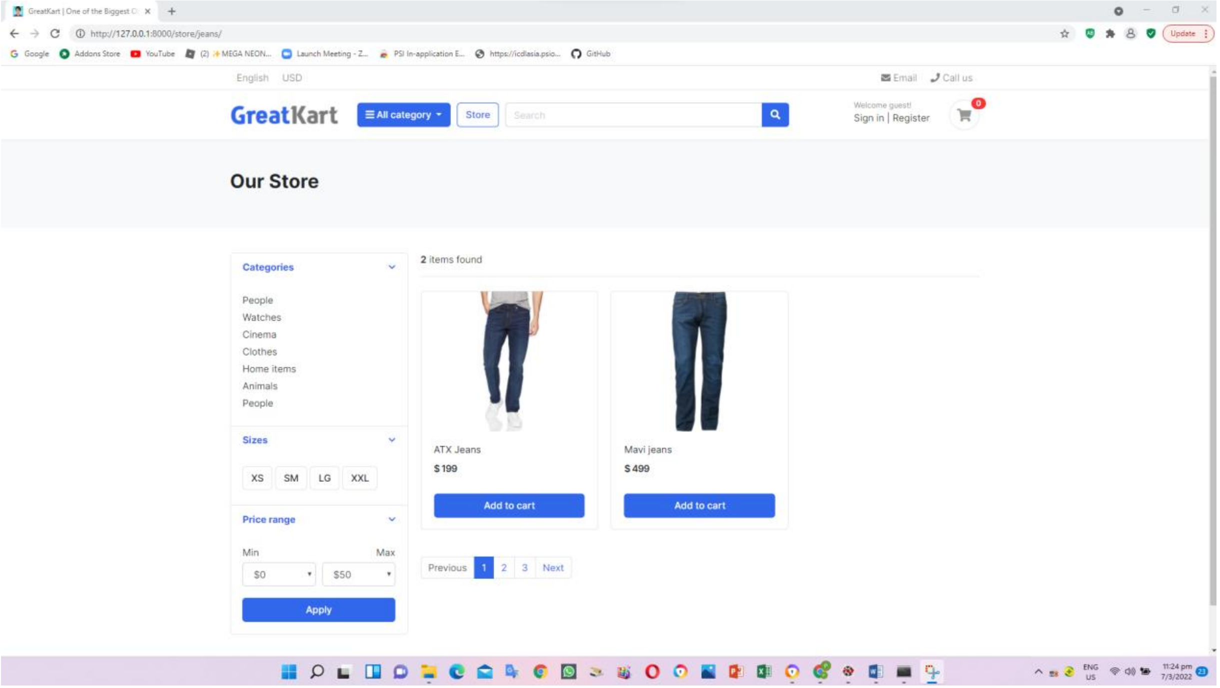
```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.store, name='store'),
    path('<slug:category_slug>/', views.store, name='products_by_category'),
]
```

91. Adjust the lines in the views.py under store folder

```
from django.shortcuts import render, get_object_or_404
from .models import Product
from category.models import Category
# Create your views here.
def store(request, category_slug=None):
    categories = None
```

```
products = None
if category_slug != None:
    categories = get_object_or_404(Category, slug=category_slug)
    products = Product.objects.filter(category=categories, is_available=True)
    product_count = products.count()
else:
    products = Product.objects.all().filter(is_available=True)
    product_count = products.count()
context = {
    'products': products,
    'product_count': product_count,
}
return render(request, 'store/store.html', context)
```





92. Create and add lines in the context_processors.py under category folder (Create linkage between category to web click)

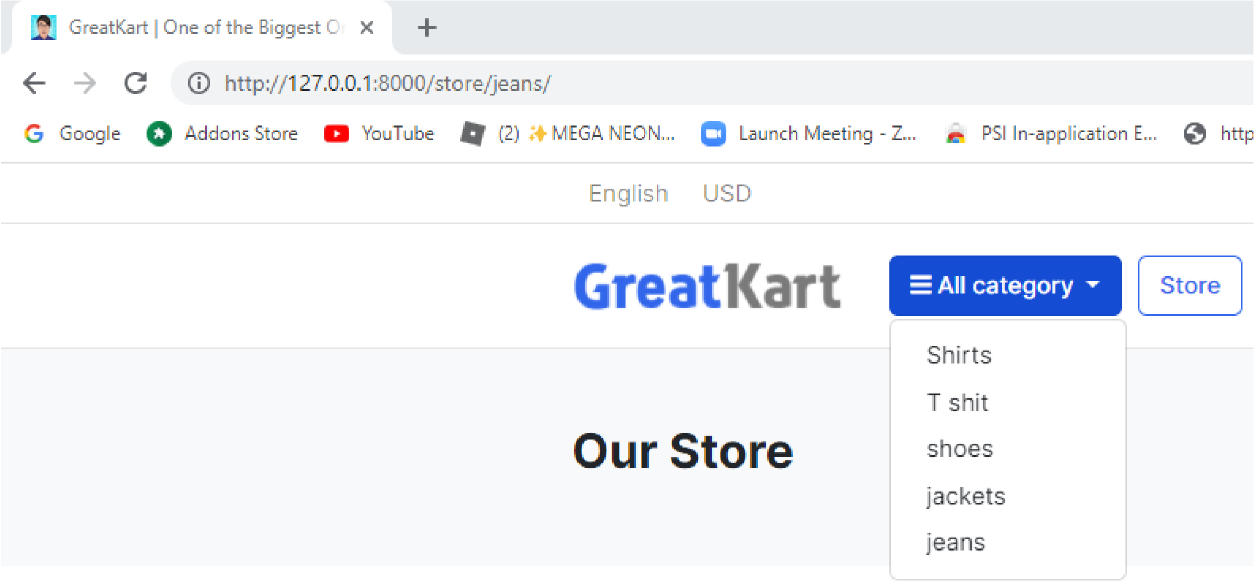
```
from .models import Category
def menu_links(request):
  links = Category.objects.all()
  return dict(links=links)
```

93. Add line in the setting.py in the online folder

```
TEMPLATES = [
  {
    'BACKEND': 'django.template.backends.django.DjangoTemplates',
    'DIRS': ['templates'],
    'APP_DIRS': True,
    'OPTIONS': {
      'context_processors': [
        'django.template.context_processors.debug',
        'django.template.context_processors.request',
        'django.contrib.auth.context_processors.auth',
        'django.contrib.messages.context_processors.messages',
        'category.context_processors.menu_links',
      ],
    },
  },
]
```

94. Edit the codes in the navbar.html under includes folder under templates folder

```
                    <i class="fa fa-bars"></i> All category
            </button>
            <div class="dropdown-menu">
                    {% for category in links %}
                    <a class="dropdown-item" href="{{ category.get_url }}">{{ category.category_name}}</a>
                    {% endfor%}
            </div>
    </div>  <!-- category-wrap.// -->
```

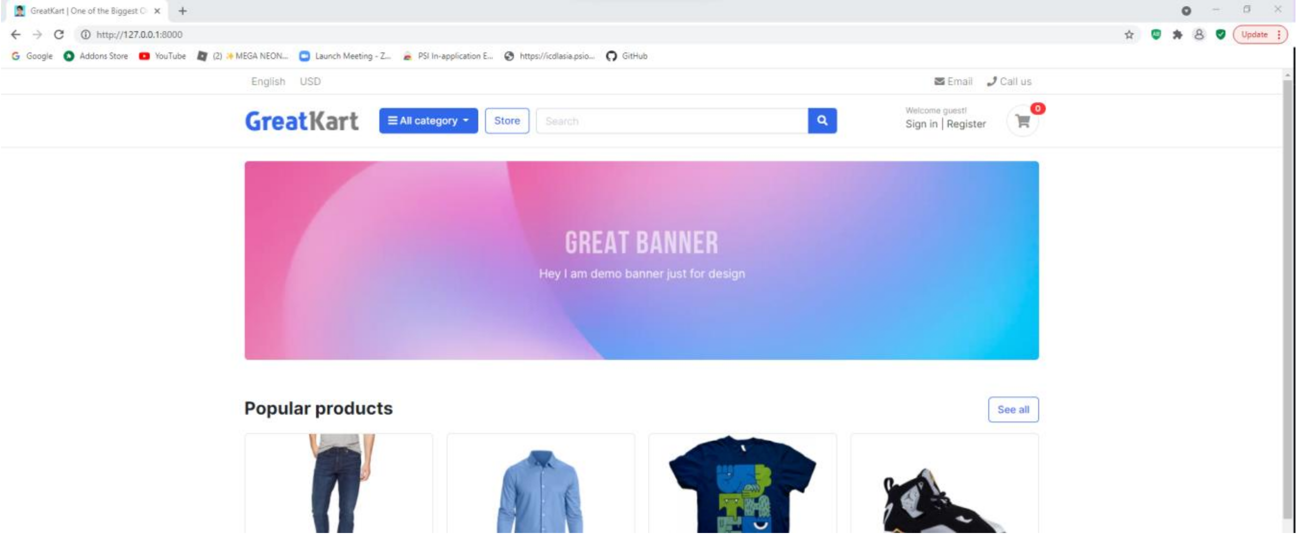


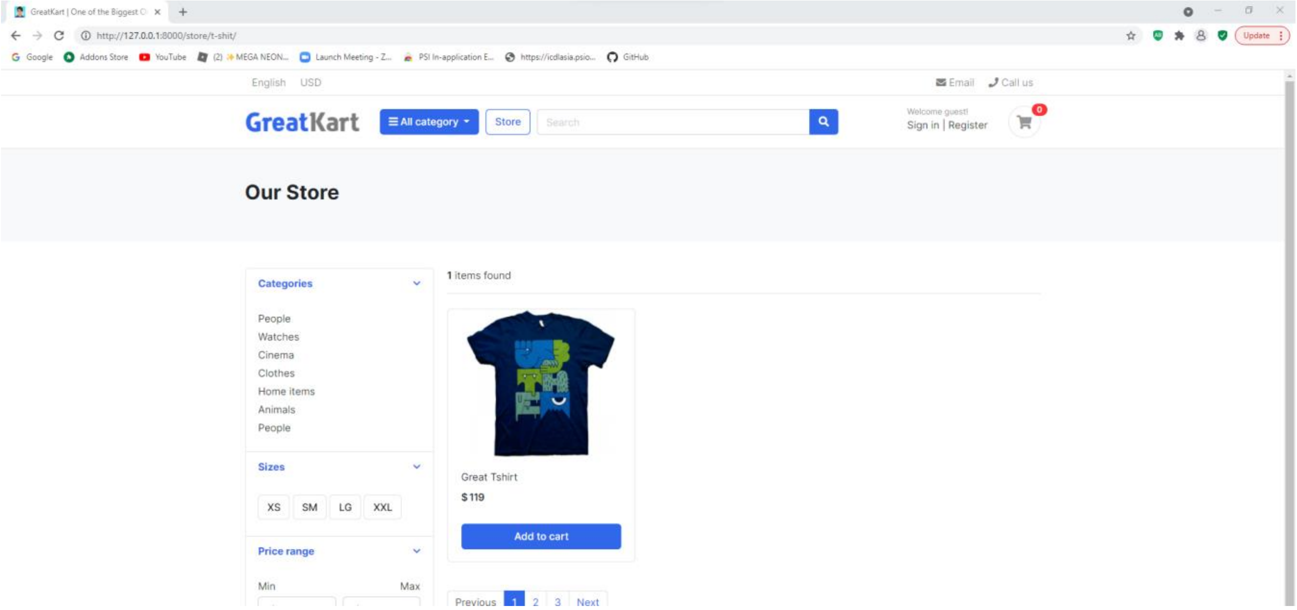
95. Adjust models.py under category folder

```
from django.db import models
from django.urls import reverse
# Create your models here.
class Category(models.Model):
    category_name = models.CharField(max_length=50, unique = True)
    slug = models.SlugField(max_length=100, unique=True)
    description = models.TextField(max_length=255, blank=True)
    cat_image = models.ImageField(upload_to='photos/categories', blank=True)
    class Meta:
        verbose_name = 'Category'
        verbose_name_plural = 'Categories'
    def get_url(self):
        return reverse('products_by_category', args=[self.slug])
    def __str__(self):
        return self.category_name
```

96. Adjust the home.html under templates folder (remove the s)

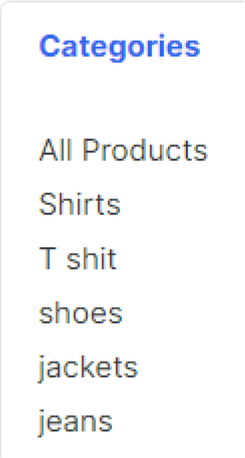
```
{% block content %}
```





97. Adjust codes in the store.html under store folder under templates folder (display category in the store page)

```
<div class="card-body">
    <ul class="list-menu">
        <li><a href="{% url 'store' %}">All Products </a></li>
        {% for category in links %}
    <li><a href="{{ category.get_url }}">{{ category.category_name }}  </a></li>
        {% endfor %}
    </ul>
</div> <!-- card-body.// -->
```



Categories

All Products

Shirts

T shit

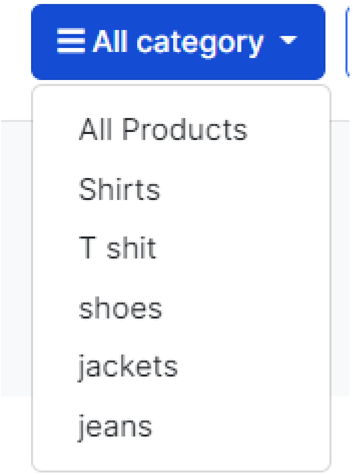
shoes

jackets

jeans

98. Adjust codes in the navbar.html under includes folder under templates folder

```
<div class="dropdown-menu">
    <a class="dropdown-item" href="{% url 'store' %}">All Products</a>
    {% for category in links %}
    <a class="dropdown-item" href="{{ category.get_url }}">{{ category.category_name}}</a>
    {% endfor %}
</div>
```



All Products

Shirts

T shit

shoes

jackets

jeans

99. Add line in the urls.py in the store folder (implement product detail URL and design)

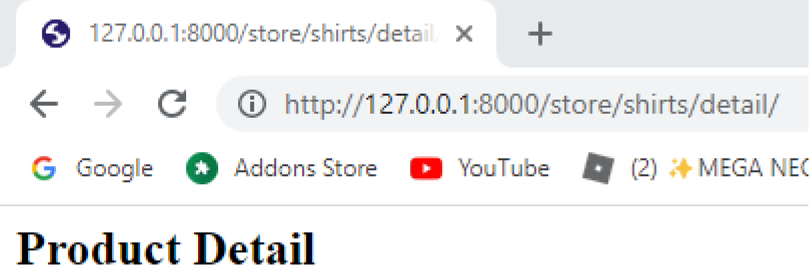
```
from django.urls import path
from . import views
urlpatterns = [
  path('', views.store, name='store'),
  path('<slug:category_slug>/', views.store, name='products_by_category'),
  path('<slug:category_slug>/<slug:product_slug>/', views.product_detail, name='product_detail'),
]
```

100. Add lines in the views.py in the store folder

```
    return render(request, 'store/store.html', context)
def product_detail(request,category_slug, product_slug):
    return render(request, 'store/product_detail.html')
```

101. Create product_details.html in the store folder in the templates folder

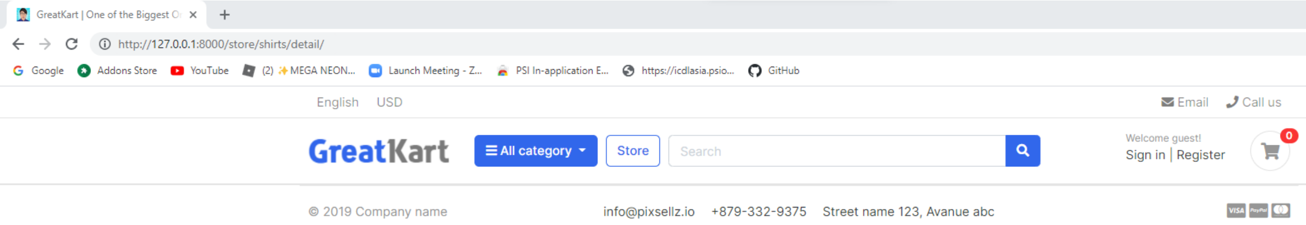
```
<h2>Product Detail</h2>
```


127.0.0.1:8000/store/shirts/detail
http://127.0.0.1:8000/store/shirts/detail/
Google
Addons Store
YouTube
(2) ✨MEGA NEC

Product Detail

102. Replace the codes in the product_details.html in the store folder in the templates folder

```
{% extends 'base.html' %}
{% block content %}
{% endblock %}
```



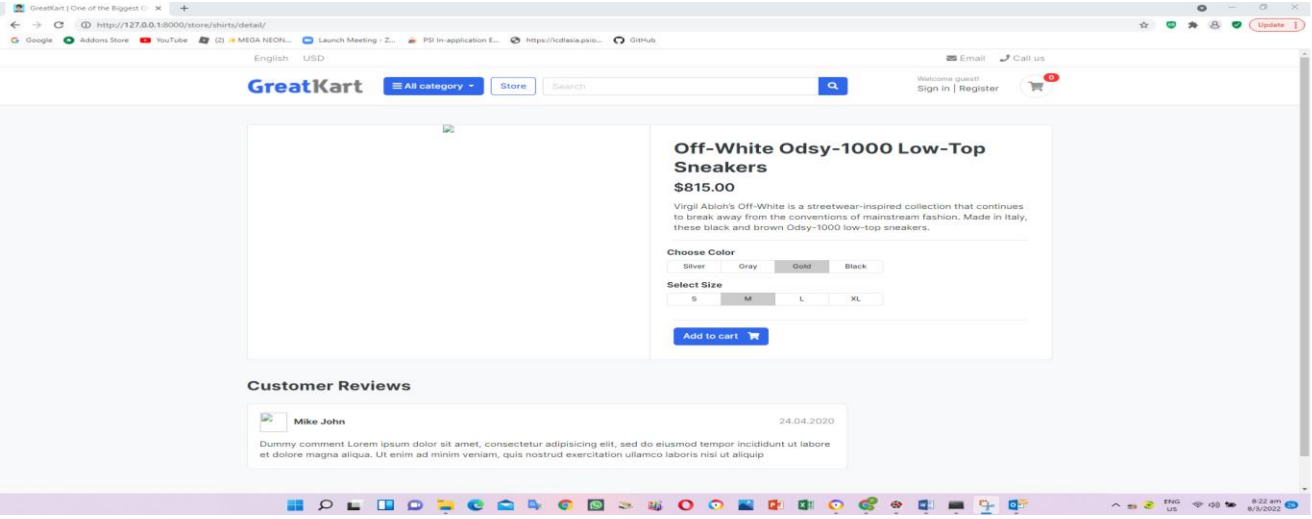
103. Copy the codes from product-detail.html from js folder from project sample folder to product_details.html in the store folder in the templates folder

Copy

```
<section class="section-content padding-y bg">
...
<!-- ========================= SECTION CONTENT END// ========================= -->
```

Paste

```
{% extends 'base.html' %}
{% block content %}
...
{% endblock %}
```



Off-White Odsy-1000 Low-Top Sneakers

$815.00

Virgil Abloh's Off-White is a streetwear-inspired collection that continues to break away from the conventions of mainstream fashion. Made in Italy, these black and brown Odsy-1000 low-top sneakers.

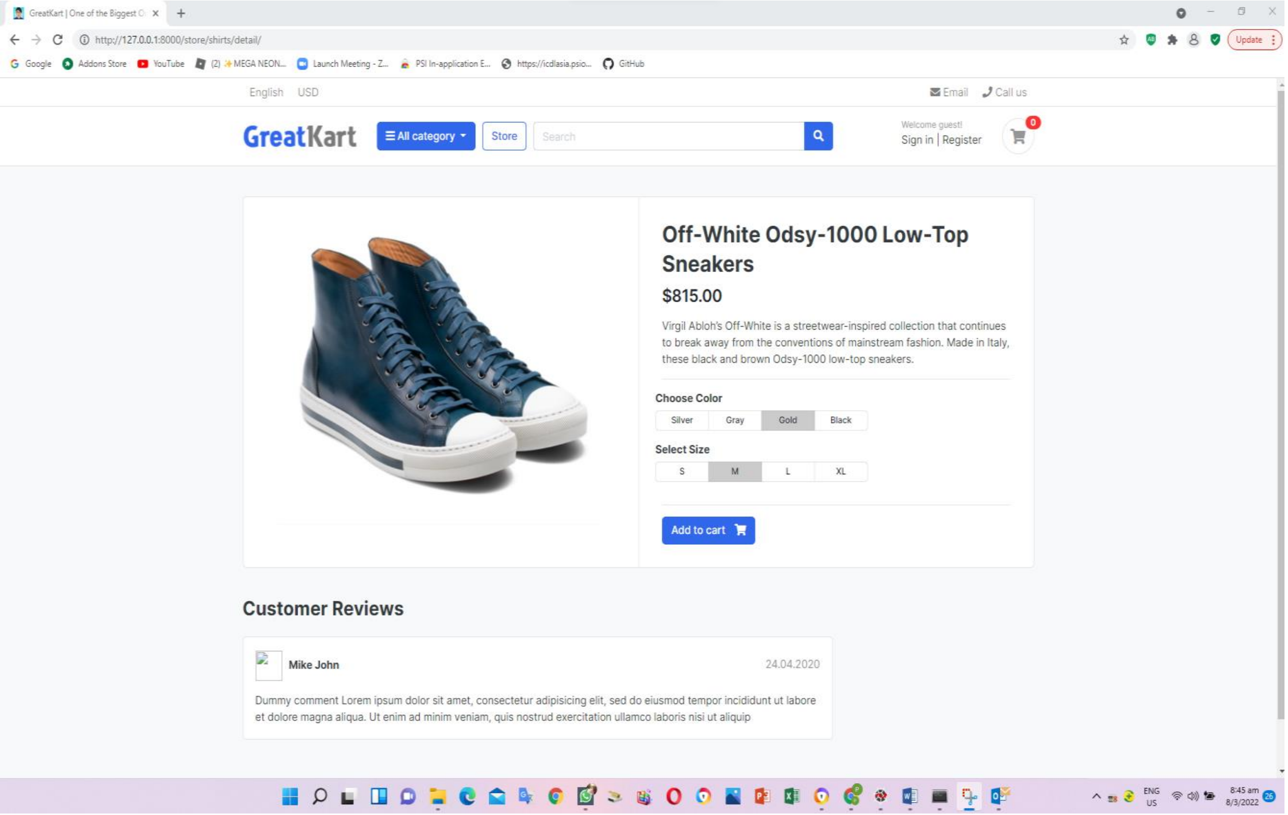
Customer Reviews

Mike John 24.04.2020

Dummy comment Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip

104. Adjust codes in the product_details.html in the store folder in the templates folder

```
{% extends 'base.html' %}
{% load static %}
{% block content %}
<section class="section-content padding-y bg">
<div class="container">
<!-- ============================ COMPONENT 1 ================================ -->
<div class="card">
    <div class="row no-gutters">
            <aside class="col-md-6">
<article class="gallery-wrap">
    <div class="img-big-wrap">
      <a href="#"><img src="{% static './images/items/12.jpg' %}"></a>
```



Off-White Odsy-1000 Low-Top Sneakers

$815.00

Virgil Abloh's Off-White is a streetwear-inspired collection that continues to break away from the conventions of mainstream fashion. Made in Italy, these black and brown Odsy-1000 low-top sneakers.

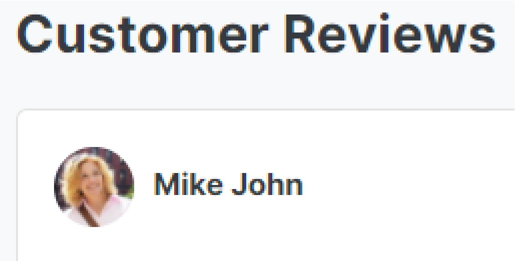
Customer Reviews

Mike John 24.04.2020

Dummy comment Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip

105. Adjust codes in the product_details.html in the store folder in the templates folder

```
    <article class="box mb-3">
            <div class="icontext w-100">
                    <img src="{% static './images/avatars/avatar1.jpg' %}" class="img-xs icon rounded-circle">
                    <div class="text">
                            <span class="date text-muted float-md-right">24.04.2020 </span>
                            <h6 class="mb-1">Mike John </h6>
```



Customer Reviews

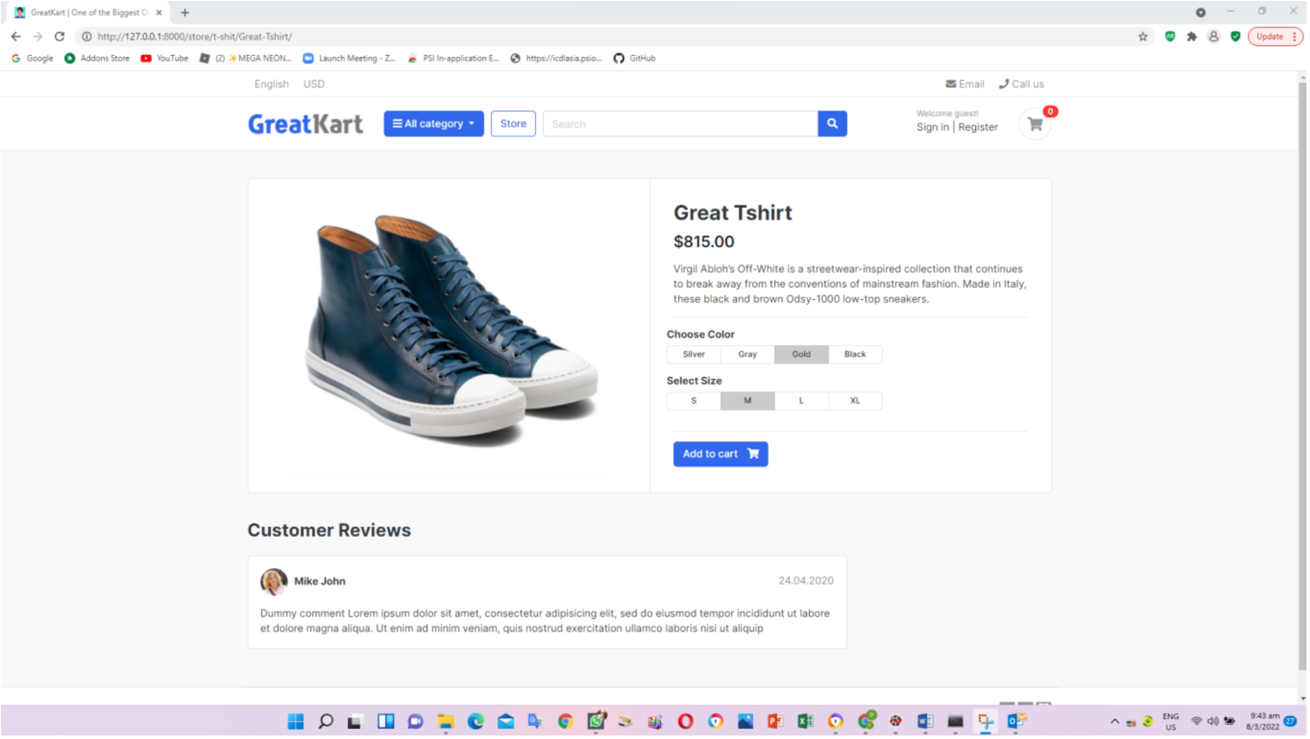
Mike John

106. Adjust codes in the views.py in the store folder (Single Product View)

```
def product_detail(request,category_slug, product_slug):
    try:
        single_product = Product.objects.get(category__slug = category_slug, slug=product_slug)
    except Exception as e:
        raise e
    context = {
        'single_product': single_product,
    }
    return render(request, 'store/product_detail.html', context)
```

107. Adjust codes in the product_details.html in the store folder in the templates folder

```
<article class="content-body">
<h2 class="title">{{ single_product.product_name }}</h2>
<div class="mb-3">
    <var class="price h4">$815.00</var>
```



Great Tshirt

$815.00

Virgil Abloh's Off-White is a streetwear-inspired collection that continues to break away from the conventions of mainstream fashion. Made in Italy, these black and brown Odsy-1000 low-top sneakers.

Choose Color

Silver | Gray | Gold | Black

Select Size

S | M | L | XL

Add to cart

Customer Reviews

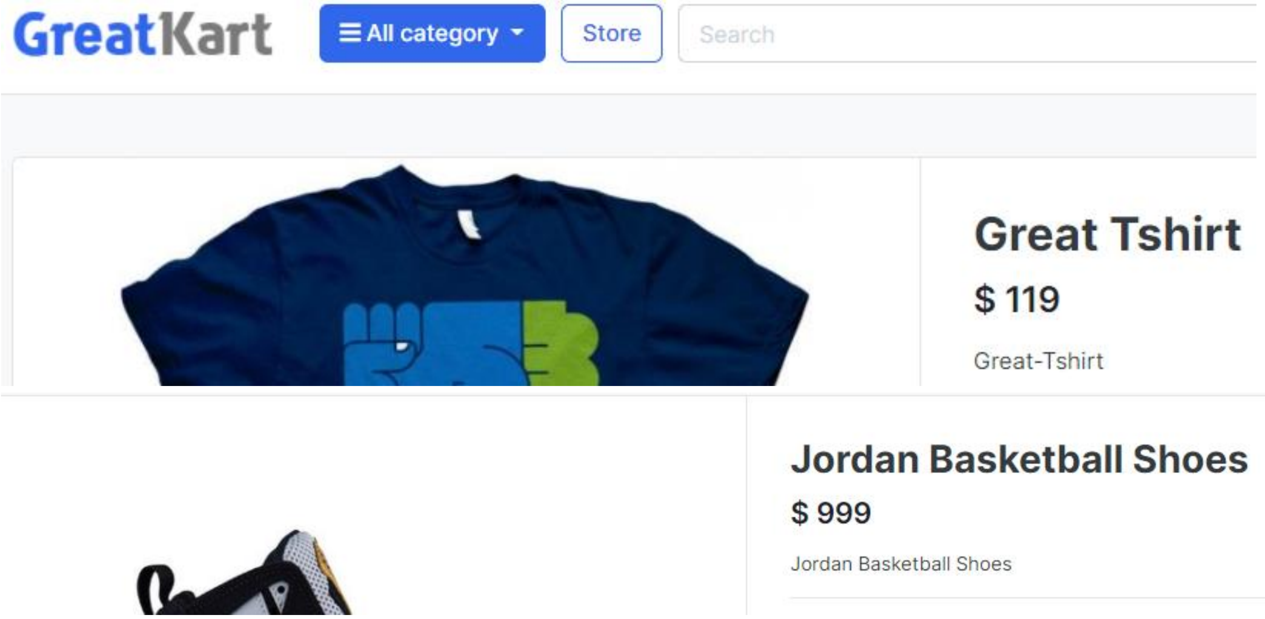
Mike John 24.04.2020

Dummy comment Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip

108. Adjust photo, price and description in the product_details.html in the store folder in the templates folder

```
<article class="gallery-wrap">
    <div class="img-big-wrap">
      <a href="#"><img src="{{ single_product.images.url }}"></a>
    </div> <!-- img-big-wrap.// -->
</article> <!-- gallery-wrap .end// -->
            </aside>
            <main class="col-md-6 border-left">
<article class="content-body">
<h2 class="title">{{ single_product.product_name }}</h2>
<div class="mb-3">
    <var class="price h4">$ {{ single_product.price }}</var>
</div>
```

```
<p>{{ single_product.description }}</p>
<hr>
```



109. Adjust the line in the navbar.html under includes folder under templates folder (Fix Logo and store button)

Old

```
<a href="./store.html" class="btn btn-outline-primary">Store</a>
```

New

```
<a href="{% url 'store' %}" class="btn btn-outline-primary">Store</a>
```

Old

```
<a href="./" class="brand-wrap">
```

New

```
<a href="{% url 'home' %}" class="brand-wrap">
```

110. Adjust the line in the home.html under templates folder (Fix see all button)

Old

```
<a href="./store.html" class="btn btn-outline-primary float-right">See all</a>
```

New

```
<a href="{% url 'store' %}" class="btn btn-outline-primary float-right">See all</a>
```

111. Adjust store.html under store folder under templates folder (click photo and item tittle to see each item)

Old

```
<a href="./product-detail.html" class="title">{{ product.product_name }}</a>
```

New

```
<a href="{{ product.get_url }}" class="title">{{ product.product_name }}</a>
```

Old

```
<img src="{{ product.images.url }}">
```

New

```
<a href="{{ product.get_url }}"><img src="{{ product.images.url }}"></a>
```

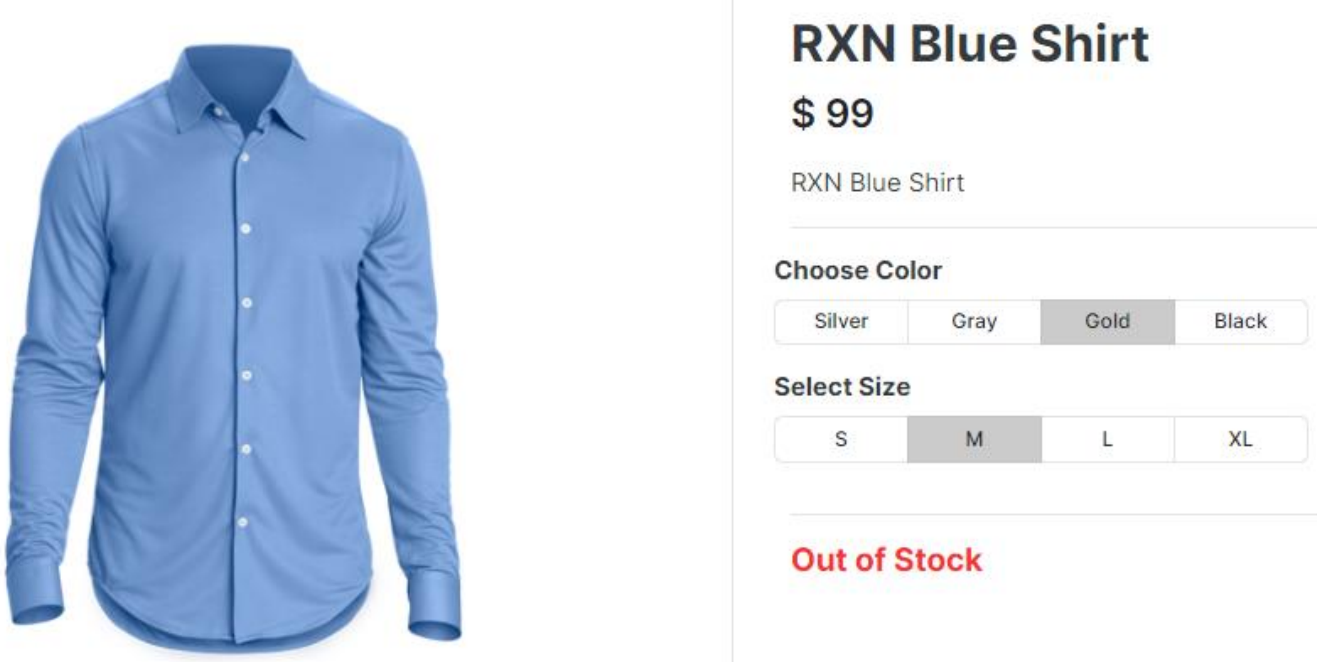
User able to click the photo from our store page to individual page

112. User change the banker codes from home.html under templates folder (Change banner)

```
<img src={% static 'images/banners/cover.jpg' %} class="img-fluid rounded">
```

113. Product available in the product_details.html in the store folder in the templates folder

```
{% if single_product.stock <= 0 %}
        <h5 class="text-danger">Out of Stock</h5>
{% else %}
        <a href="./product-detail.html" class="btn  btn-primary"> <span class="text">Add to cart</span> <i class="fas fa-shopping-cart"></i>  </a>
{% endif %}
```



RXN Blue Shirt

$ 99

RXN Blue Shirt

Choose Color

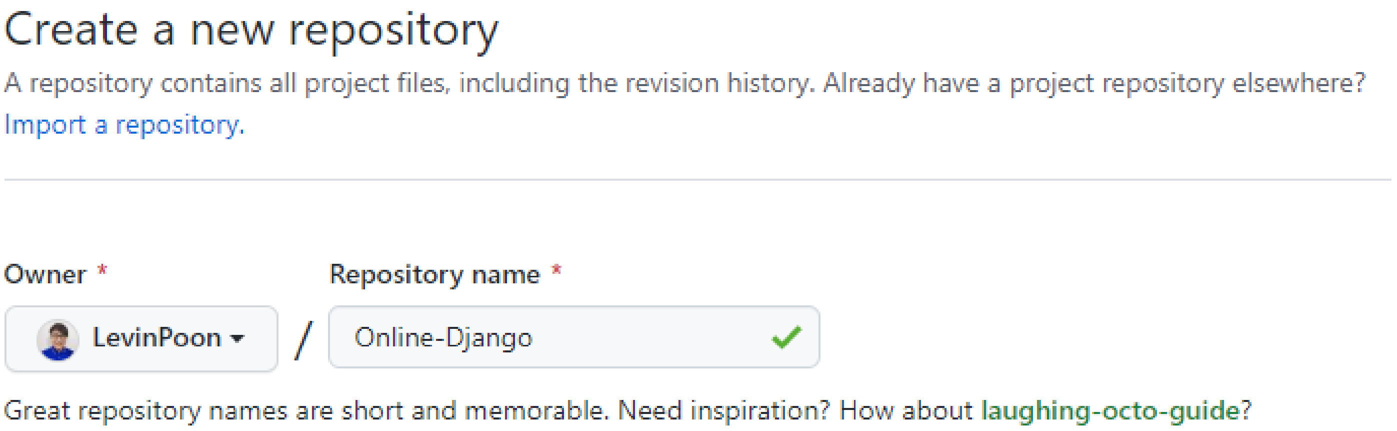
Silver | Gray | Gold | Black

Select Size

S | M | L | XL

Out of Stock

114. Log in GitHub and create a new repository and press Create repository



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * / Repository name *

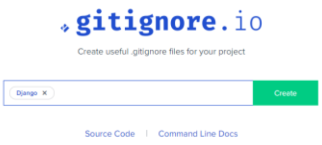
LevinPoon / Online-Django ✓

Great repository names are short and memorable. Need inspiration? How about laughing-octo-guide?

115. Enter command in the Anaconda Prompt (Anaconda3)

```
cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts
activate
cd C:\Users\angel\OneDrive\Desktop\Python Django
git init
```

116. Create an .gitignore file in online root folder (gitignore.io -> Django)



gitignore.io

Create useful .gitignore files for your project

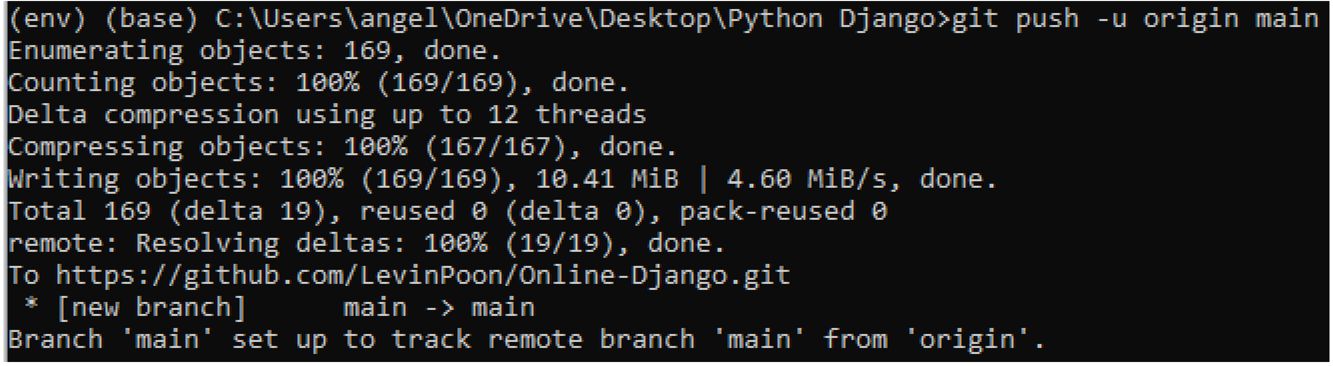
Django × | Create

Source Code | Command Line Docs

117. Copy and paste code from website to .gitignore file

118. Add codes to git repository

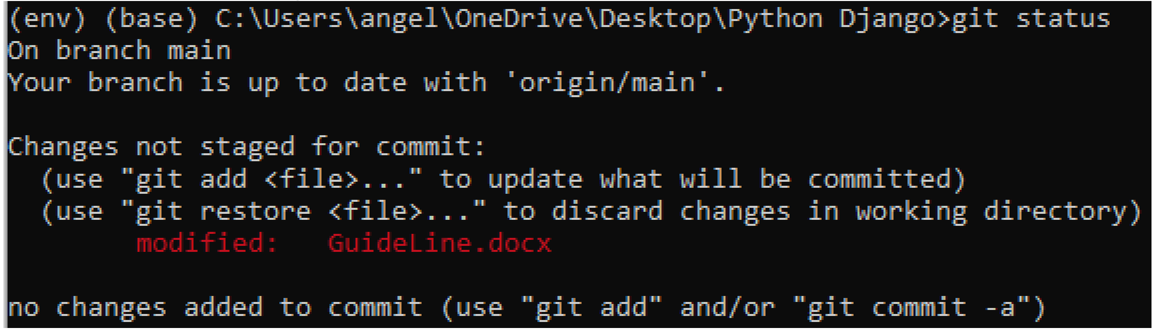
```
Git add –A
Git commit –m “this is my first commit”
git remote add origin https://github.com/LevinPoon/Online-Django.git
git branch -M main
git push -u origin main
```



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git push -u origin main
Enumerating objects: 169, done.
Counting objects: 100% (169/169), done.
Delta compression using up to 12 threads
Compressing objects: 100% (167/167), done.
Writing objects: 100% (169/169), 10.41 MiB | 4.60 MiB/s, done.
Total 169 (delta 19), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (19/19), done.
To https://github.com/LevinPoon/Online-Django.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```

User please reload the page

Git status check any updates



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   GuideLine.docx

no changes added to commit (use "git add" and/or "git commit -a")
```

119. Enter CMD codes to create carts application folder

```
python manage.py startapp carts
```

120. Insert line in the setting.py under online folder

```
    'store',
    'carts',
]
```

121. Create urls.py in the carts folder

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.cart, name='cart'),
]
```

122. Insert 1 line below store in the urls.py in the online folder

```
    path('store/', include('store.urls')),
    path('cart/', include('carts.urls')),
```

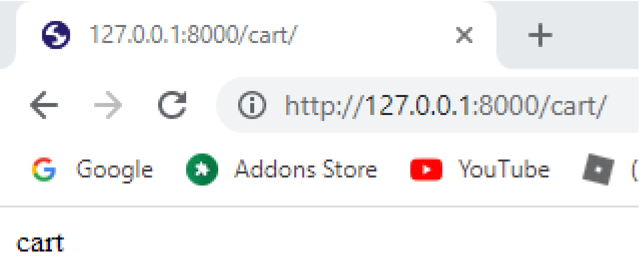
123. Edit views.py in the carts folder

```
from django.shortcuts import render
# Create your views here.
def cart(request):
    return render(request, 'store/cart.html')
```

124. Create cart.html under store folder under templates folder

cart.html

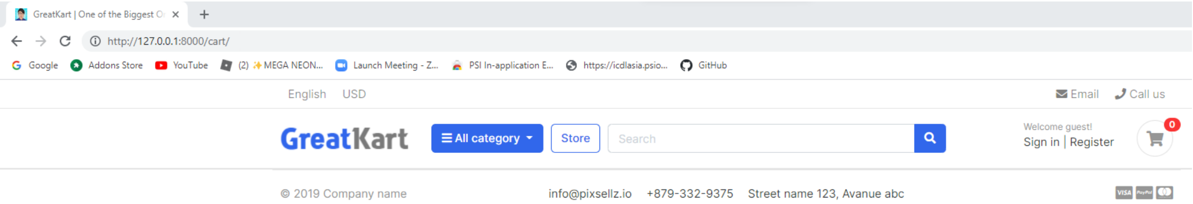
cart



cart

125. Adjust codes in cart.html under store folder under templates folder

```
{% extends 'base.html' %}
{% load static %}
{% block content %}
{% endblock %}
```



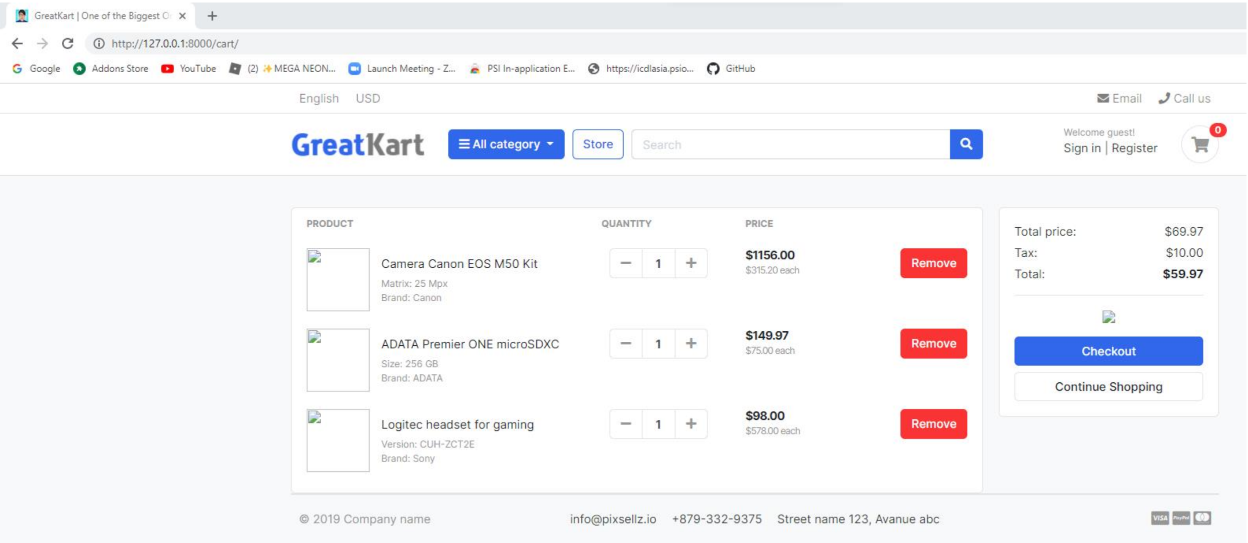
© 2019 Company name info@pixsellz.io +879-332-9375 Street name 123, Avanue abc

126. Copy session codes cart.html codes in project sample folder to cart.html under store folder under templates folder

```
<section class="section-content padding-y bg">
…
<!-- ========================= SECTION CONTENT END// ========================= -->
```

Paste in between

```
{% block content %}
…
{% endblock %}
```



PRODUCT	QUANTITY	PRICE
Camera Canon EOS M50 Kit Matrix: 25 Mpx Brand: Canon	1	$1156.00 $315.20 each
ADATA Premier ONE microSDXC Size: 256 GB Brand: ADATA	1	$149.97 $75.00 each
Logitec headset for gaming Version: CUH-ZCT2E Brand: Sony	1	$98.00 $578.00 each

Total price:	$69.97
Tax:	$10.00
Total:	**$59.97**

© 2019 Company name info@pixsellz.io +879-332-9375 Street name 123, Avanue abc

127. Edit the cart.html to display photo in the store folder in the templates folder (display item and visa logo)

Old

```
<div class="aside"><img src="./images/items/11.jpg" class="img-sm"></div>
```

New

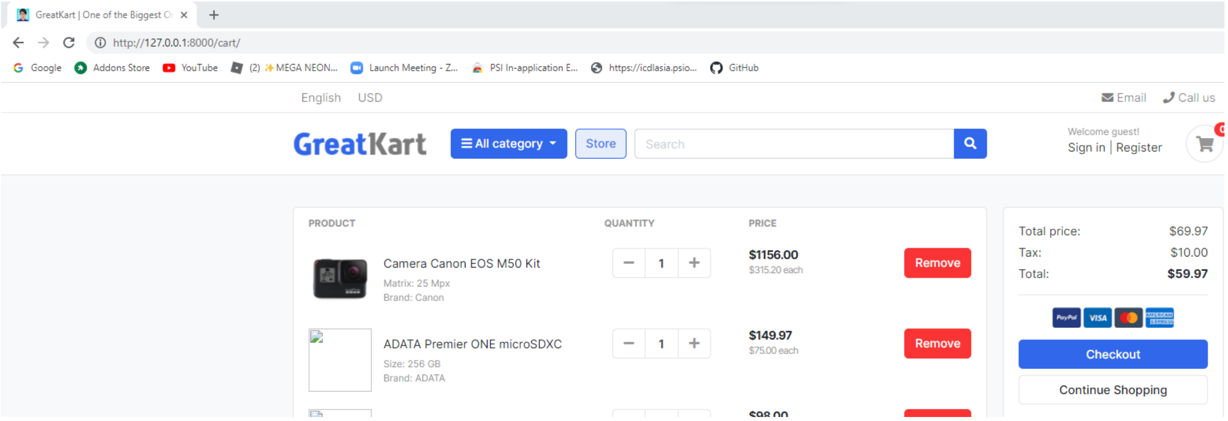
```
<div class="aside"><img src="{% static './images/items/11.jpg' %}" class="img-sm"></div>
```

Old

```
<img src="./images/misc/payments.png" height="26">
```

New

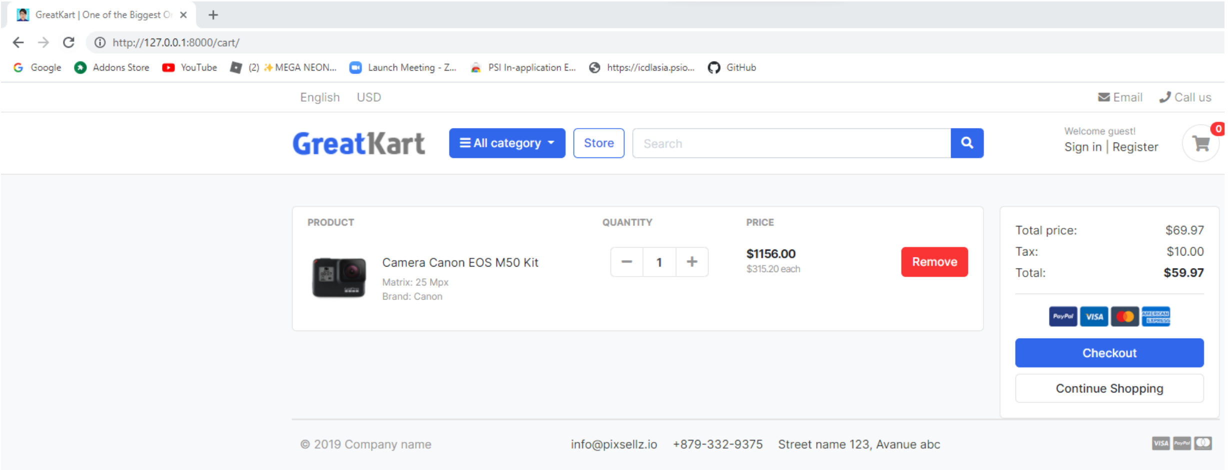
```
<img src="{% static './images/misc/payments.png' %}" height="26">
```



PRODUCT	QUANTITY	PRICE
Camera Canon EOS M50 Kit Matrix: 25 Mpx Brand: Canon	1	$1156.00 $315.20 each
ADATA Premier ONE microSDXC Size: 256 GB Brand: ADATA	1	$149.97 $75.00 each
		$98.00

Total price:	$69.97
Tax:	$10.00
Total:	$59.97

128. Delete 2nd tr to 3rd tr codes in the cart.html in the store folder in the templates folder



PRODUCT	QUANTITY	PRICE
Camera Canon EOS M50 Kit Matrix: 25 Mpx Brand: Canon	1	$1156.00 $315.20 each

Total price:	$69.97
Tax:	$10.00
Total:	$59.97

© 2019 Company name info@pixsellz.io +879-332-9375 Street name 123, Avanue abc

129. Add lines in models.py in the carts folder

```
from django.db import models
from store.models import Product
# Create your models here.
class Cart(models.Model):
  cart_id = models.CharField(max_length=250, blank=True)
  date_added = models.DateField(auto_now_add=True)
  def __str__(self):
    return self.cart_id
class CartItem(models.Model):
  product = models.ForeignKey(Product, on_delete=models.CASCADE)
  cart    = models.ForeignKey(Cart, on_delete=models.CASCADE)
  quantity = models.IntegerField()
  is_active = models.BooleanField(default=True)
  def __str__(self):
    return self.product
```

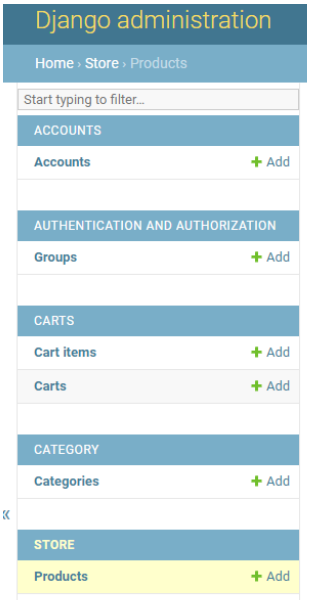
130. Add lines in admin.py in the carts folder

```
from django.contrib import admin
from .models import Cart, CartItem
# Register your models here.
```

```
admin.site.register(Cart)
admin.site.register(CartItem)
```

131. Enter codes CMD to migrate in admin page

```
python manage.py makemigrations
python manage.py migrate
```



132. Add line to views.py under the carts folder (add item to cart from individual tab)

```
from django.shortcuts import render, redirect
from store.models import Product
from .models import Cart, CartItem
# Create your views here.
def _cart_id(request):
   cart = request.session.session_key
   if not cart:
      cart = request.session.create()
   return cart
def add_cart(request, product_id):
   product = Product.objects.get(id=product_id) #get the product
   try:
      cart = Cart.objects.get(cart_id=_cart_id(request)) #get the cart using the cart_id present in the session
   except Cart.DoesNotExist:
      cart = Cart.objects.create(
         cart_id = _cart_id(request)
      )
   cart.save()
   try:
      cart_item = CartItem.objects.get(product=product, cart=cart)
      cart_item.quantity += 1 # cart_item.quantity = cart_item.quantity + 1
      cart_item.save()
   except CartItem.DoesNotExist:
      cart_item = CartItem.objects.create(
         product = product,
         quantity = 1,
            cart = cart,
        )
        cart_item.save()
    return redirect('cart')
def cart(request):
    return render(request, 'store/cart.html')
```

133. Add line to urls.py under the carts file

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.cart, name='cart'),
    path('add_cart/<int:product_id>/', views.add_cart, name='add_cart'),
]
```

134. Add lines in the product_detail.html in the store folder in the templates folder

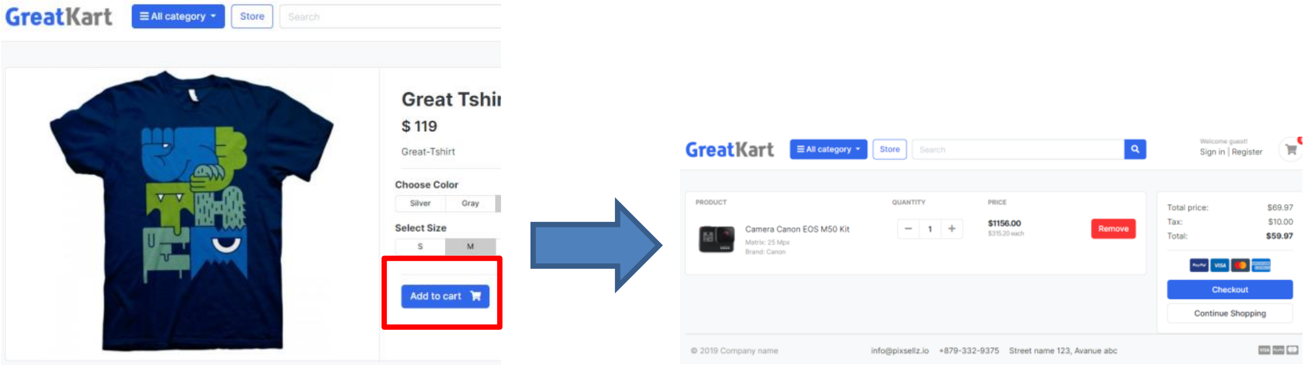
Old

```
            <a href="./product-detail.html" class="btn  btn-primary"> <span class="text">Add to cart</span> <i
class="fas fa-shopping-cart"></i>  </a>
```

New

```
            <a href="{% url 'add_cart' single_product.id %}" class="btn  btn-primary"> <span class="text">Add to
cart</span> <i class="fas fa-shopping-cart"></i>  </a>
```

User able to add to cart by press “Add to cart” button



135. Adjust line in the view.py under carts folder (Insert new item to cart)

```
From Django.core.exceptions import ObjectDoesNotExist
&
def cart(request, total=0, quantity=0, cart_items=None):
    try:
        tax = 0
        grand_total = 0
        cart = Cart.objects.get(cart_id=_cart_id(request))
        cart_items = CartItem.objects.filter(cart=cart, is_active=True)
        for cart_item in cart_items:
            total += (cart_item.product.price * cart_item.quantity)
            quantity += cart_item.quantity
        tax = (7 * total)/100
        grand_total = total + tax
    except ObjectDoesNotExist:
        pass #just ignore
    context = {
        'total': total,
        'quantity': quantity,
        'cart_items': cart_items,
        'tax': tax,
        'grand_total': grand_total,
    }
    return render(request, 'store/cart.html', context)
```

136. Add for loop in front and end <tr> in the cart.html under store folder under templates folder

```
{% for cart_item in cart_items %}
<tr>
</tr>
{% endfor %}
```

137. Display cart item image in the cart.html under store folder under templates folder

Old

```
                <div class="aside"><img src="{% static './images/items/11.jpg' %}" class="img-sm"></div>
```

New

```
                <div class="aside"><img src="{{ cart_item.product.images.url }}" class="img-sm"></div>
```

138. Display cart item name in the cart.html under store folder under templates folder

Old

```
                    <a href="#" class="title text-dark">Camera Canon EOS M50 Kit</a>
```

New

```
                    <a href="#" class="title text-dark">{{ cart_item.product.product_name }}</a>1
```

139. Display no of cart item in the cart.html under store folder under templates folder

Old

```
                <input type="text" class="form-control"  value="1">
```

New

```
                <input type="text" class="form-control"  value="{{ cart_item.quantity }}">
```

140. Display total price of cart item in the cart.html under store folder under templates folder

Old

```
                <var class="price">$1156.00</var>
                <small class="text-muted"> $315.20 each </small>
```

New

```
                <var class="price">$ {{ cart_item.sub_total }}</var>
                <small class="text-muted"> $ {{ cart_item.product.price }} each </small>
```

141. Add function codes in the models.py under carts folder

```
class CartItem(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    cart    = models.ForeignKey(Cart, on_delete=models.CASCADE)
    quantity = models.IntegerField()
    is_active = models.BooleanField(default=True)
    def sub_total(self):
        return self.product.price * self.quantity
    def __str__(self):
        return self.product
```

142. Adjust codes to increase item in cart.html under store folder under templates folder

Old

```
<button class="btn btn-light" type="button" id="button-minus"> <i class="fa fa-plus"></i> </button>
```

New

```
<a href="{% url 'add_cart' cart_item.product.id %}" class="btn btn-light" type="button" id="button-minus"> <i class="fa fa-plus"></i> </a>
```

143. Adjust codes to display total price in cart.html under store folder under templates folder

Old

```
                    <dd class="text-right">$69.97</dd>
                    <dd class="text-right"> $10.00</dd>
                    <dd class="text-right text-dark b"><strong>$59.97</strong></dd>
```

New

```
                    <dd class="text-right">$ {{ total }}</dd>
                    <dd class="text-right"> $ {{ tax }}</dd>
                    <dd class="text-right text-dark b"><strong>${{ grand_total }}</strong></dd>
```

144. Add lines in the view.py under the carts folder

```
from django.shortcuts import render, redirect, get_object_or_404
```

&

```
def remove_cart(request, product_id):
  cart = Cart.objects.get(cart_id=_cart_id(request))
  product = get_object_or_404(Product, id=product_id)
  cart_item = CartItem.objects.get(product=product, cart=cart)
  if cart_item.quantity > 1:
    cart_item.quantity -= 1
    cart_item.save()
  else:
    cart_item.delete()
  return redirect('cart')
```

145. Add lines in the view.py under the carts folder

```
  path('add_cart/<int:product_id>/', views.add_cart, name='add_cart'),
  path('remove_cart/<int:product_id>/', views.remove_cart, name='remove_cart'),
```

146. Adjust codes to display total price in cart.html under store folder under templates folder (decrement & remove)

Old

```
<button class="btn btn-light" type="button" id="button-plus"> <i class="fa fa-minus"></i> </button>
```

New

```
<a href="{% url 'remove_cart' cart_item.product.id %}" class="btn btn-light" type="button" id="button-plus"> <i class="fa fa-minus"></i> </a>
```

147. Add lines in the view.py under the carts folder

```
def remove_cart_item(request, product_id):
  cart = Cart.objects.get(cart_id=_cart_id(request))
  product = get_object_or_404(Product, id=product_id)
  cart_item = CartItem.objects.get(product=product, cart=cart)
  cart_item.delete()
  return redirect('cart')
```

148. Add lines in the view.py under the carts folder

```
  path('remove_cart/<int:product_id>/', views.remove_cart, name='remove_cart'),
    path('remove_cart_item/<int:product_id>/', views.remove_cart_item, name='remove_cart_item'),
```

149. Adjust codes to remove item in cart.html under store folder under templates folder

Old

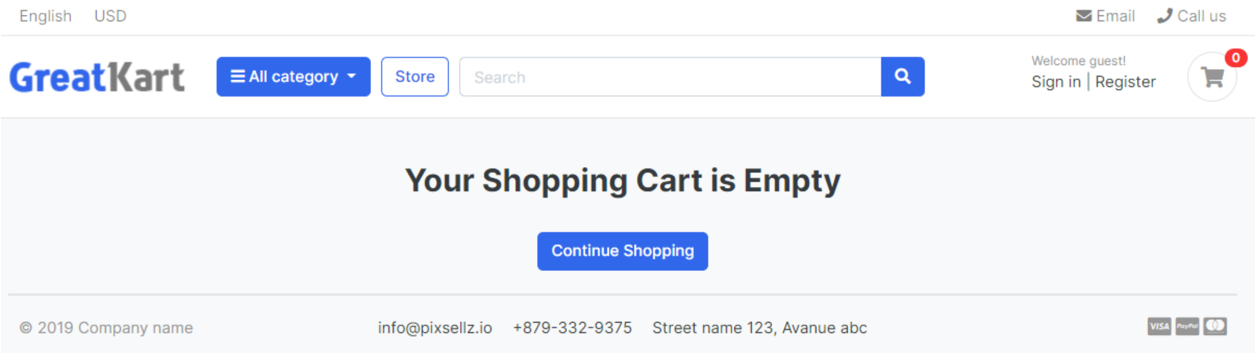
```
    <a href="" class="btn btn-danger"> Remove</a>
```

New

```
    <a href="{% url 'remove_cart_item' cart_item.product.id %}" class="btn btn-danger"> Remove</a>
```

150. Add if else code in the cart.html under store folder under templates folder

```
<!-- ============================= COMPONENT 1 ================================== -->
{% if not cart_items %}
  <h2 class="text-center" > Your Shopping Cart is Empty</h2>
  <br>
  <div class="text-center">
    <a href="{% url 'store' %}" class="btn btn-primary">Continue Shopping</a>
  </div>
  <br>
{% else %}
<div class="row">
...
</div> <!-- row.// -->
{% endif %}
<!-- ============================= COMPONENT 1 END .// ================================== -->
```



English USD

Email Call us

GreatKart

All category

Store

Search

Welcome guest!

Sign in | Register

0

Your Shopping Cart is Empty

Continue Shopping

© 2019 Company name info@pixsellz.io +879-332-9375 Street name 123, Avanue abc

151. Adjust line to add to cart in the store page under store.html under store under templates

Old

```
<a href="#" class="btn btn-block btn-primary">Add to cart </a>
```

New

```
<a href="{% url 'add_cart' product.id %}" class="btn btn-block btn-primary">Add to cart </a>
```

152. Zoom into each item page from cart page under cart.html under store folder under templates folder

Old

```
<a href="#" class="title text-dark">{{ cart_item.product.product_name }}</a>
```

New

```
<a href="{{ cart_item.product.get_url }}" class="title text-dark">{{ cart_item.product.product_name }}</a>
```

153. Continue shopping under cart.html under store folder under templates folder

Old

```
<a href="./store.html" class="btn btn-light btn-block">Continue Shopping</a>
```

New

```
<a href="{% url 'store' %}" class="btn btn-light btn-block">Continue Shopping</a>
```

154. Display added to cart logo under cart.html under store folder under templates folder

```
from django.shortcuts import render, get_object_or_404
from .models import Product
from category.models import Category
from carts.models import CartItem
from carts.views import _cart_id
from django.http import HttpResponse
# Create your views here.
def store(request, category_slug=None):
    categories = None
    products = None
    if category_slug != None:
        categories = get_object_or_404(Category, slug=category_slug)
        products = Product.objects.filter(category=categories, is_available=True)
        product_count = products.count()
    else:
        products = Product.objects.all().filter(is_available=True)
        product_count = products.count()
    context = {
        'products': products,
        'product_count': product_count,
    }
    return render(request, 'store/store.html', context)
def product_detail(request, category_slug, product_slug):
    try:
        single_product = Product.objects.get(category__slug = category_slug, slug=product_slug)
        in_cart = CartItem.objects.filter(cart__cart_id=_cart_id(request), product=single_product).exists()
        return HttpResponse(in_cart)
        exit()
    except Exception as e:
        raise e
    context = {
        'single_product': single_product,
        'in_cart': in_cart,
    }
    return render(request, 'store/product_detail.html', context)
```

User able to see true or false after add to cart to match with existing cart status.

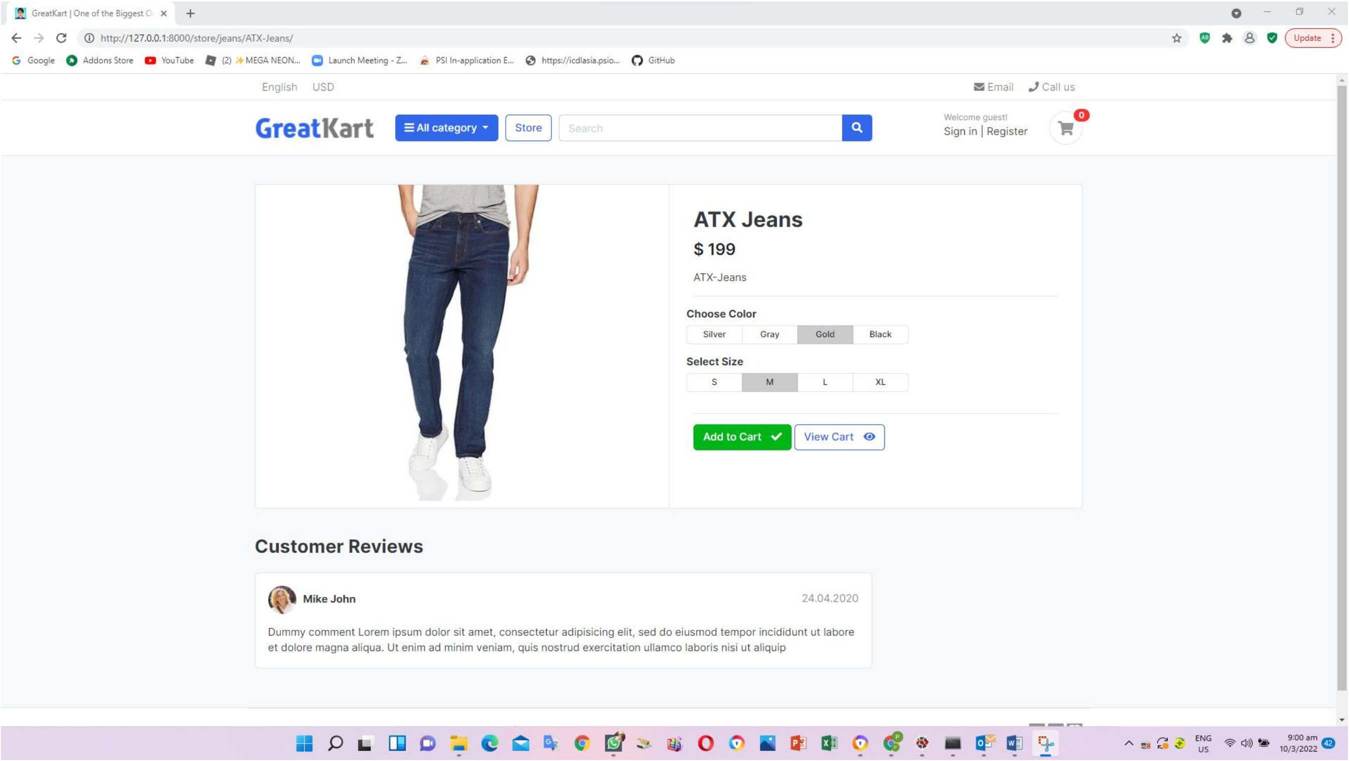
155. Add lines in the product_detail.html under store folder under templates folder

```
    {% if single_product.stock <= 0 %}
            <h5 class="text-danger">Outof Stock</h5>
    {% else %}
            {% if in_cart %}
            <a href="#" class="btn  btn-success"> <span class="text">Add to Cart</span> <i class="fas fa-check"></i>
</a>
            <a href="{% url 'cart' %}" class="btn  btn-outline-primary"> <span class="text">View Cart</span> <i class="fas
fa-eye"></i>  </a>
      {% else %}
```

```
                <a href="{% url 'add_cart' single_product.id %}" class="btn  btn-primary"> <span class="text">Add to
Cart</span> <i class="fas fa-shopping-cart"></i>  </a>
          {% endif %}
       {% endif %}
```

156. Remove code cart.html under store folder under templates folder

```
        return HttpResponse(in_cart)
        exit()
```



Dummy comment Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip

157. Create context_processors.py under carts folder (Linkage Cart Icon)

```
from .models import Cart, CartItem
from .views import _cart_id
def counter(request):
    cart_count = 0
    if 'admin' in request.path:
        return {}
    else:
        try:
            cart = Cart.objects.filter(cart_id=_cart_id(request))
            cart_items = CartItem.objects.all().filter(cart=cart[:1])
            for cart_item in cart_items:
                cart_count += cart_item.quantity
        except Cart.DoesNotExist:
            cart_count = 0
    return dict(cart_count=cart_count)
```

158. Add lines in settings.py under online folder

```
                'category.context_processors.menu_links',
                'carts.context_processors.counter',
```

159. Add line in the navbar.html under includes under templates folder

Old

```
a href="./cart.html" class="widget-header pl-3 ml-3">
<span class="badge badge-pill badge-danger notify">0</span>
```

New

```
<a href="{% url 'cart' %}" class="widget-header pl-3 ml-3">
<span class="badge badge-pill badge-danger notify">{{ cart_count }}</span>
```

160. Replace add to cart with view each item in the store.html under store folder under templates folder

Old

```
<a href="{% url 'add_cart' product.id %}" class="btn btn-block btn-primary">Add to cart </a>
```

New

```
<a href="{{ product.get_url }}" class="btn btn-block btn-primary">View Details </a>
```

161. Add lines in the view.py under store folder (view 6 items in the website)

```
from django.shortcuts import render, get_object_or_404
from .models import Product
from category.models import Category
from carts.models import CartItem
from carts.views import _cart_id
from django.http import HttpResponse
from django.core.paginator import EmptyPage, PageNotAnInteger, Paginator
# Create your views here.
def store(request, category_slug=None):
    categories = None
    products = None
    if category_slug != None:
        categories = get_object_or_404(Category, slug=category_slug)
        products = Product.objects.filter(category=categories, is_available=True)
        paginator = Paginator(products, 6)
        page = request.GET.get('page')
        paged_products = paginator.get_page(page)
        product_count = products.count()
    else:
        products = Product.objects.all().filter(is_available=True).order_by('id')
        paginator = Paginator(products, 6)
        page = request.GET.get('page')
        paged_products = paginator.get_page(page)
        product_count = products.count()
    context = {
        'products': paged_products,
        'product_count': product_count,
    }
    return render(request, 'store/store.html', context)
def product_detail(request, category_slug, product_slug):
    try:
        single_product = Product.objects.get(category__slug = category_slug, slug=product_slug)
        in_cart = CartItem.objects.filter(cart__cart_id=_cart_id(request), product=single_product).exists()
    except Exception as e:
        raise e
```

```
context = {
  'single_product': single_product,
  'in_cart': in_cart,
}
return render(request, 'store/product_detail.html', context)
```

162. Edit codes in the store.html under store folder under templates folder display paginator correctly

Old

```
<nav class="mt-4" aria-label="Page navigation sample">
 <ul class="pagination">
  <li class="page-item disabled"><a class="page-link" href="#">Previous</a></li>
  <li class="page-item active"><a class="page-link" href="#">1</a></li>
  <li class="page-item"><a class="page-link" href="#">2</a></li>
  <li class="page-item"><a class="page-link" href="#">3</a></li>
  <li class="page-item"><a class="page-link" href="#">Next</a></li>
 </ul>
</nav>
```

New

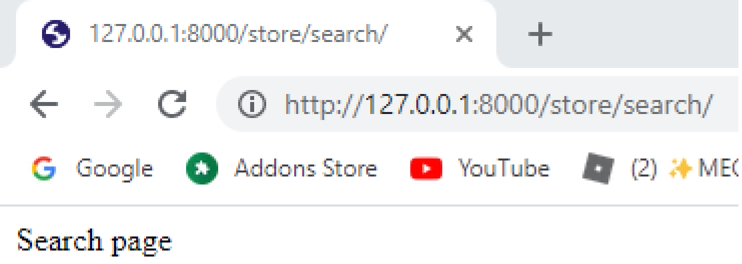
```
<nav class="mt-4" aria-label="Page navigation sample">
  {% if products.has_other_pages %}
    <ul class="pagination">
   {% if products.has_previous %}
     <li class="page-item"><a class="page-link"
href="?page={{products.previous_page_number}}">Previous</a></li>
          {% else %}
          <li class="page-item disabled"><a class="page-link" href="#">Previous</a></li>
          {% endif %}

          {% for i in products.paginator.page_range %}
          {% if products.number == i %}
                  <li class="page-item active"><a class="page-link" href="#">{{i}}</a></li>
                  {% else %}
                          <li class="page-item"><a class="page-link" href="?page={{i}}">{{i}}</a></li>
                  {% endif %}
     {% endfor %}

          {% if products.has_next %}
          <li class="page-item"><a class="page-link" href="?page={{products.next_page_number}}">Next</a></li>
          {% else %}
                  <li class="page-item disabled"><a class="page-link" href="#">Next</a></li>
          {% endif %}
    </ul>
   {% endif %}
</nav>
```

163. Adjust lines in the urls.py under store folder (search function)

```
  path('category/<slug:category_slug>/', views.store, name='products_by_category'),
  path('category/<slug:category_slug>/<slug:product_slug>/', views.product_detail, name='product_detail'),
```

164. Add lines in the view.py under store folder (search function)

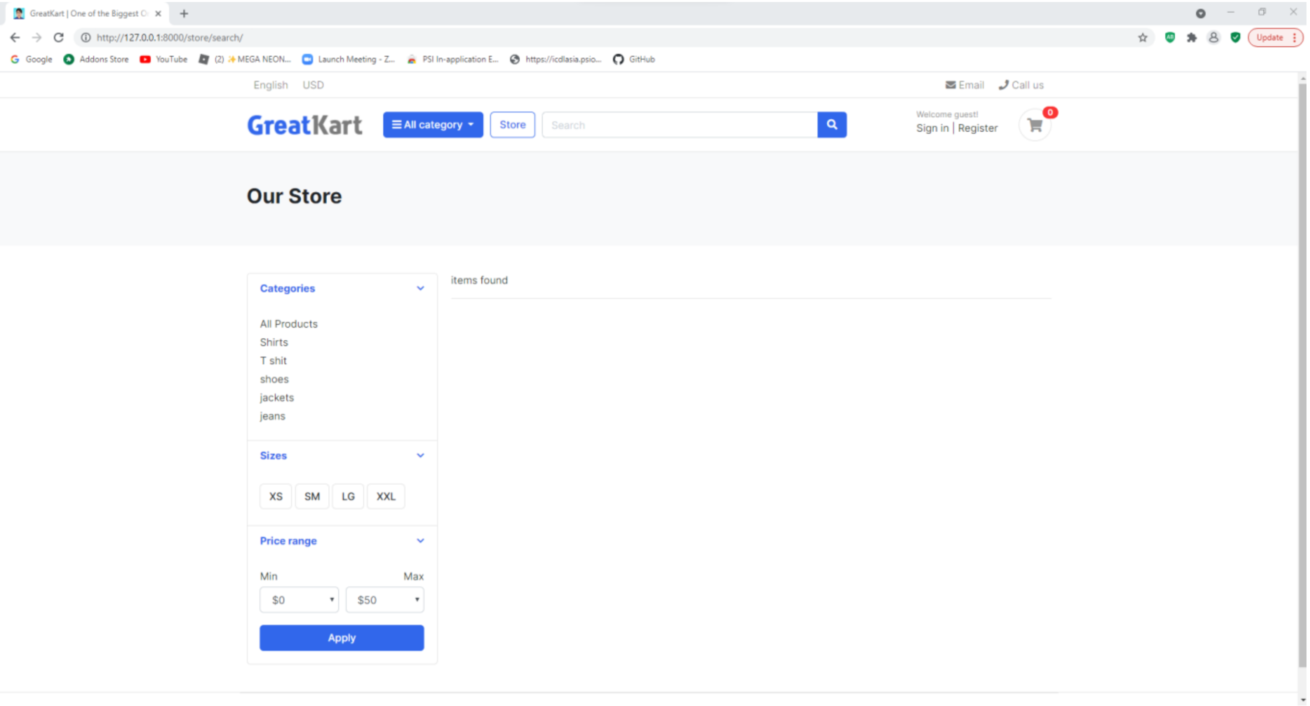
```
from django.http import HttpResponse
&
def search(request):
  return HttpResponse('Search page')
```


127.0.0.1:8000/store/search/
http://127.0.0.1:8000/store/search/
Google
Addons Store
YouTube
(2) ✨MEC

Search page

165. Replace the codes in the view.py under store folder (search function)

```
def search(request):
  return render(request, 'store/store.html')
```



166. Edit navbar.html under includes folder under templates

Old

```
          <form action="#" class="search">
                 <div class="input-group w-100">
                   <input type="text" class="form-control" style="width:60%;" placeholder="Search">
```

New

```
          <form action="{% url 'search' %}" class="search" method='GET'>
                 <div class="input-group w-100">
                   <input type="text" class="form-control" style="width:60%;" placeholder="Search"
name="keyword">
```

167. Edit codes in views.py under store folder

Old

```
def search(request):
  return render(request, 'store/store.html')
```

New

```
from django.db.models import Q
```

&

```
def search(request):
    if 'keyword' in request.GET:
        keyword = request.GET['keyword']
        if keyword:
            products = Product.objects.order_by('-created_date').filter(Q(description__icontains=keyword) |
Q(product_name__icontains=keyword))
            product_count = products.count()
    context = {
        'products': products,
        'product_count': product_count,
    }
    return render(request, 'store/store.html', context)
```

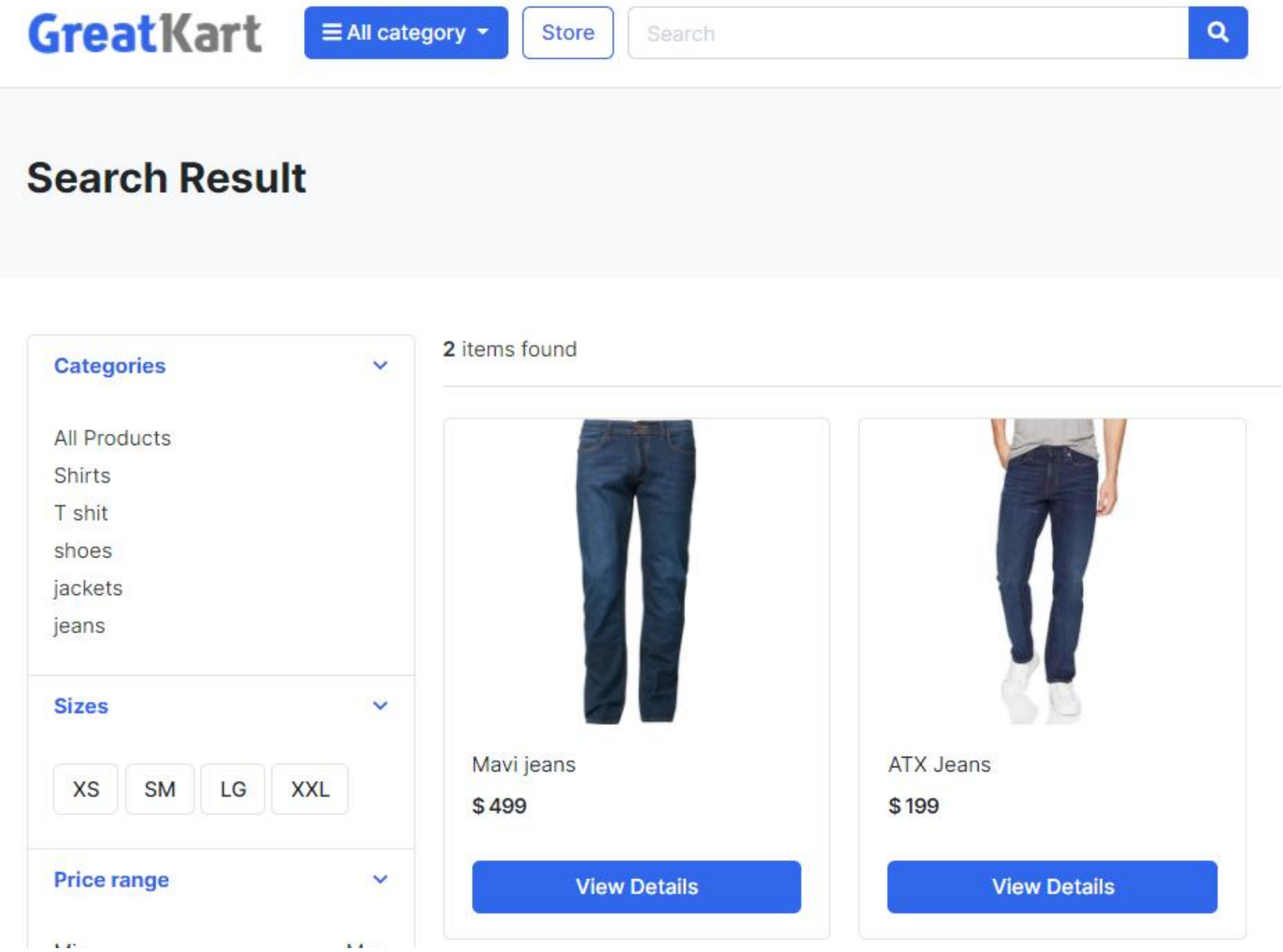
168. Edits store.html under store folder under templates folder

```
<!-- ========================= SECTION PAGETOP ========================= -->
<section class="section-pagetop bg">
<div class="container">
  {% if 'search' in request.path %}
    <h2 class="title-page">Search Result</h2>
    {% else %}
    <h2 class="title-page">Our Store</h2>
    {% endif %}
</div> <!-- container //  -->
</section>
<!-- ========================= SECTION INTRO END// ========================= -->
```

&

```
<div class="row">
  {% if products %}
  {% for product in products %}
    <div class="col-md-4">
            <figure class="card card-product-grid">
                    <div class="img-wrap">
                            <a href="{{ product.get_url }}"><img src="{{ product.images.url }}"></a>
                    </div> <!-- img-wrap.// -->
                    <figcaption class="info-wrap">
                            <div class="fix-height">
                                    <a href="{{ product.get_url }}" class="title">{{ product.product_name }}</a>
                                    <div class="price-wrap mt-2">
                                            <span class="price">$ {{ product.price }}</span>
                                    </div> <!-- price-wrap.// -->
                            </div>
                            <a href="{{ product.get_url }}" class="btn btn-block btn-primary">View Details </a>
                    </figcaption>
            </figure>
```

```
        </div> <!-- col.// -->
      {% endfor %}
      {% else %}
      <div>
         <h2> No result found. Please try again.</h2>
      </div>
      {% endif %}
  </div> <!-- row end.// -->
```



169. Edit the product_detail.html in store folder in templates folder (product variation color and size selection)

```
<form action="">
<article class="content-body">
…
</article> <!-- product-info-aside .// -->
</form>
```

170. Edit the product_detail.html under store folder under templates folder (product variation)

Old

```
<a href="{% url 'add_cart' single_product.id %}" class="btn  btn-primary"> <span class="text">Add to Cart</span> <i class="fas fa-shopping-cart"></i>  </a>
```

New

```
<button type="submit” class="btn  btn-primary"> <span class="text">Add to Cart</span> <i class="fas fa-shopping-cart"></i>  </button>
```

&

Old

```
<form action="">
```

New

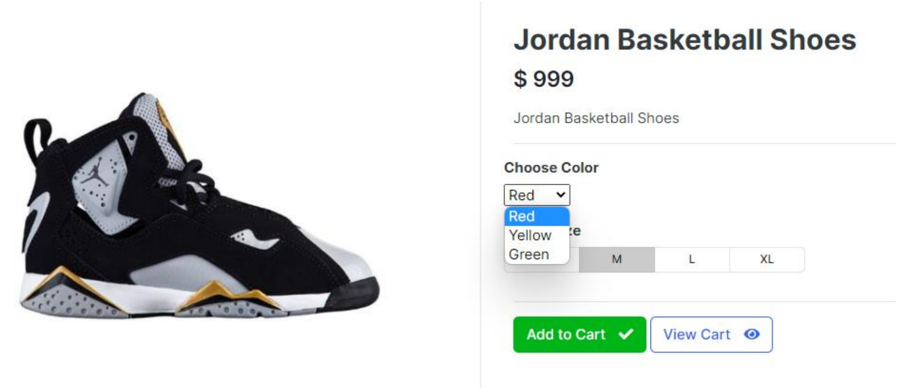
```
<form action="{% url 'add_cart' single_product.id %}" method="GET">
```

171. Remove the lines in the product_detail.html under store folder under templates folder (color selection)

```
<div class="btn-group btn-group-sm btn-group-toggle" data-toggle="buttons">
  <label class="btn btn-light">
    <input type="radio" name="radio_color"> Silver
  </label>
  <label class="btn btn-light">
    <input type="radio" name="radio_color" > Gray
  </label>
  <label class="btn btn-light active">
    <input type="radio" name="radio_color checked"> Gold
  </label>
  <label class="btn btn-light">
    <input type="radio" name="radio_color"> Black
  </label>
</div>
```

172. Add the lines in the product_detail.html under store folder under templates folder (color selection)

```
<div class="item-option-select">
  <h6>Choose Color</h6>
  <select name ="color">
    <option value="red">Red</option>
    <option value="yellow">Yellow</option>
    <option value="green">Green</option>
   </select>
</div>
```



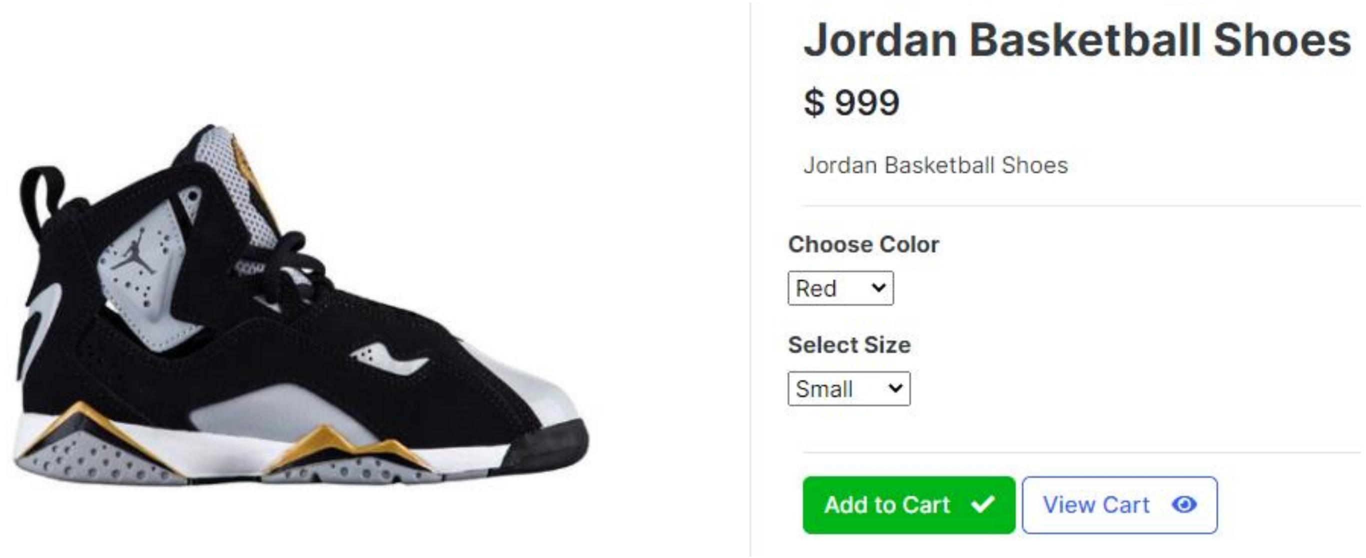
173. Remove the lines in the product_detail.html under store folder under templates folder (size selection)

```
<div class="btn-group btn-group-sm btn-group-toggle" data-toggle="buttons">
  <label class="btn btn-light">
    <input type="radio" name="radio_color"> S
  </label>
  <label class="btn btn-light active">
    <input type="radio" name="radio_color" checked> M
  </label>
  <label class="btn btn-light">
```

```
                                          <input type="radio" name="radio_color"> L
                                        </label>
                                        <label class="btn btn-light">
                                          <input type="radio" name="radio_color"> XL
                                        </label>
                                      </div>
```

174. Add the lines in the product_detail.html under store folder under templates folder (size selection)

```
<div class="item-option-select">
  <h6>Select Size</h6>
  <select name ="size">
    <option value="small">Small</option>
    <option value="medium">Medium</option>
    <option value="large">Large</option>
  </select>
</div>
```



Jordan Basketball Shoes

$ 999

Jordan Basketball Shoes

Choose Color

Red

Select Size

Small

175. Replace the codes in the product_detail.html under store folder under templates folder

Old

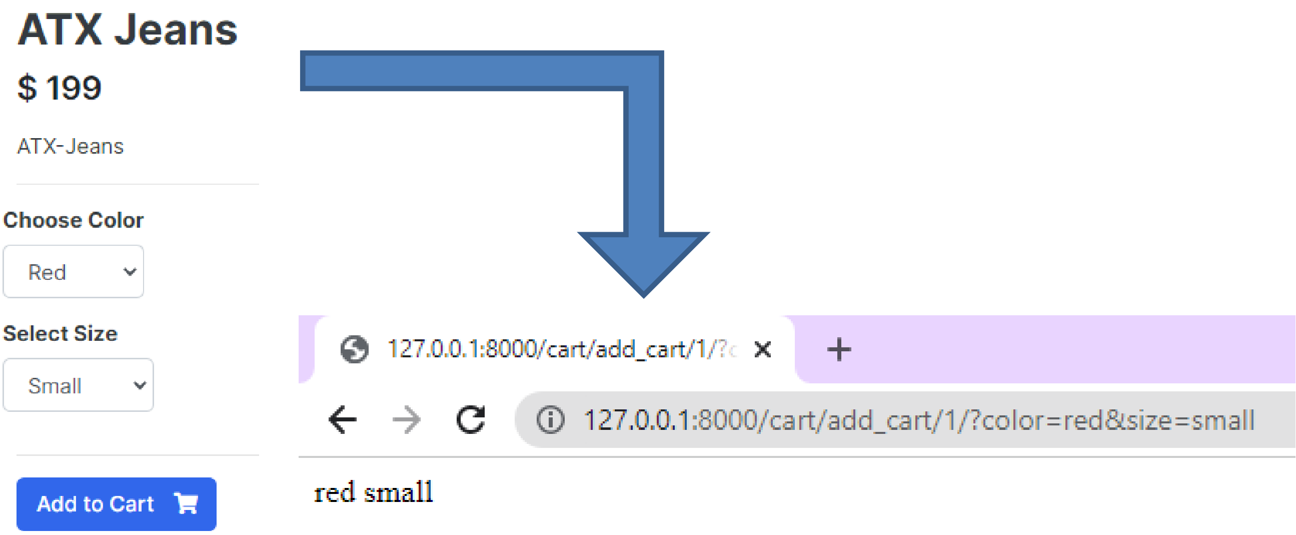
```
<select name ="color">
<select name ="size">
```

New

```
<select name ="size" class="form-control">
<select name ="color" class="form-control">
```

176. Add lines in the views.py from carts folder

```
from django.http import HttpResponse
&
def add_cart(request, product_id):
  color = request.GET['color']
  size = request.GET['size']
  return HttpResponse(color + ' ' + size)
  exit()
```



ATX Jeans

$ 199

ATX-Jeans

Choose Color: Red

Select Size: Small

127.0.0.1:8000/cart/add_cart/1/?color=red&size=small

red small

177. Add lines in the models.py in the store folder

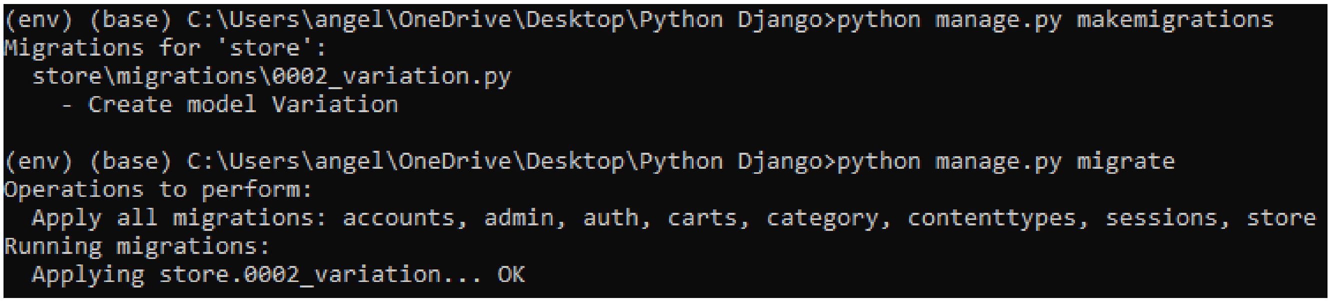
```
    def __str__(self):
        return self.product_name
variation_category_choice = (
    ('color','color'),
    ('size','size'),
)
class Variation (models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    variation_category = models.CharField(max_length=100, choices=variation_category_choice)
    variation_value = models.CharField(max_length=100)
    is_active = models.BooleanField(default=True)
    created_date = models.DateTimeField(auto_now=True)
    def __unicode__(self):
        return self.product
```

178. Add lines in the admin.py in the store folder

```
from .models import Product, Variation
admin.site.register(Variation)
```

179. Migrate the changes thru CMD

```
python manage.py makemigrations
python manage.py migrate
```



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py makemigrations
Migrations for 'store':
  store\migrations\0002_variation.py
    - Create model Variation

(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py migrate
Operations to perform:
  Apply all migrations: accounts, admin, auth, carts, category, contenttypes, sessions, store
Running migrations:
  Applying store.0002_variation... OK
```

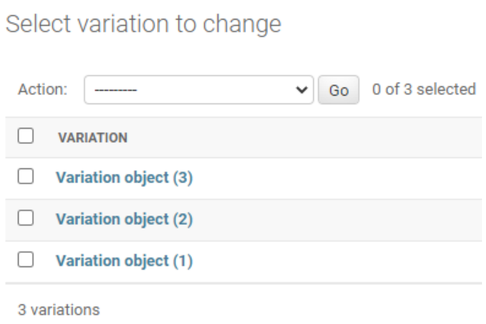
180. Log in admin page and ADD VARIATION in the Variations under STORE and press SAVE

Product: ATX Jeans

Variation category: color

Variation value: Red

Repeat the loop for Blue and Green



Select variation to change

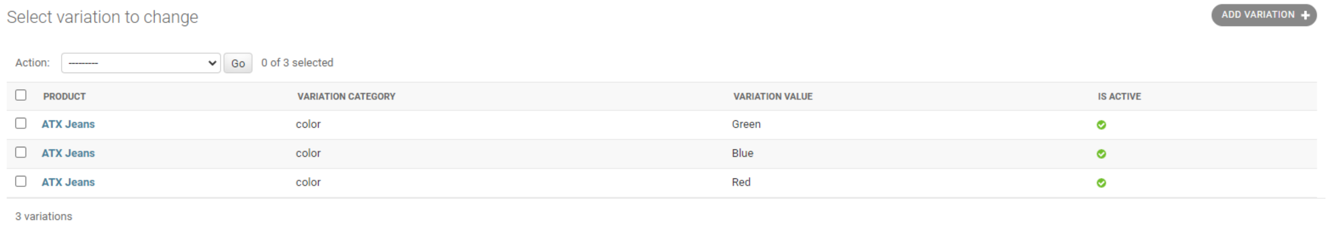
0 of 3 selected

□	VARIATION
□	Variation object (3)
□	Variation object (2)
□	Variation object (1)

3 variations

181. Add lines in the admin.py under store folder

```
class VariationAdmin(admin.ModelAdmin):
    list_display =('product', 'variation_category', 'variation_value', 'is_active')
admin.site.register(Product, ProductAdmin)
admin.site.register(Variation, VariationAdmin)
```



Select variation to change

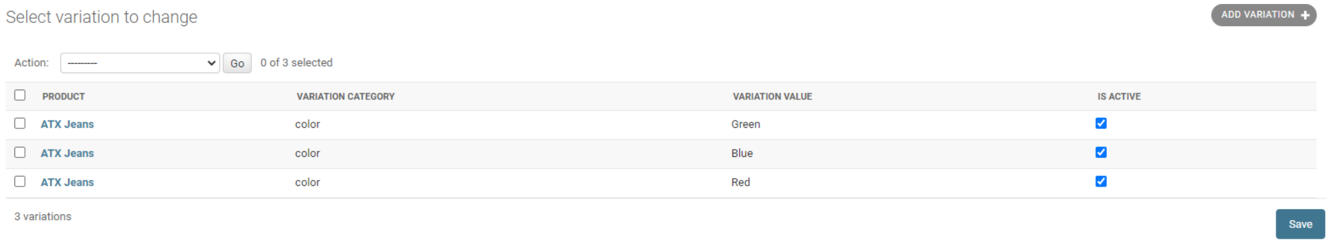
0 of 3 selected

□	PRODUCT	VARIATION CATEGORY	VARIATION VALUE	IS ACTIVE
□	ATX Jeans	color	Green	✓
□	ATX Jeans	color	Blue	✓
□	ATX Jeans	color	Red	✓

3 variations

182. Add 1 line in the admin.py under store folder

```
    list_display =('product', 'variation_category', 'variation_value', 'is_active')
    list_editable = ('is_active',)
```



Select variation to change

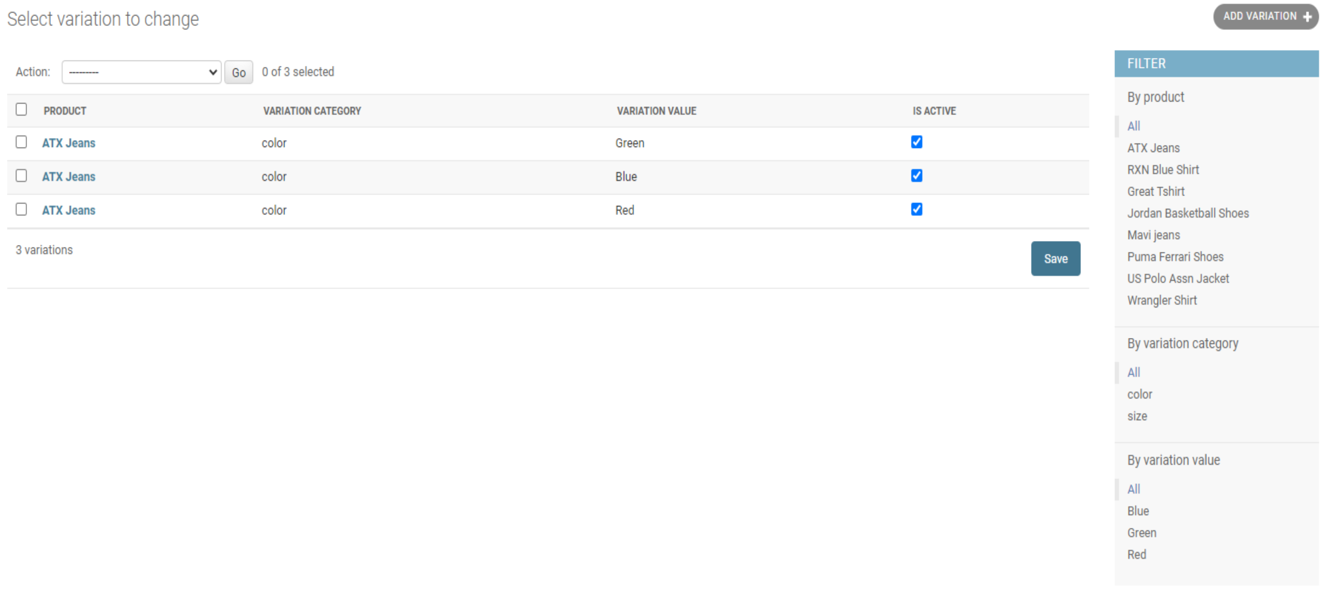
0 of 3 selected

□	PRODUCT	VARIATION CATEGORY	VARIATION VALUE	IS ACTIVE
□	ATX Jeans	color	Green	☑
□	ATX Jeans	color	Blue	☑
□	ATX Jeans	color	Red	☑

3 variations

183. Add 1 line in the admin.py under store folder

```
    list_editable = ('is_active',)
    list_filter =  ('product', 'variation_category', 'variation_value')
```



Select variation to change

0 of 3 selected

□	PRODUCT	VARIATION CATEGORY	VARIATION VALUE	IS ACTIVE
□	ATX Jeans	color	Green	☑
□	ATX Jeans	color	Blue	☑
□	ATX Jeans	color	Red	☑

3 variations

FILTER

By product

All
ATX Jeans
RXN Blue Shirt
Great Tshirt
Jordan Basketball Shoes
Mavi jeans
Puma Ferrari Shoes
US Polo Assn Jacket
Wrangler Shirt

By variation category

All
color
size

By variation value

All
Blue
Green
Red

184. Edit lines in the product_detail.html under store folder in template folder

Old

```
<select name ="color" class="form-control">
  <option value="red">Red</option>
  <option value="yellow">Yellow</option>
  <option value="green">Green</option>
</select>
```

New

```
<select name ="color" class="form-control">
  {% for i in single_product.variation_set.all %}
  <option value="{{ i.variation_value }}">{{ i.variation_value }}</option>
  {% endfor %}
</select>
```

& edit and add line as follow

```
<select name ="color" class="form-control" required>
  <option value ="" disabled selected>Choose Color</option>
```

185. Add lines in the models.py under store folder

```
class VariationManager(models.Manager):
  def colors(self):
    return super (VariationManager, self).filter(variation_category='color', is_active=True)
  def sizes(self):
    return super (VariationManager, self).filter(variation_category='size', is_active=True)
```

186. Edit line in the product_detail.html under store folder in template folder

Old

```
          <div class="item-option-select">
                  <h6>Choose Color</h6>
      <select name ="color" class="form-control" required>
        <option value ="" disabled selected>Choose Color</option>
        {% for i in single_product.variation_set.all %}
        <option value="{{ i.variation_value }}">{{ i.variation_value }}</option>
        {% endfor %}
      </select>
          </div>
</div> <!-- row.// -->
<div class="row">
          <div class="item-option-select">
                  <h6>Select Size</h6>
      <select name ="size" class="form-control">
        <option value="small">Small</option>
        <option value="medium">Medium</option>
        <option value="large">Large</option>
      </select>
          </div>
```

New

```
          <div class="item-option-select">
                  <h6>Choose Color</h6>
      <select name ="color" class="form-control" required>
        <option value ="" disabled selected>Choose Color</option>
        {% for i in single_product.variation_set.colors %}
        <option value="{{ i.variation_value | lower }}">{{ i.variation_value | capfirst }}</option>
```

```
                    {% endfor %}
                </select>
                    </div>
            </div> <!-- row.// -->
            <div class="row">
                    <div class="item-option-select">
                            <h6>Select Size</h6>
                <select name ="size" class="form-control">
                  {% for i in single_product.variation_set.sizes %}
                  <option value="{{ i.variation_value | lower }}">{{ i.variation_value | capfirst }}</option>
                  {% endfor %}
                </select>
                    </div>
```

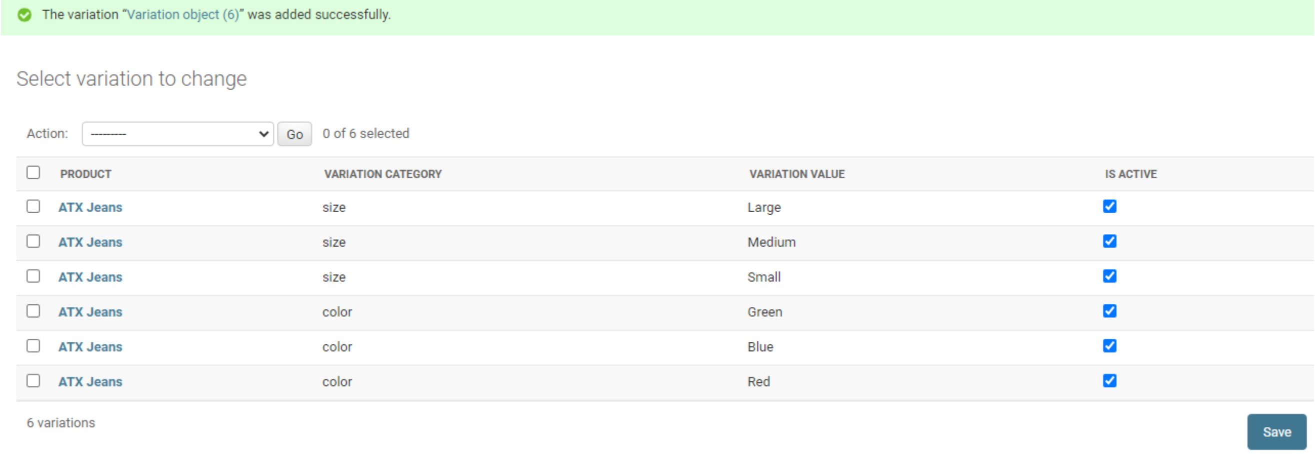
187. Insert 1 line in the models.py in the store folder

```
objects = VariationManager()
def __str__(self):
  return self.variation_value
```

188. Edit line in the product_detail.html under store folder in template folder

```
<select name ="size" class="form-control" required>
  <option value ="" disabled selected>Choose Size</option>
```

189. Add size variation in admin page as follow



The variation "Variation object (6)" was added successfully.

Select variation to change

Action: --------- Go 0 of 6 selected

□	PRODUCT	VARIATION CATEGORY	VARIATION VALUE	IS ACTIVE
□	ATX Jeans	size	Large	☑
□	ATX Jeans	size	Medium	☑
□	ATX Jeans	size	Small	☑
□	ATX Jeans	color	Green	☑
□	ATX Jeans	color	Blue	☑
□	ATX Jeans	color	Red	☑

6 variations

Save

190. Edit codes in the product_detail.html under store folder in template folder

Old

```
<form action="{% url 'add_cart' single_product.id %}" method="GET">
```

New

```
<form action="{% url 'add_cart' single_product.id %}" method="POST">
    {% csrf_token %}
```

191. Edit codes in the views.py under carts folder

Old

```
def add_cart(request, product_id):
  color = request.GET['color']
  size = request.GET['size']
  return HttpResponse(color + ' ' + size)
  exit()
```

New

```
def add_cart(request, product_id):
  if request.method == 'POST':
        color = request.POST['color']
        size = request.POST['size']
```

192. Replace codes in the views.py under carts folder

Old

```
    if request.method == 'POST':
        color = request.POST['color']
        size = request.POST['size']
```

New

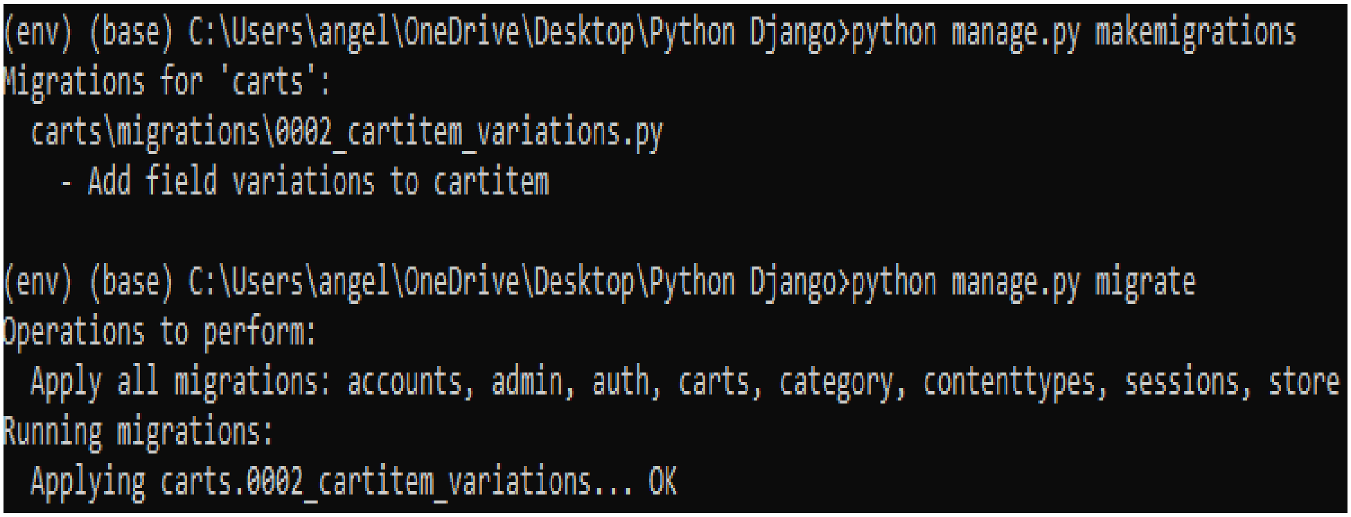
```
from store.models import Product, Variation
&
def add_cart(request, product_id):
    product = Product.objects.get(id=product_id) #get the product
    product_variation = []
    if request.method == 'POST':
        for item in request.POST:
            key = item
            value = request.POST[key]

            try:
                variation = Variation.objects.get(product=product, variation_category__iexact=key,
variation_value__iexact=value)
                product_variation.append(variation)
            except:
                pass
```

193. Add line in the models.py under carts folder

```
from store.models import Product, Variation
&
variations = models.ManyToManyField(Variation, blank=True)
&
    def __unicode__(self):
        return self.product
```

194. Migrate the changes thru CMD

```
python manage.py makemigrations
python manage.py migrate
```



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py makemigrations
Migrations for 'carts':
  carts\migrations\0002_cartitem_variations.py
    - Add field variations to cartitem

(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>python manage.py migrate
Operations to perform:
  Apply all migrations: accounts, admin, auth, carts, category, contenttypes, sessions, store
Running migrations:
  Applying carts.0002_cartitem_variations... OK
```

195. Edit admin.py under carts folder

```
from django.contrib import admin
from .models import Cart, CartItem
# Register your models here.
class CartAdmin(admin.ModelAdmin):
    list_display = ('cart_id' , 'date_added')
class CartItemAdmin(admin.ModelAdmin):
    list_display = ('product' , 'cart' , 'quantity','is_active')
admin.site.register(Cart)
admin.site.register(CartItem, CartItemAdmin)
```

196. Add comments and codes in the view.py under carts folder

```
    #product variation codes
    product = Product.objects.get(id=product_id) #get the product
&
    #cart codes
    try:
        cart = Cart.objects.get(cart_id=_cart_id(request)) #get the cart using the cart_id present in the session
&
    #cart item codes
    try:
        cart_item = CartItem.objects.get(product=product, cart=cart)
&
        cart_item = CartItem.objects.get(product=product, cart=cart)
        if len(product_variation) > 0:
            for item in product_variation:
                cart_item.variations.add(item)
        cart_item.quantity += 1 # cart_item.quantity = cart_item.quantity + 1
&
            cart = cart,
        )
        if len(product_variation) > 0:
            for item in product_variation:
                cart_item.variations.add(item)
        cart_item.save()
    return redirect('cart')
```

197. Edit the lines in product_detail.html under store folder in template folder

Old

```
            {% if single_product.stock <= 0 %}
            <h5 class="text-danger">Out of Stock</h5>
            {% else %}
            {% if in_cart %}
            <a href="#" class="btn  btn-success"> <span class="text">Add to Cart</span> <i class="fas fa-check"></i>  </a>
            <a href="{% url 'cart' %}" class="btn  btn-outline-primary"> <span class="text">View Cart</span> <i class="fas fa-eye"></i>  </a>
           {% else %}
```

```
            <button type="submit" class="btn  btn-primary"> <span class="text">Add to Cart</span> <i class="fas fa-shopping-cart"></i>  </button>
           {% endif %}
         {% endif %}
```

New

```
            {% if single_product.stock <= 0 %}
            <h5 class="text-danger">Out of Stock</h5>
            {% else %}
            <button type="submit" class="btn  btn-primary"> <span class="text">Add to Cart</span> <i class="fas fa-shopping-cart"></i>  </button>
         {% endif %}
```

198. Edit views.py under carts folder

```
      if len(product_variation) > 0:
        cart_item.variations.clear()
        for item in product_variation:
          cart_item.variations.add(item)

      cart_item.quantity += 1 # cart_item.quantity = cart_item.quantity + 1
      cart_item.save()
    except CartItem.DoesNotExist:
      cart_item = CartItem.objects.create(
        product = product,
        quantity = 1,
        cart = cart,
      )
      if len(product_variation) > 0:
        cart_item.variations.clear()
        for item in product_variation:
```

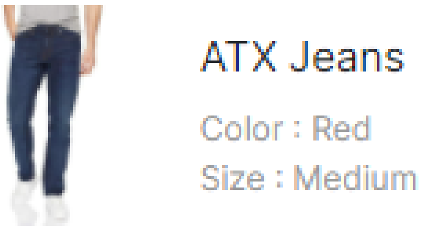
199. Edit cart.html under store folder under templates folder

Old

```
<p class="text-muted small">Matrix: 25 Mpx <br> Brand: Canon</p>
```

New

```
            <p class="text-muted small">
                {% if cart_item.variations.all %}
                    {% for item in cart_item.variations.all %}
                    {{ item.variation_category | capfirst }} : {{ item.variation_value | capfirst }} <br>
                    {% endfor %}
                {% endif %}
            </p>
```

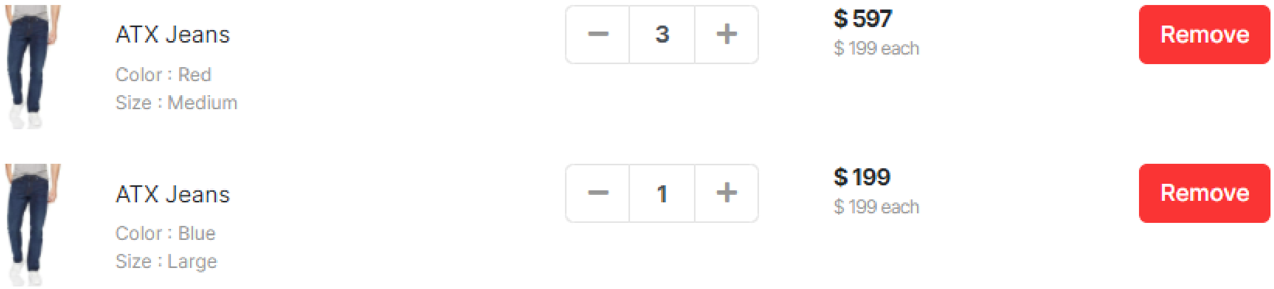


ATX Jeans

Color : Red
Size : Medium

200. Edit views.py under carts folder

```
Old
  #cart item codes
  try:
    cart_item = CartItem.objects.get(product=product, cart=cart)
New
  #cart item codes
  try:
    cart_item = CartItem.objects.create(product=product, quantity=1, cart=cart)
    if len(product_variation) > 0:
      cart_item.variations.clear()
      for item in product_variation:
        cart_item.variations.add(item)
    #cart_item.quantity += 1 # cart_item.quantity = cart_item.quantity + 1
```



ATX Jeans
Color : Red
Size : Medium
− 3 +
$ 597
$ 199 each
Remove

ATX Jeans
Color : Blue
Size : Large
− 1 +
$ 199
$ 199 each
Remove

201. Edit codes in views.py under carts folder

```
#cart item codes
is_cart_item_exists = CartItem.objects.filter(product=product, cart=cart).exists()
if is_cart_item_exists:
  cart_item = CartItem.objects.filter(product=product, cart=cart)
  # existing_variation -> database
  # current variation -> product_variation
  # item_id -> database
  ex_var_list = []
  id = []
  for item in cart_item:
    existing_variation = item.variations.all()
    ex_var_list.append(list(existing_variation))
    id.append(item.id)
  print(ex_var_list)
  if product_variation in ex_var_list:
    # increase the cart item quantity
    index = ex_var_list.index(product_variation)
    item_id = id[index]
    item = CartItem.objects.get(product=product, id=item_id)
    item.quantity += 1
    item.save()
  else:
    item = CartItem.objects.create(product=product, quantity=1, cart=cart)
    if len(product_variation) > 0:
```

```
                item.variations.clear()
                item.variations.add(*product_variation)
            item.save()
    else:
        cart_item = CartItem.objects.create(
            product = product,
            quantity = 1,
            cart = cart,
        )
        if len(product_variation) > 0:
            cart_item.variations.clear()
            cart_item.variations.add(*product_variation)
        cart_item.save()
    return redirect('cart')
```

The - + remove button not working.

202. Edit codes in the cart.html in the store folder under the templates folder

```
                        <div class="input-group-append">
                        <form action="{% url 'add_cart' cart_item.product.id %}" method="POST">
                        {% csrf_token %}
                        {% for item in cart_item.variations.all %}
                            <input type="hidden" name="{{ item.variation_category | lower }}" value="{{ item.variation_value | capfirst }}">
        {% endfor %}
        <button class="btn btn-light" type="submit" id="button-minus"> <i class="fa fa-plus"></i> </button>
       </form>
                        </div>
```

User able to add item no in the cart page

203. Edit views.py in the carts folder

```
def remove_cart(request, product_id, cart_item_id):
```

204. Edit codes in the cart.html in the store folder under the templates folder

```
<a href="{% url 'remove_cart' cart_item.product.id cart_item.id %}" class="btn btn-light" type="button" id="button-plus"> <i class="fa fa-minus"></i> </a>
```

205. Edit codes in the urls.py under carts folder

```
path('remove_cart/<int:product_id>/<int:cart_item_id>/', views.remove_cart, name='remove_cart'),
```

206. Edit views.py in the carts folder

```
def remove_cart(request, product_id, cart_item_id):
    cart = Cart.objects.get(cart_id=_cart_id(request))
    product = get_object_or_404(Product, id=product_id)
    try:
        cart_item = CartItem.objects.get(product=product, cart=cart, id=cart_item_id)
        if cart_item.quantity > 1:
            cart_item.quantity -= 1
            cart_item.save()
        else:
            cart_item.delete()
    except:
        pass
```

return redirect('cart')

User able to decrease the no of item in the cart page

207. Edit codes in the cart.html in the store folder under the templates folder

```
<a href="{% url 'remove_cart_item' cart_item.product.id cart_item.id %}" class="btn btn-danger"> Remove</a>
```

208. Edit codes in the urls.py under carts folder

```
path('remove_cart_item/<int:product_id>/<int:cart_item_id>/', views.remove_cart_item,
name='remove_cart_item'),
```

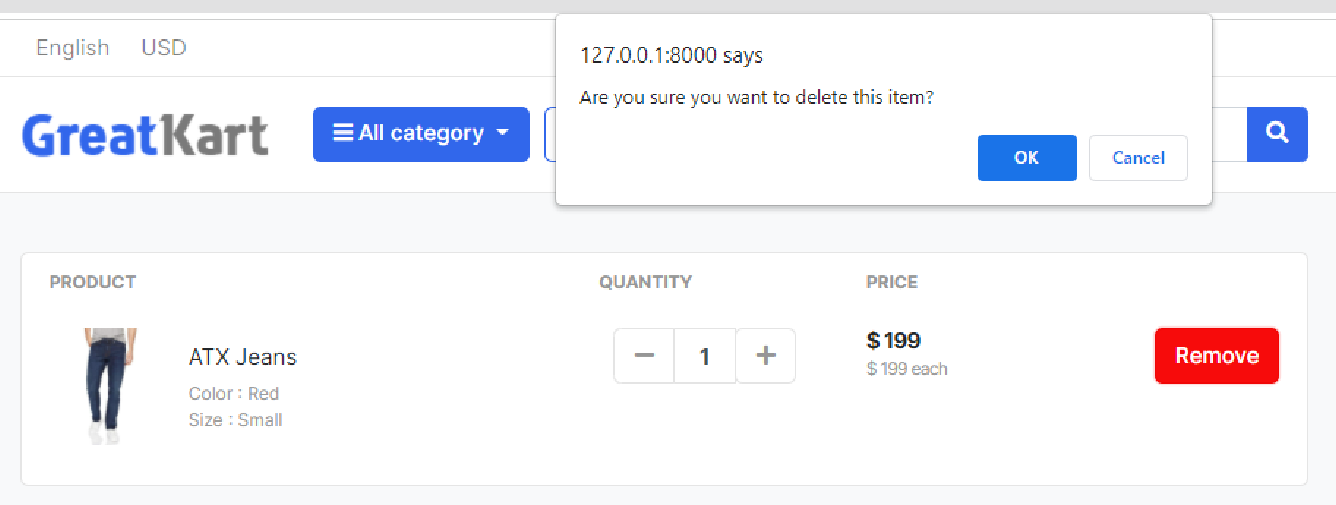
209. Edit views.py in the carts folder

```
def remove_cart_item(request, product_id, cart_item_id):
    cart = Cart.objects.get(cart_id=_cart_id(request))
    product = get_object_or_404(Product, id=product_id)
    cart_item = CartItem.objects.get(product=product, cart=cart, id=cart_item_id)
    cart_item.delete()
    return redirect('cart')
```

user able to remove item in the cart page

210. Edit code in the cart.html in the store folder under the templates folder

```
<a href="{% url 'remove_cart_item' cart_item.product.id cart_item.id %}" onclick="return confirm('Are you sure you want to delete this item?')" class="btn btn-danger"> Remove</a>
```

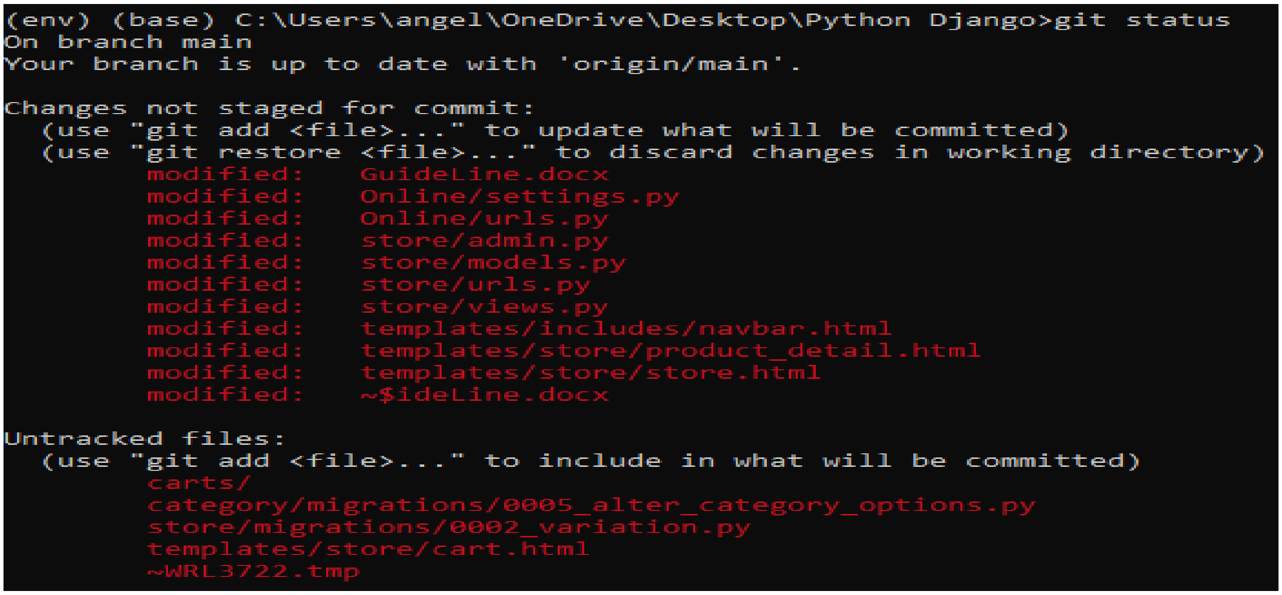


127.0.0.1:8000 says

Are you sure you want to delete this item?

211. Push Code to GitHub on CMD

git status

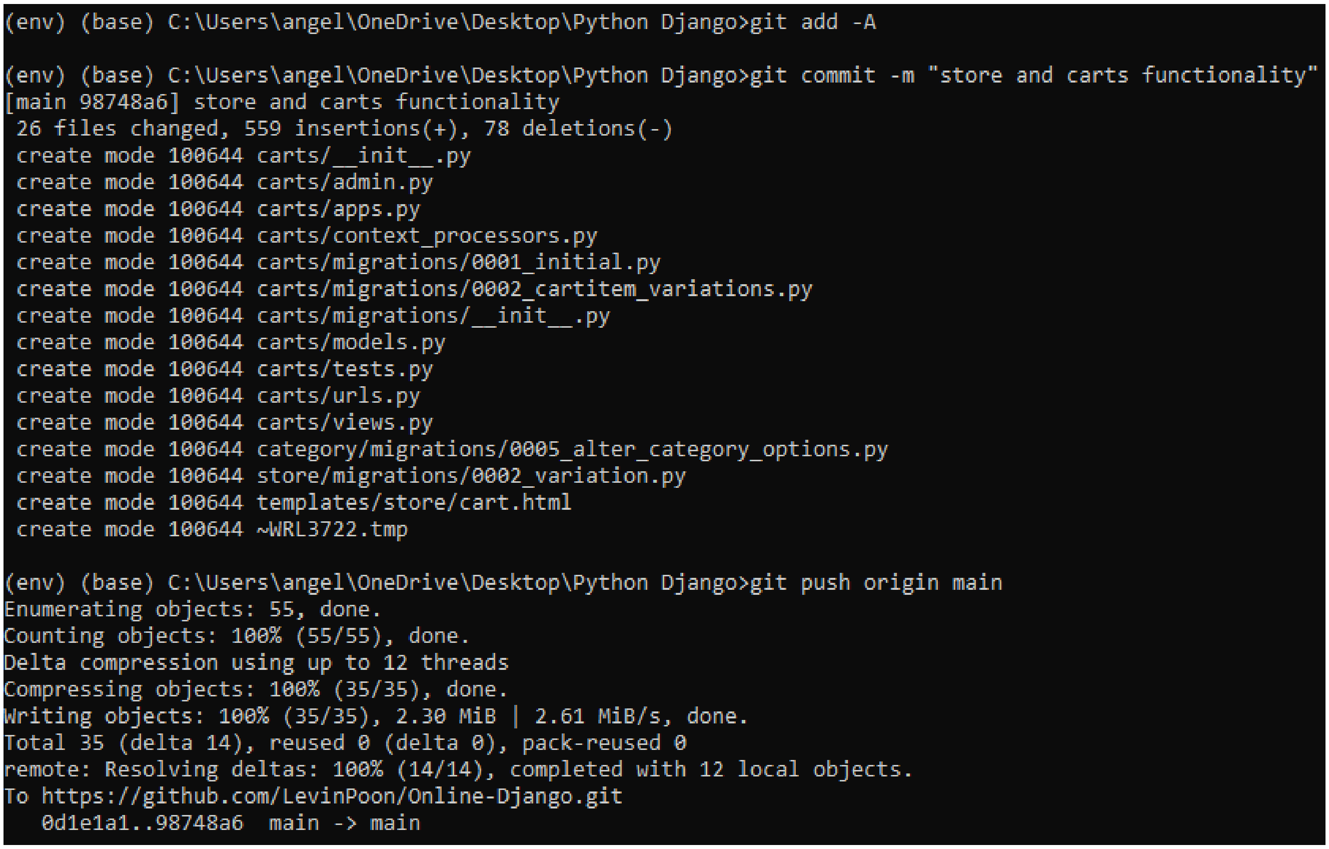


```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   GuideLine.docx
        modified:   Online/settings.py
        modified:   Online/urls.py
        modified:   store/admin.py
        modified:   store/models.py
        modified:   store/urls.py
        modified:   store/views.py
        modified:   templates/includes/navbar.html
        modified:   templates/store/product_detail.html
        modified:   templates/store/store.html
        modified:   ~$ideLine.docx

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        carts/
        category/migrations/0005_alter_category_options.py
        store/migrations/0002_variation.py
        templates/store/cart.html
        ~WRL3722.tmp
```

git add –A

git commit -m "store and carts functionality"

git push origin main



```
(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git add -A

(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git commit -m "store and carts functionality"
[main 98748a6] store and carts functionality
 26 files changed, 559 insertions(+), 78 deletions(-)
 create mode 100644 carts/__init__.py
 create mode 100644 carts/admin.py
 create mode 100644 carts/apps.py
 create mode 100644 carts/context_processors.py
 create mode 100644 carts/migrations/0001_initial.py
 create mode 100644 carts/migrations/0002_cartitem_variations.py
 create mode 100644 carts/migrations/__init__.py
 create mode 100644 carts/models.py
 create mode 100644 carts/tests.py
 create mode 100644 carts/urls.py
 create mode 100644 carts/views.py
 create mode 100644 category/migrations/0005_alter_category_options.py
 create mode 100644 store/migrations/0002_variation.py
 create mode 100644 templates/store/cart.html
 create mode 100644 ~WRL3722.tmp

(env) (base) C:\Users\angel\OneDrive\Desktop\Python Django>git push origin main
Enumerating objects: 55, done.
Counting objects: 100% (55/55), done.
Delta compression using up to 12 threads
Compressing objects: 100% (35/35), done.
Writing objects: 100% (35/35), 2.30 MiB | 2.61 MiB/s, done.
Total 35 (delta 14), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (14/14), completed with 12 local objects.
To https://github.com/LevinPoon/Online-Django.git
   0d1e1a1..98748a6  main -> main
```

212. Create urls.py in the accounts folder and add lines

```
from django.urls import path
from . import views
urlpatterns = [
    path('register/', views.register, name='register'),
    path('login/', views.login, name='login'),
    path('logout/', views.logout, name='logout'),
]
```

213. Add line in the urls.py in the online folder

```
    path('cart/', include('carts.urls')),
    path('accounts/', include('accounts.urls')),
```

214. Add lines in the views.py under accounts folder

```
from django.shortcuts import render
def register(request):
    return render(request, 'accounts/register.html')
def login(request):
    return render(request, 'accounts/login.html')
def logout(request):
    return
```

215. Create accounts folder in the templates folder

216. Create and add lines in register.html under accounts folder under templates folder

```
{% extends 'base.html' %}
{% block content %}
```

{% endblock %}

217. Create login.html under accounts folder under templates folder

{% extends 'base.html' %}
{% block content %}
{% endblock %}

218. Edit codes in navbar.html under includes folder under templates folder (Register page)

Old

<a href="./signin.html">Sign in</a> <span class="dark-transp">
| </span>
<a href="./register.html"> Register</a>

New

<a href="{% url 'login' %}">Sign in</a> <span class="dark-transp"> | </span>
<a href="{% url 'register' %}"> Register</a>

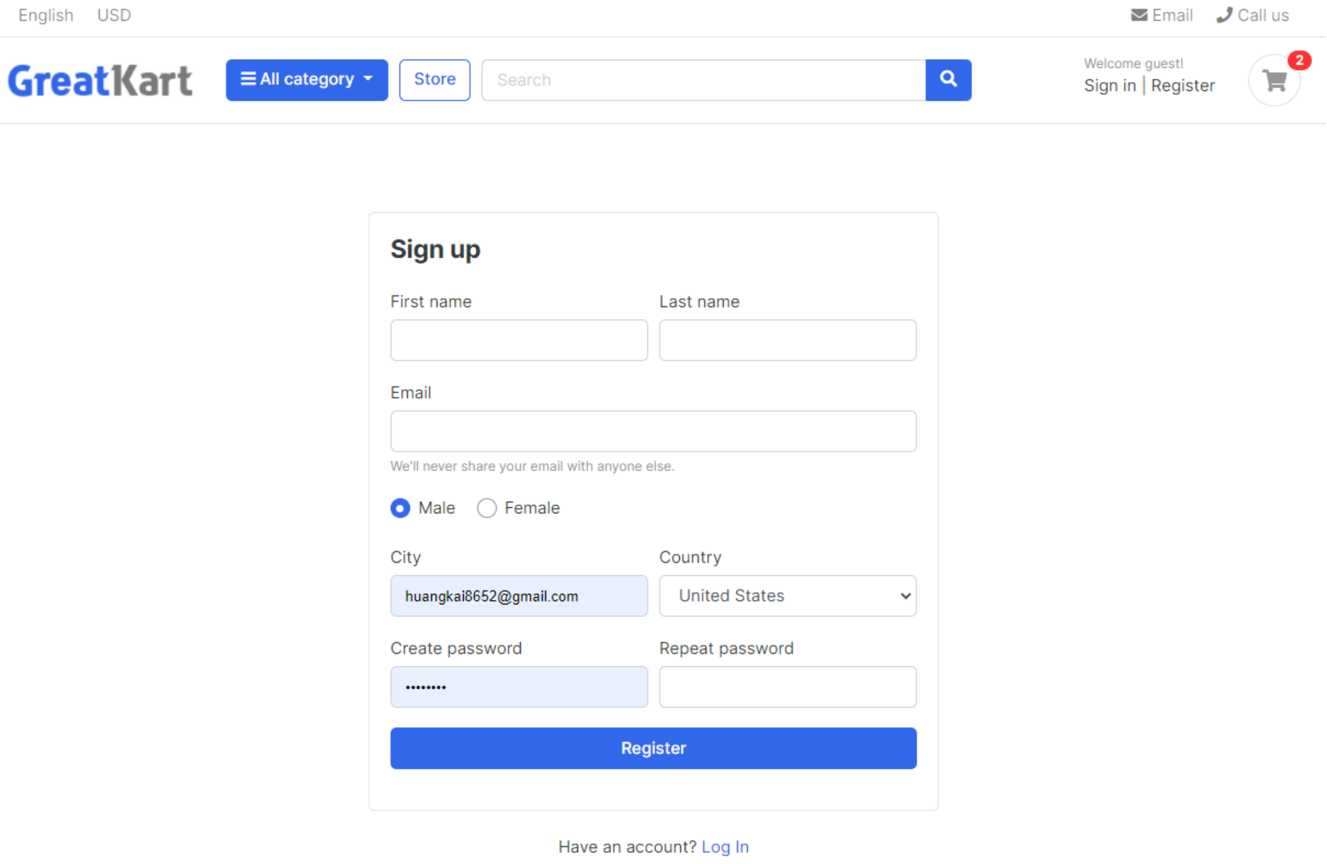
219. Copy the codes from register.html under sample project folder to register.html under accounts folder under templates folder

Copy

<!-- ========================= SECTION CONTENT ========================= -->
…
<!-- ========================= SECTION CONTENT END// ========================= -->

Paste between

{% block content %}
…
{% endblock %}



We'll never share your email with anyone else.

● Male ○ Female

City: huangkai8652@gmail.com

Country: United States

Create password: ••••••••

Have an account? Log In

220. 1

221. Create and add lines in the forms.py under accounts folder

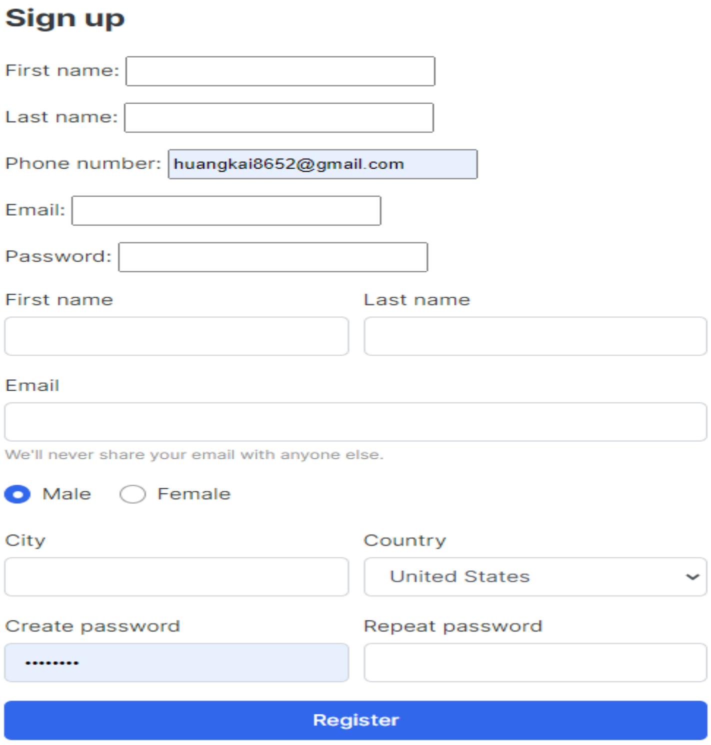
```
from django import forms
from .models import Account
class RegistrationForm (forms.ModelForm):
  class Meta:
    model = Account
    fields = ['first_name','last_name','phone_number','email','password']
```

222. Add line in the views.py under accounts folder

```
from django shortcuts import render.
from .forms import RegistrationForm
def register(request):
  form = RegistrationForm()
  context = {
    'form': form,
  }
  return render(request, 'accounts/register.html',context)

def login(request):
  return render(request, 'accounts/login.html')

def logout(request):
  return
```

223. Add line in the register.html under accounts folder under templates folder

```
<header class="mb-4"><h4 class="card-title">Sign up</h4></header>
<form action="{% url 'register’ %}" method="POST">
{% csrf_token %}
{{ form.as_p }}
              <div class="form-row">
```



Sign up

First name:

Last name:

Phone number: huangkai8652@gmail.com

Email:

Password:

First name

Last name

Email

We'll never share your email with anyone else.

● Male ○ Female

City

Country: United States

Create password: ········

Repeat password

Register

224. Edit line in the register.html under accounts folder under templates folder

Remove this line

```
{{ form.as_p }}
```

Remove these line

```
<div class="form-group">
        <label class="custom-control custom-radio custom-control-inline">
          <input class="custom-control-input" checked="" type="radio" name="gender" value="option1">
          <span class="custom-control-label"> Male </span>
        </label>
        <label class="custom-control custom-radio custom-control-inline">
          <input class="custom-control-input" type="radio" name="gender" value="option2">
          <span class="custom-control-label"> Female </span>
        </label>
</div> <!-- form-group end.// -->
```

Old

```
<label>First name</label>
<input type="text" class="form-control" placeholder="">
```

&

```
<label>Last name</label>
<input type="text" class="form-control" placeholder="">
```

New

```
<label>First name</label>
  {{ form.first_name }}
```

&

```
<label>Last name</label>
  {{ form.last_name }}
```

Old

```
<label>City</label>
<input type="text" class="form-control">
```

New

```
<label>Email Address</label>
{{ form.email }}
```

Remove the email codes

```
<div class="form-group">
        <label>Email</label>
        <input type="email" class="form-control" placeholder="">
        <small class="form-text text-muted">We'll never share your email with anyone else.</small>
</div> <!-- form-group end.// -->
```

&

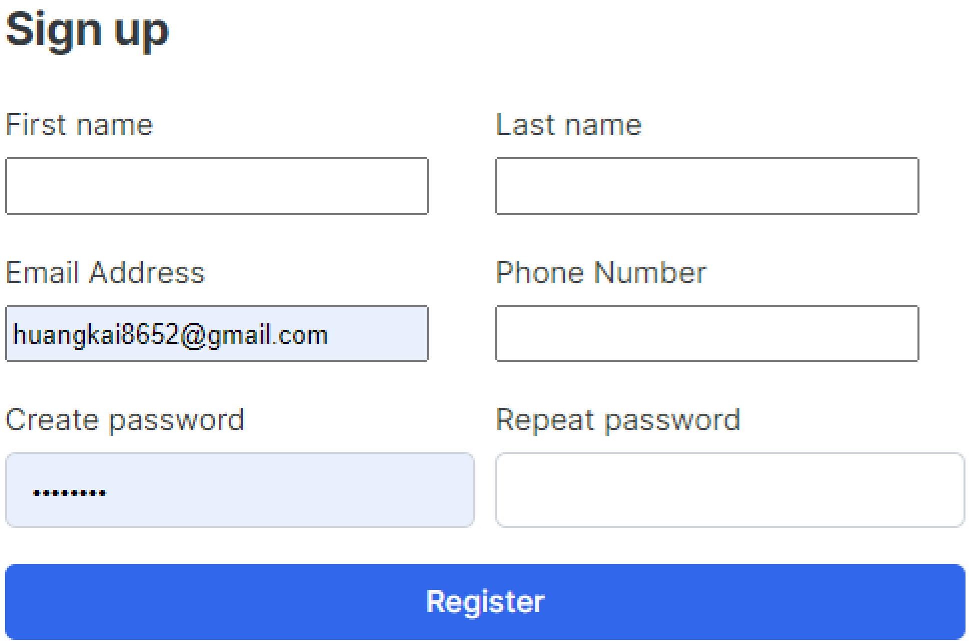
```
<select id="inputState" class="form-control">
  <option> Choose...</option>
    <option>Uzbekistan</option>
    <option>Russia</option>
    <option selected="">United States</option>
    <option>India</option>
    <option>Afganistan</option>
</select>
```

Old

```
<label>Country</label>
```

New

```
<label>Phone Number</label>
{{ form.phone_number }}
```



huangkai8652@gmail.com

225. Edit forms.py in the accounts

```
from django import forms
from .models import Account
class RegistrationForm (forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput(attrs={
        'placeholder':'Enter Password',
    }))
    class Meta:
        model = Account
        fields = ['first_name','last_name','phone_number','email','password']
```

226. Edit line in the register.html under accounts folder under templates folder

Old

```
<label> Create password</label>
 <input class="form-control" type="password">
```

New

```
<label>Create password</label>
{{ form.password }}
```

227. Edit forms.py in the accounts

```
    }))
    confirm_password = forms.CharField(widget=forms.PasswordInput(attrs={
        'placeholder':'Confirm Password'
    }))
    class Meta:
```

228. Edit line in the register.html under accounts folder under templates folder

Old

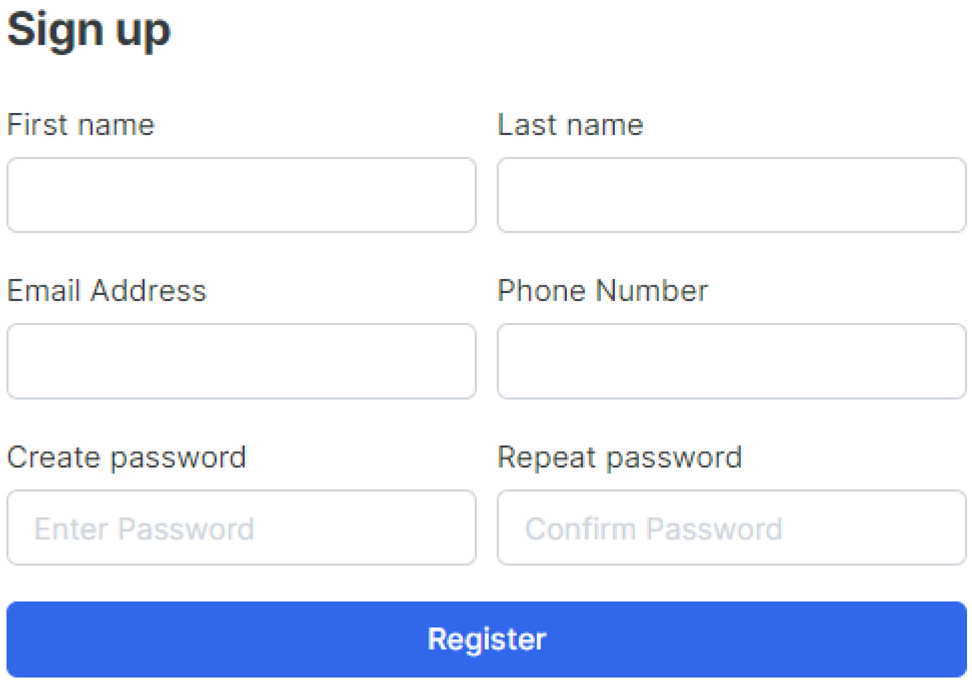
```
<label>Repeat password</label>
<input class="form-control" type="password">
```

New

```
<label>Repeat password</label>
{{ form.confirm_password }}
```

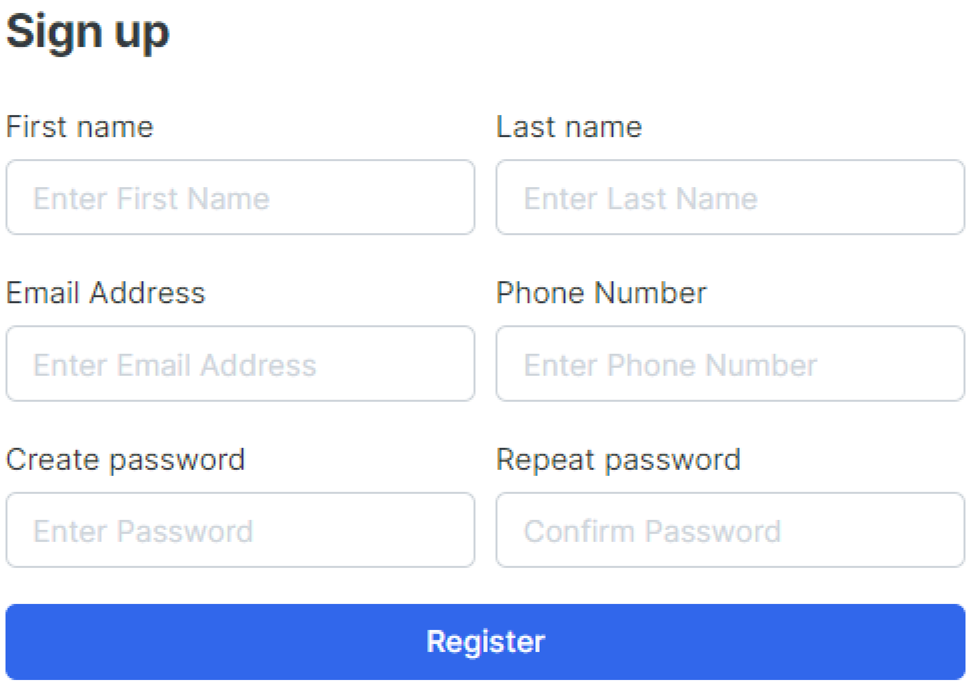
229. Edit forms.py in the accounts

```
        'placeholder':'Enter Password',
        'class':'form-control',
    }))
    confirm_password = forms.CharField(widget=forms.PasswordInput(attrs={
        'placeholder':'Confirm Password'
    }))
    class Meta:
        model = Account
        fields = ['first_name','last_name','phone_number','email','password']

    def __init__(self, *args, **kwargs):
        super (RegistrationForm, self).__init__(*args, **kwargs)
        for field in self.fields:
            self.fields[field].widget.attrs['class'] = 'form-control'
```



230. Edit forms.py in the accounts

```
    def __init__(self, *args, **kwargs):
        super (RegistrationForm, self).__init__(*args, **kwargs)
        self.fields['first_name'].widget.attrs['placeholder'] = 'Enter First Name'
        self.fields['last_name'].widget.attrs['placeholder'] = 'Enter Last Name'
        self.fields['phone_number'].widget.attrs['placeholder'] = 'Enter Phone Number'
        self.fields['email'].widget.attrs['placeholder'] = 'Enter Email Address'
    for field in self.fields:
        self.fields[field].widget.attrs['class'] = 'form-control'
```



Sign up

First name

Enter First Name

Last name

Enter Last Name

Email Address

Enter Email Address

Phone Number

Enter Phone Number

Create password

Enter Password

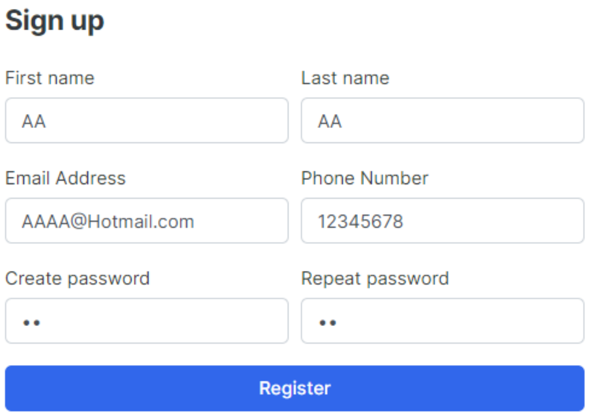
Repeat password

Confirm Password

231. Edit codes in the views.py under accounts folder

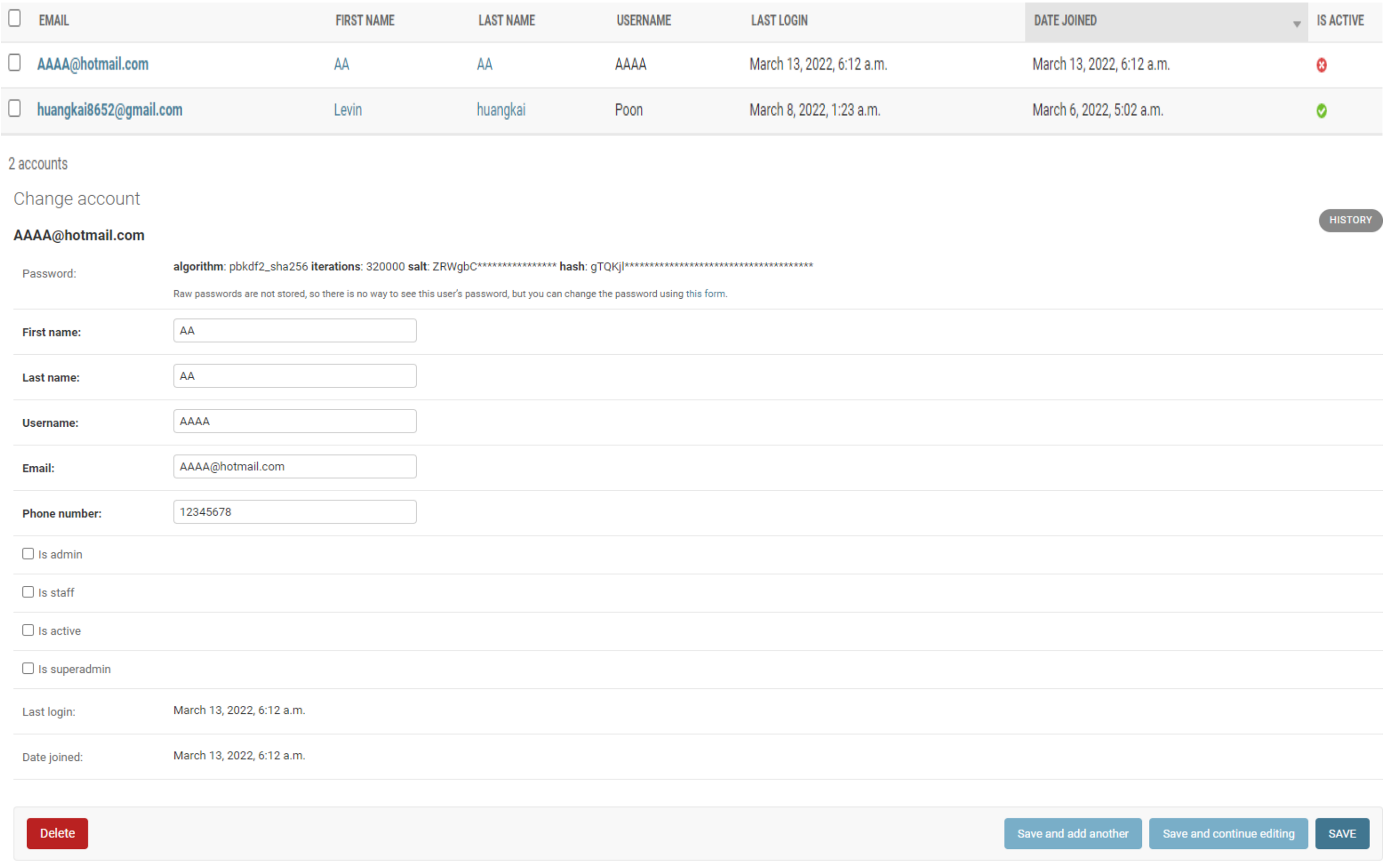
```
from django.shortcuts import render
from .forms import RegistrationForm
from .models import Account

def register(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            first_name = form.cleaned_data['first_name']
            last_name = form.cleaned_data['last_name']
            phone_number = form.cleaned_data['phone_number']
            email = form.cleaned_data['email']
            password = form.cleaned_data['password']
            username = email.split("@")[0]
            user = Account.objects.create_user(first_name=first_name,last_name=last_name, email=email, username=username, password=password)
            user.phone_number = phone_number
            user.save()
    else:
        form = RegistrationForm()
    context = {
        'form': form,
    }
    return render(request, 'accounts/register.html',context)

def login(request):
    return render(request, 'accounts/login.html')

def logout(request):
    return
```

User register a new user



Sign up

First name	Last name
AA	AA
Email Address	Phone Number
AAAA@Hotmail.com	12345678
Create password	Repeat password
••	••



☐	EMAIL	FIRST NAME	LAST NAME	USERNAME	LAST LOGIN	DATE JOINED	IS ACTIVE
☐	AAAA@hotmail.com	AA	AA	AAAA	March 13, 2022, 6:12 a.m.	March 13, 2022, 6:12 a.m.	✗
☐	huangkai8652@gmail.com	Levin	huangkai	Poon	March 8, 2022, 1:23 a.m.	March 6, 2022, 5:02 a.m.	✓

2 accounts

Change account

AAAA@hotmail.com

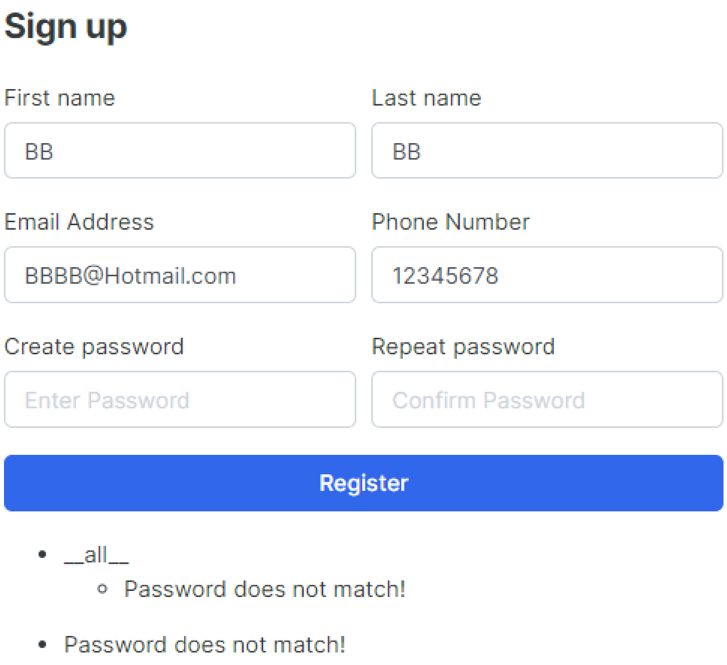
Password:	**algorithm**: pbkdf2_sha256 **iterations**: 320000 **salt**: ZRWgbC*************** **hash**: gTQKjl************************************* Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form.
First name:	AA
Last name:	AA
Username:	AAAA
Email:	AAAA@hotmail.com
Phone number:	12345678
☐ Is admin	
☐ Is staff	
☐ Is active	
☐ Is superadmin	
Last login:	March 13, 2022, 6:12 a.m.
Date joined:	March 13, 2022, 6:12 a.m.

232. Edit codes in the forms.py under accounts folder

```
from django import forms
from .models import Account
class RegistrationForm (forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput(attrs={
        'placeholder':'Enter Password',
        'class':'form-control',
    }))
    confirm_password = forms.CharField(widget=forms.PasswordInput(attrs={
        'placeholder':'Confirm Password'
    }))
    class Meta:
        model = Account
        fields = ['first_name','last_name','phone_number','email','password']

    def clean(self):
        cleaned_data = super(RegistrationForm, self).clean()
        password = cleaned_data.get('password')
        confirm_password = cleaned_data.get('confirm_password')

        if password != confirm_password:
            raise forms.ValidationError(
                "Password does not match!"
            )

    def __init__(self, *args, **kwargs):
        super (RegistrationForm, self).__init__(*args, **kwargs)
        self.fields['first_name'].widget.attrs['placeholder'] = 'Enter First Name'
        self.fields['last_name'].widget.attrs['placeholder'] = 'Enter Last Name'
        self.fields['phone_number'].widget.attrs['placeholder'] = 'Enter Phone Number'
        self.fields['email'].widget.attrs['placeholder'] = 'Enter Email Address'
        for field in self.fields:
            self.fields[field].widget.attrs['class'] = 'form-control'
```

233. Edit line in the register.html under accounts folder under templates folder

```
    </div> <!-- form-group// -->
{{ form.errors }}
{{ form.non_field_errors }}
</form>
```



Sign up

First name: BB

Last name: BB

Email Address: BBBB@Hotmail.com

Phone Number: 12345678

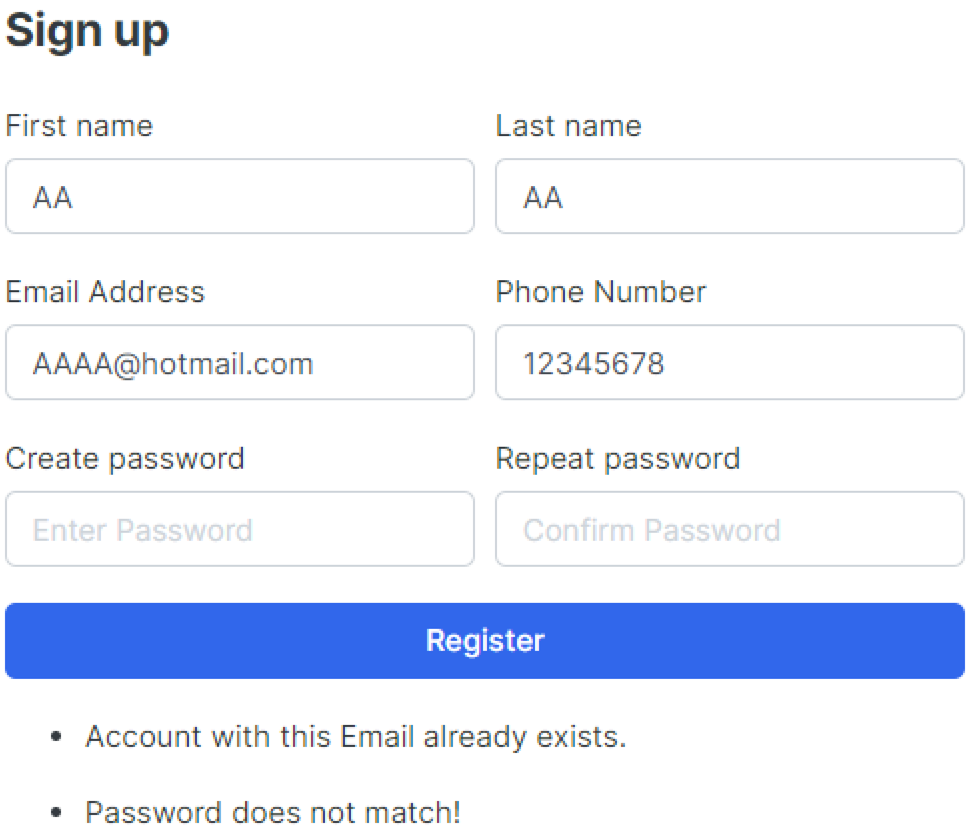
Create password: Enter Password

Repeat password: Confirm Password

- __all__
    - Password does not match!
- Password does not match!

234. Edit code in the register.html under accounts folder under templates folder

```
                        </div> <!-- form-group// -->
                      {{ form.email.errors }}
                      {{ form.non_field_errors }}
```



Sign up

First name: AA

Last name: AA

Email Address: AAAA@hotmail.com

Phone Number: 12345678

Create password: Enter Password

Repeat password: Confirm Password

- Account with this Email already exists.
- Password does not match!

235. Add lines in the settings.py under the online folder

```
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
from django.contrib.messages import constants as messages
MESSAGE_TAGS = {
    messages.ERROR: 'danger',
}
```

236. Create and add lines alerts.html in the includes folder in the templates folder

```
{% if messages %}
<ul class="messages">
    {% for message in messages %}
    <li{% if message.tags %} class="{{ message.tags }}" {% endif %}>
        {% if message.level == DEFAULT_MESSAGE_LEVELS>ERROR %} IMportant: {% endif %}
        {{ message }}
    </li>
    {% endfor %}
</ul>
{% endif %}
```

237. Edit code in the register.html under accounts folder under templates folder

Old

```
      <article class="card-body">
                    <header class="mb-4"><h4 class="card-title">Sign up</h4></header>
```

New

```
      <article class="card-body">
```

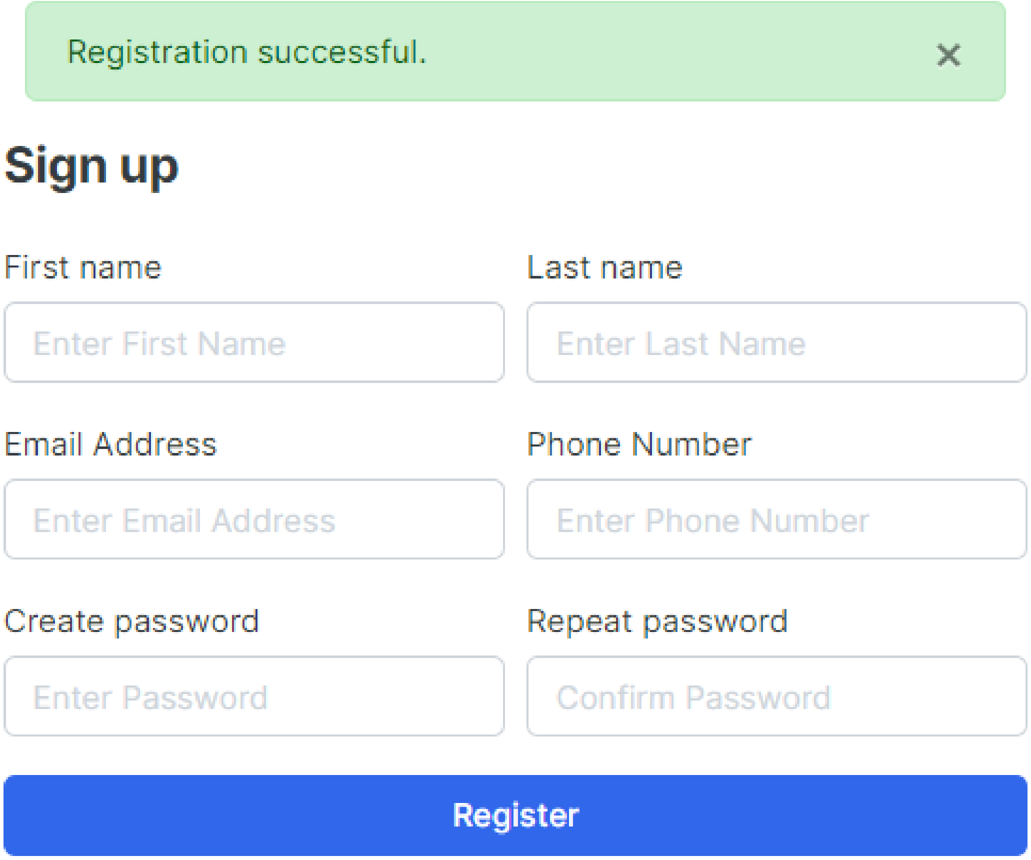
```
    {% include 'includes/alerts.html' %}
                <header class="mb-4"><h4 class="card-title">Sign up</h4></header>
```

238. Edit codes in the views.py under accounts folder

```
from django.shortcuts import render, redirect
from .forms import RegistrationForm
from .models import Account
from django.contrib import messages
&
            user.save()
            messages.success(request, 'Registration successful.')
            return redirect('register')
```

239. Edit lines alerts.html in the includes folder in the templates folder

```
{% if messages %}
  {% for message in messages %}
  <div id="message" class="container">
  <div {% if message.tags %} class="alert alert-{{ message.tags }}" {% endif %} role="alert">
    <button type="button" class="close" data-dismiss="alert"><span aria-hidden="true">&times;</span></button>
      {% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %} Important: {% endif %}
      {{ message }}
  </div>
  </div>
  {% endfor %}
{% endif %}
```



Registration successful.

240. Add 3 lines in the after last lines in script.js in the JS folder in the static folder under online folder

```
setTimeout(function(){
   $('#message').fadeOut('slow')
},4000)
```

Registration successful message will auto disappear in 4 seconds

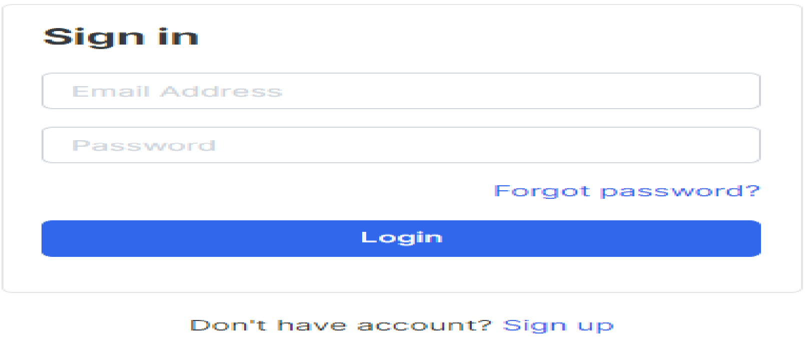
241. Copy the codes from signin.html under sample project folder to login.html under accounts folder under templates folder (login page)

Copy

```
<!-- ========================= SECTION CONTENT ========================= -->
…
<!-- ========================= SECTION CONTENT END// ========================= -->
```

Paste

```
{% block content %}
<!-- ========================= SECTION CONTENT ========================= -->
<section class="section-conten padding-y" style="min-height:84vh">
<!-- ============================ COMPONENT LOGIN   ================================= -->
    <div class="card mx-auto" style="max-width: 380px; margin-top:100px;">
      <div class="card-body">
      <h4 class="card-title mb-4">Sign in</h4>
      <form>
          <div class="form-group">
                                 <input type="email" class="form-control" placeholder="Email Address" >
          </div> <!-- form-group// -->
          <div class="form-group">
                                <input type="password" class="form-control" placeholder="Password" >
          </div> <!-- form-group// -->

          <div class="form-group">
                  <a href="#" class="float-right">Forgot password?</a>

          </div> <!-- form-group form-check .// -->
          <div class="form-group">
              <button type="submit" class="btn btn-primary btn-block"> Login  </button>
          </div> <!-- form-group// -->
      </form>
      </div> <!-- card-body.// -->
    </div> <!-- card .// -->

     <p class="text-center mt-4">Don't have account? <a href="{% url 'register' %}">Sign up</a></p>
     <br><br>
<!-- ============================ COMPONENT LOGIN  END.// ================================= -->
</section>
<!-- ========================= SECTION CONTENT END// ========================= -->
{% endblock %}
```



242. Edit codes in the register.html under accounts folder under templates holder

Old

```
  <p class="text-center mt-4">Have an account? <a href="">Log In</a></p>
```

New

```
  <p class="text-center mt-4">Have an account? <a href="{% url ‘login’ %}">Log In</a></p>
```

243. Edit codes in login.html in the accounts folder in the templates folder

Old

```
    <h4 class="card-title mb-4">Sign in</h4>
    <form>
      <div class="form-group">
```

&

```
<input type="email" class="form-control" placeholder="Email Address" >
```

&

```
<input type="password" class="form-control" placeholder="Password" >
```

New

```
    <form action="{% url 'login' %}" method="POST">
      {% csrf_token %}
      <div class="form-group">
```

&

```
<input type="email" class="form-control" placeholder="Email Address" name=”email”>
```

&

```
<input type="password" class="form-control" placeholder="Password" name=”password”>
```

244. Edit views.py under accounts folder

Old

```
def login(request):
  return render(request, 'accounts/login.html')
```

New

```
from django.contrib import messages, auth
```

&

```
def login(request):
  if request.method == 'POST':
    email = request.POST['email']
    password = request.POST['password']
    user = auth.authenticate(email=email, password=password)
    if user is not None:
      auth.login(request, user)
      # message.success(request, 'You are now logged in.')
      return redirect('home')
    else:
      messages.error(request, 'Invalid login credentials')
      return redirect(‘login’)
  return render(request, 'accounts/login.html')
```

245. Add lines in the login.html under accounts folder under templates folder

```
<h4 class="card-title mb-4">Sign in</h4>
{% include 'includes/alerts.html' %}
<form action="{% url 'login' %}" method="POST">
```

246. Edit code in alerts.html in the includes folder templates folder

Old

```
{% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %} Important: {% endif %}
```

New

```
{% if message.level == DEFAULT_MESSAGE_LEVELS.ERROR %} Error: {% endif %}
```

247. Edit navbar.html under includes folder in the templates folder

Old

```
<div class="d-flex justify-content-end mb-3 mb-lg-0">
        <div class="widget-header">
```

&

```
        </div>
        <a href="{% url 'cart' %}" class="widget-header pl-3 ml-3">
```

New

```
<div class="d-flex justify-content-end mb-3 mb-lg-0">
{% if user.id is None %}
        <div class="widget-header">
```

&

```
        </div>
        {% else %}
        <div class="widget-header">
                <small class="title text-muted">Welcome {{user.first_name}}!</small>
                <div>
                        <a href="#">Dashbaord</a> <span class="dark-transp"> | </span>
                        <a href="{% url 'logout' %}"> Logout</a>
                </div>
        </div>
        {% endif %}
        <a href="{% url 'cart' %}" class="widget-header pl-3 ml-3">
```

248. Add and Edit views.py under accounts (log out button)

Add lines

```
from django.contrib.auth.decorators import login_required
```

&

Replace logout codes

```
@login_required(login_url = 'login')
def logout(request):
  auth.logout(request)
  messages.success(request, "You are logged out.")
  return redirect('login')
```

249. Add lines in the views.py in the account folder (Activation link with token for email verification upon user register)

250.	1
251.	1
252.	1
253.	1
254.	1
255.	1
256.	1
257.	1
258.	1
259.	1
260.	1
261.	1
262.	1
263.	1
264.	1
265.	1
266.	1
267.	1
268.	1
269.	1
270.	1
271.	1
272.	1
273.	1
274.	1
275.	1
276.	1
277.	