Each time log in Step by Step to code:

1. Start env

cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts

activate

1. Display the packages install in the env

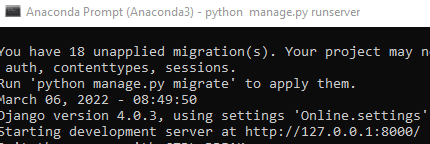
pip freeze

1. Run server to start coding

cd C:\Users\angel\OneDrive\Desktop\Python Django

python manage.py runserver

http://127.0.0.1:8000/



Website development Step by Step:

1. Find out the path name of the web development folder

cd C:\Users\angel\OneDrive\Desktop\Python Django

1. Installing virtualenv

py –m pip install –-user virtualenv

1. Creating a virtual environment folder

py –m venv env

1. Pointing to the activate the env folder

cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts

activate

1. Install Django package

pip install django

1. Display the packages install in the env

pip freeze

1. create project folder

cd C:\Users\angel\OneDrive\Desktop\Python Django

django-admin startproject Online .

1. Python start web site server

python manage.py runserver

http://127.0.0.1:8000/

1. Create a views.py in the online folder together with urls.py
2. 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'),

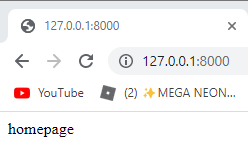
]

Views.py

from django.http import HttpResponse

def home(request):

return HttpResponse('homepage')



1. Create templates folder in the project folder
2. Create home.html in the templates folder

<h2>This is templates</h2>

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

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': ['templates'],

1. Adjust the view filer from HttpResponse to Render

#from django.http import HttpResponse

from django.shortcuts import render

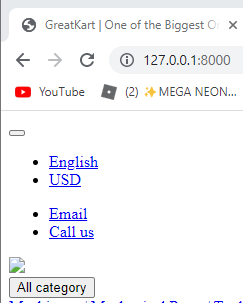
def home(request):

#return HttpResponse('homepage')

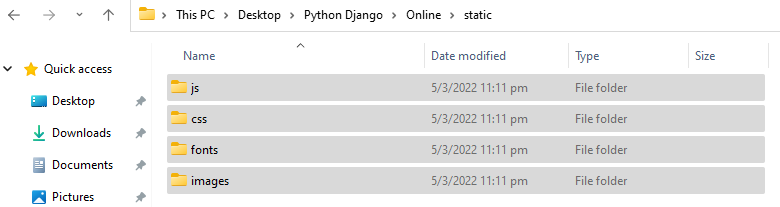
return render(request, 'home.html')



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



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



1. 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',

]

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

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

{% load static %}

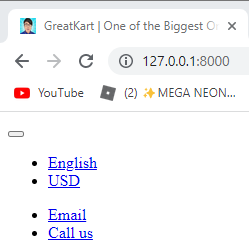
<!DOCTYPE HTML>

<html lang="en">

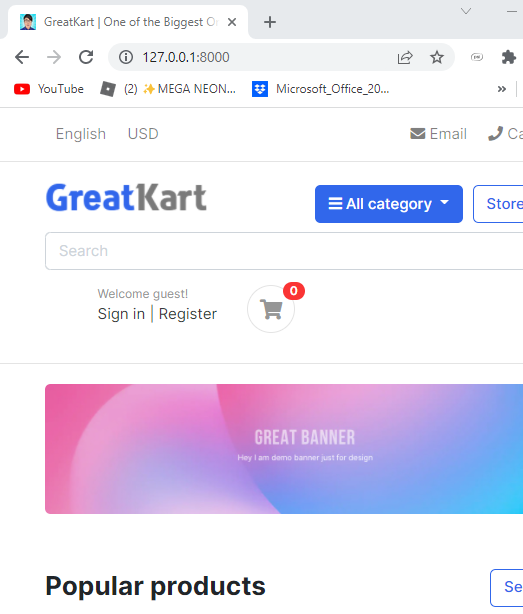
Replace old code with new codes

<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">



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



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

1st line

{% load static %}

<!DOCTYPE HTML>

Last few lines

</header> <!-- section-header.// -->

{% block contents %}

<!-- content -->

{% endblock %}

1. 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 %}

1. Create includes folder in the templates folder
2. Create footer.html & navbar.html in the includes folder
3. 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 // ========================= -->

1. Copy header sets of codes from base.html to navbar.html
2. 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' %}

1. Add 1 line at navbar.html

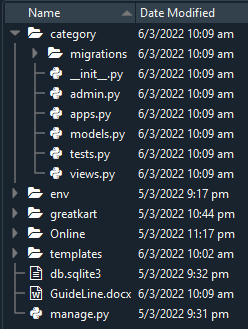
{% load static %}

1. Add 1 line in the foorer.html

{% load static %}

1. Create an category folder in the online folder

python manage.py startapp category



1. 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',

]

1. 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.

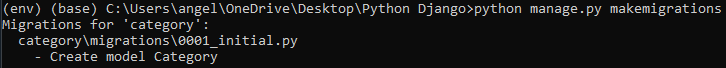
admin.site.register(Category)

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

Pip install pillow

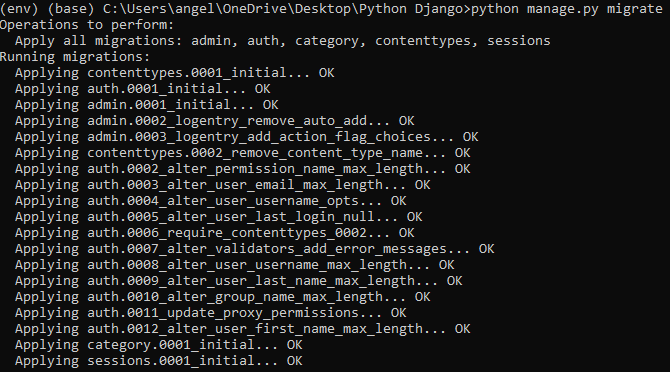
1. Type command in the Anaconda Prompt (Anaconda3) to create migrations folder

python manage.py makemigrations



1. Python run the migrate files 0001\_initial.py

python manage.py migrate



1. Create a super user

python manage.py createsuperuser

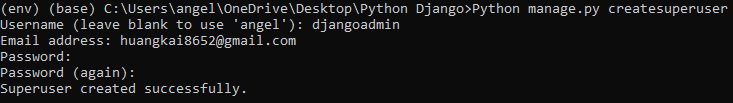
1. Create ID and Password for super user

Username: djangoadmin

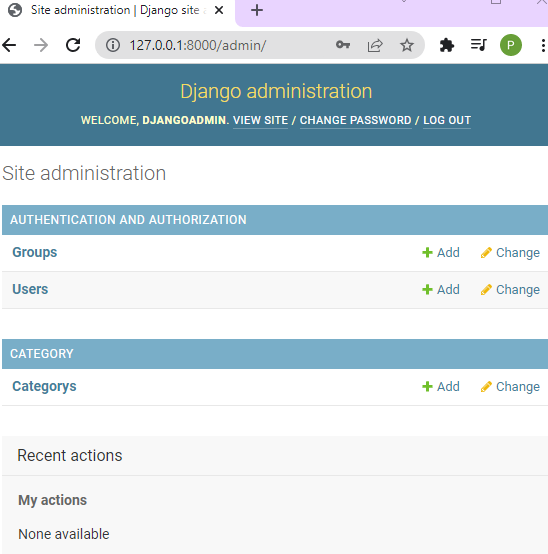
Email: huangkai8652@gamil.com

Password: !q2w3e4R

Password: !q2w3e4R



Django Admin Page (Logged In)



1. 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):

1. Run migration command again

python manage.py makemigrations



1. Create account folder for email log in

python manage.py startapp accounts

1. 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',

]

1. 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

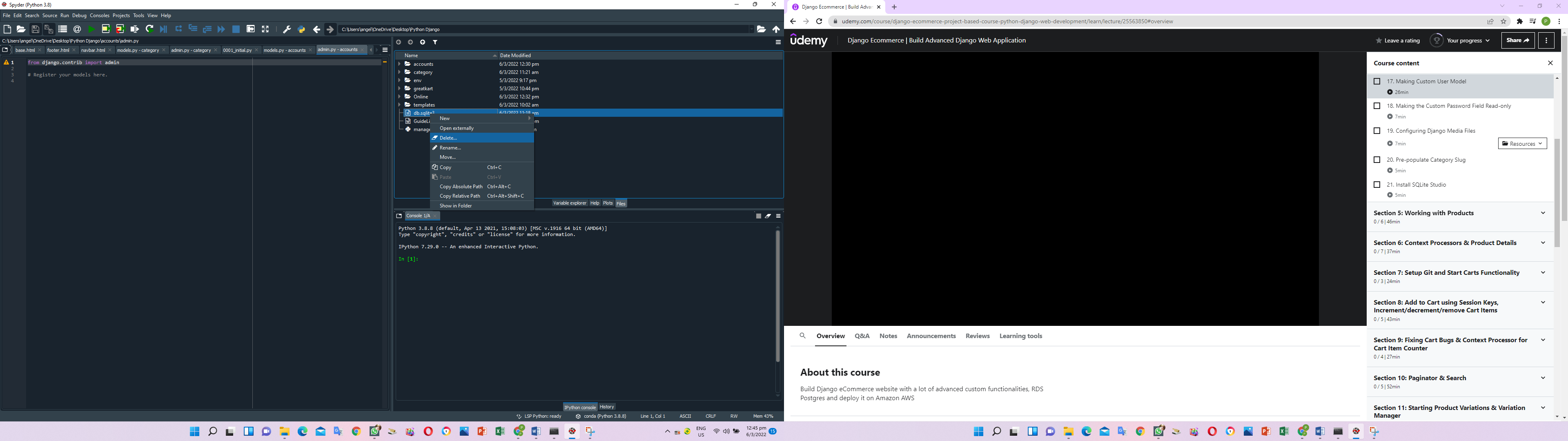
1. Add links in the settings.py file

WSGI\_APPLICATION = 'Online.wsgi.application'

AUTH\_USER\_MODEL = 'accounts.Account'

# Database

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



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

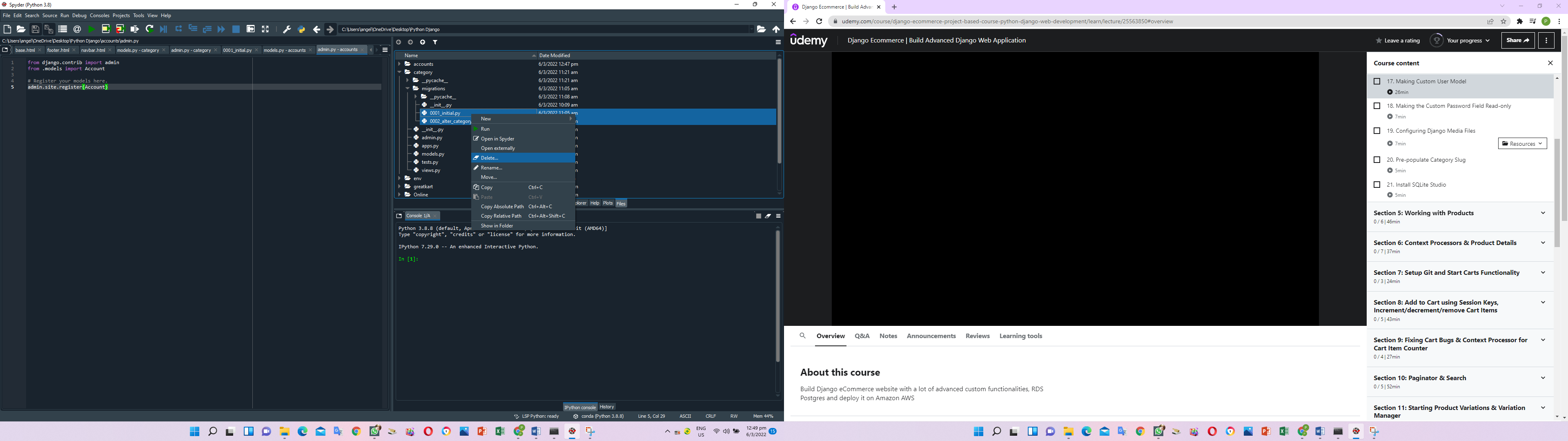
from django.contrib import admin

from .models import Account

# Register your models here.

admin.site.register(Account)

1. Delete the 0001\_initial.py & 0002\_alter\_category\_options.py in the migrations folder.

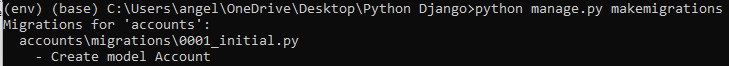


1. Create a db.sqlite3 in the online folder

python manage.py runserver

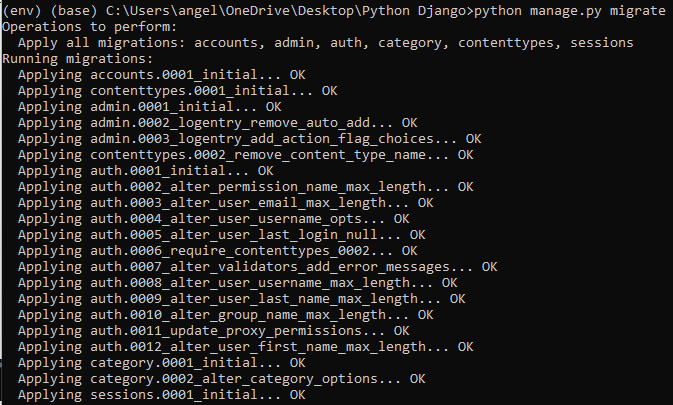
1. Migration folder again for the 0001\_inital.py

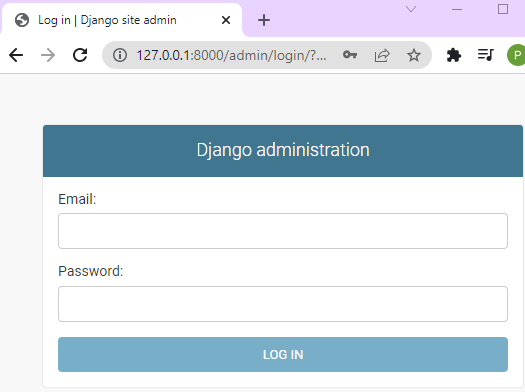
python manage.py makemigrations



1. Migrate the folder

python manage.py migrate





1. Create super user

Email: huangkai8652@gmail.com

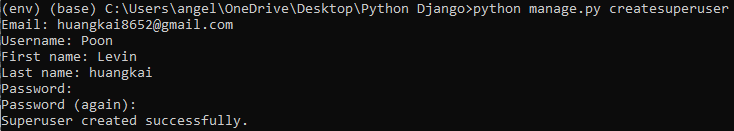
Username: Poon

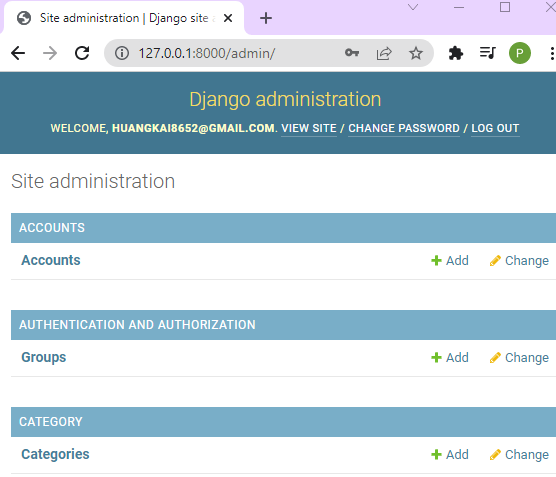
First name: Levin

Last name: huangkai

Password: !q2w3e4R

Password (again): !q2w3e4R





User able to log in thru email instead of username

1. 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')

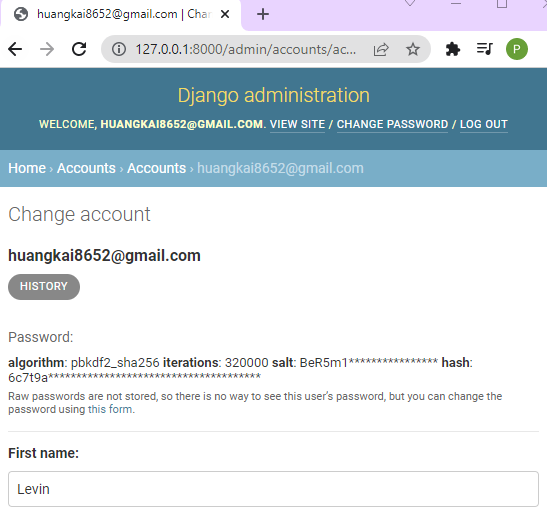
ordering = ('-date\_joined',)

filter\_horizontal = ()

list\_filter = ()

fieldsets = ()

admin.site.register(Account, AccountAdmin)



1. 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'

1. 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)

1. 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

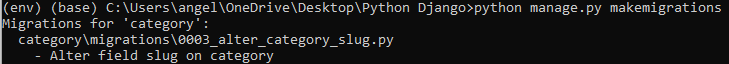
1. Slug name will auto copy from Category name

slug = models.CharField(max\_length=100, unique=True) change to =>

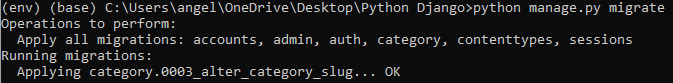
slug = models.SlugField(max\_length=100, unique=True)

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

python manage.py makemigrations



python manage.py migrate



1. 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)

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

Category name: T shirt

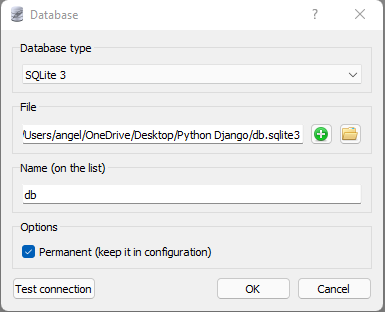
Slug: t-shirt

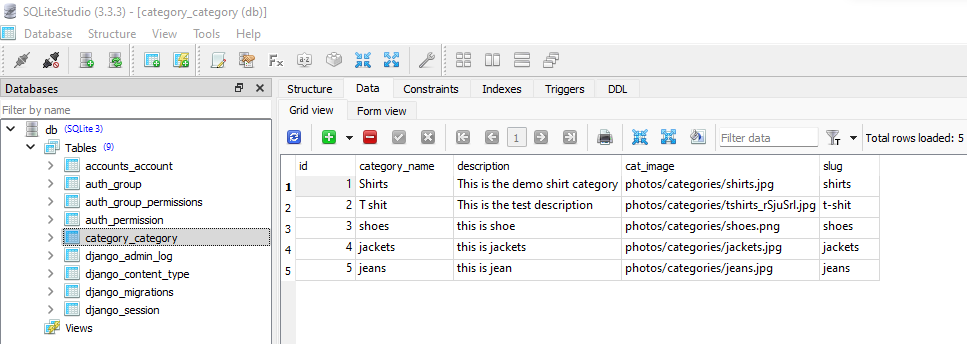
Descriptions: This is the test description

Cat Image: tshirts.jpg

Add the rest 4 category into the Django admin

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





1. Crate store folder in the online folder

Python manage.py startapp store

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

'category',

'accounts',

'store',

]

1. 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

1. 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')

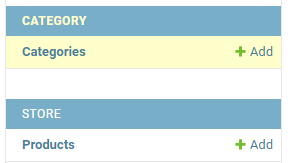
prepopluated\_fields = {'slug': ('product\_name',)}

admin.site.register(Product, ProductAdmin)

python manage.py makemigrations

python manage.py migrate

User able to see product appear on the admin page



1. Add all products under each category

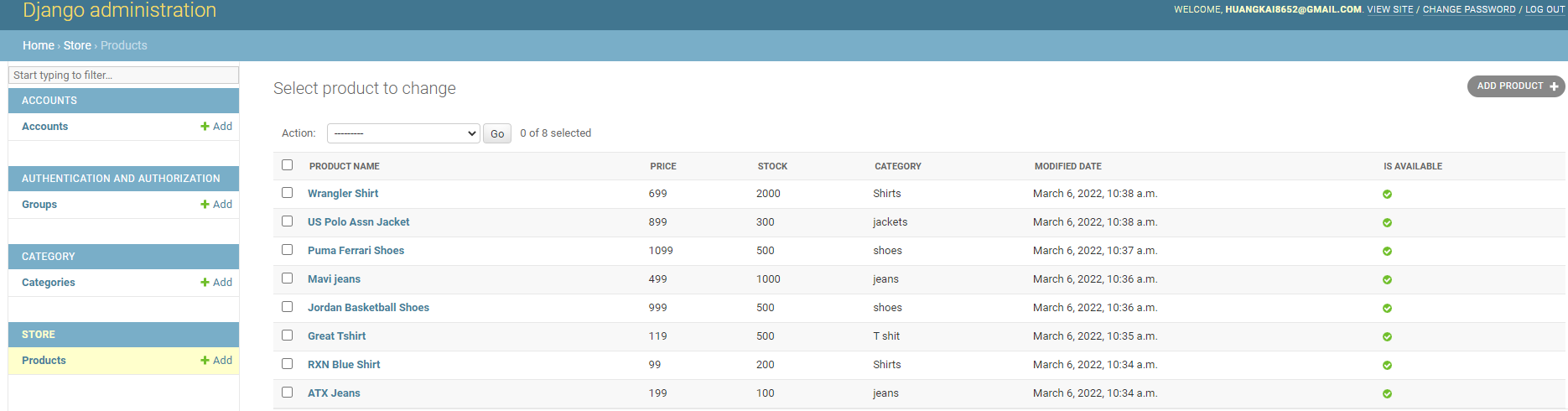
Product name:

Slug:

Description:

Price:

Images:



1. 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)

1. 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// ========================= -->

1. 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.// -->

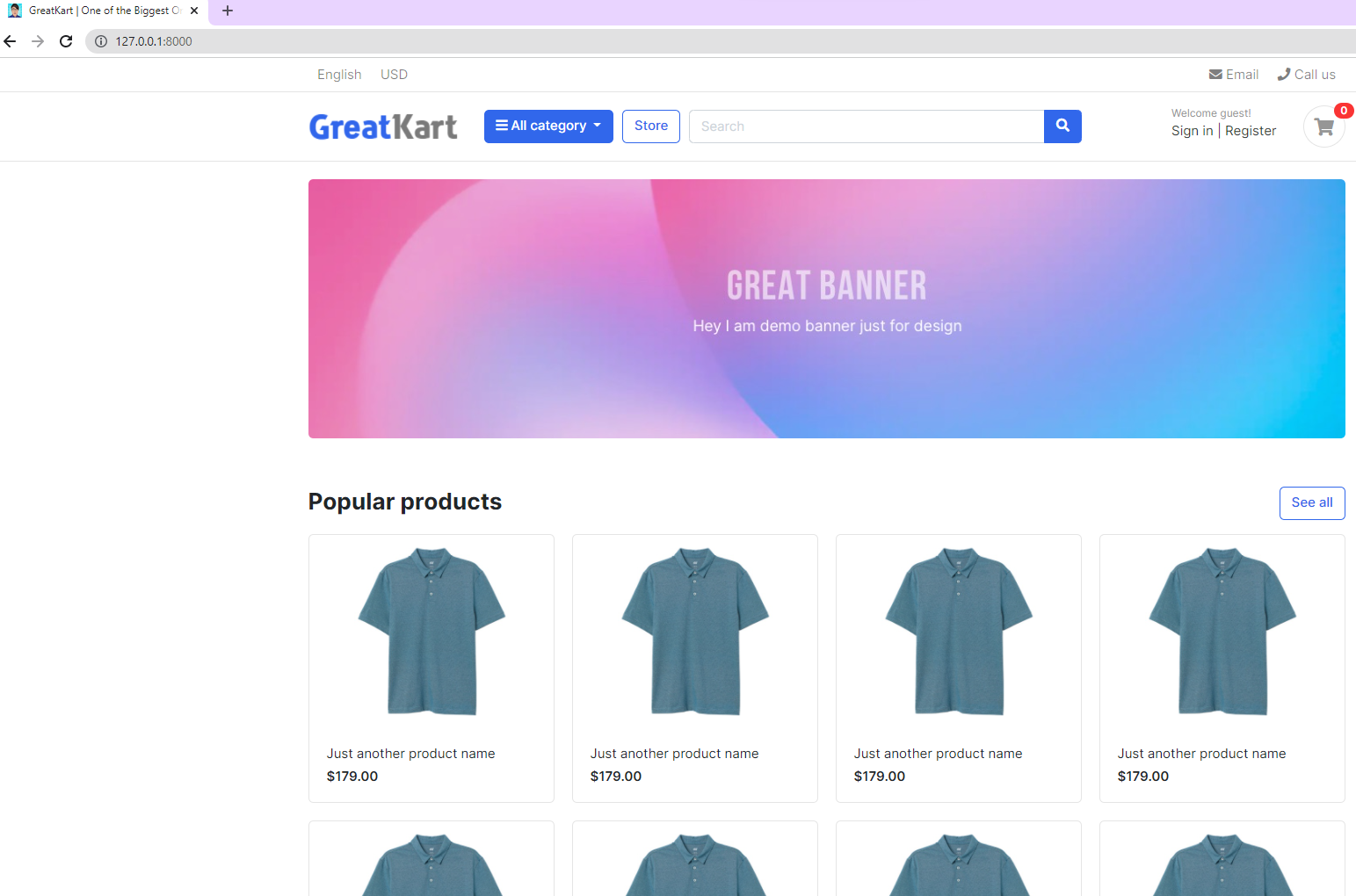
</figcaption>

</div>

</div> <!-- col.// -->

{% endfor %}

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



1. 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// ========================= -->

1. 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'),

]

1. 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)

1. 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')

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

<!-- ========================= SECTION ========================= -->

<section class="section-name padding-y-sm">

<div class="container">

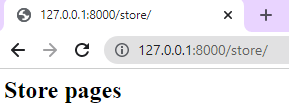
<header class="section-heading">

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

<h3 class="section-title">Popular products</h3>

</header><!-- sect-heading -->

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

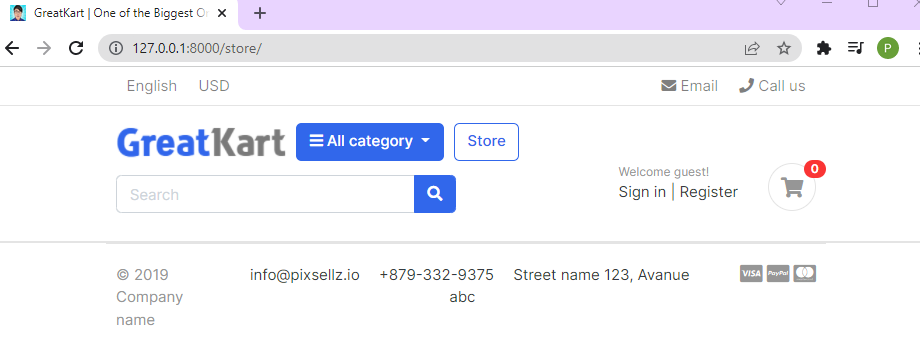


1. Replace codes in the store.html under the store folder under templates folder

{% extends 'base.html' %}

{% block content %}

{% endblock %}



1. 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// ========================= -->

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

<!-- navbar -->

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

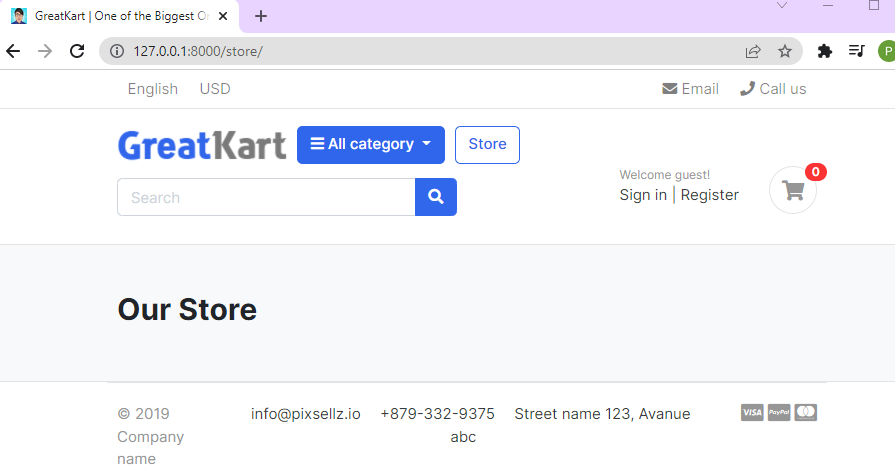
{% block content %}

<!-- content -->

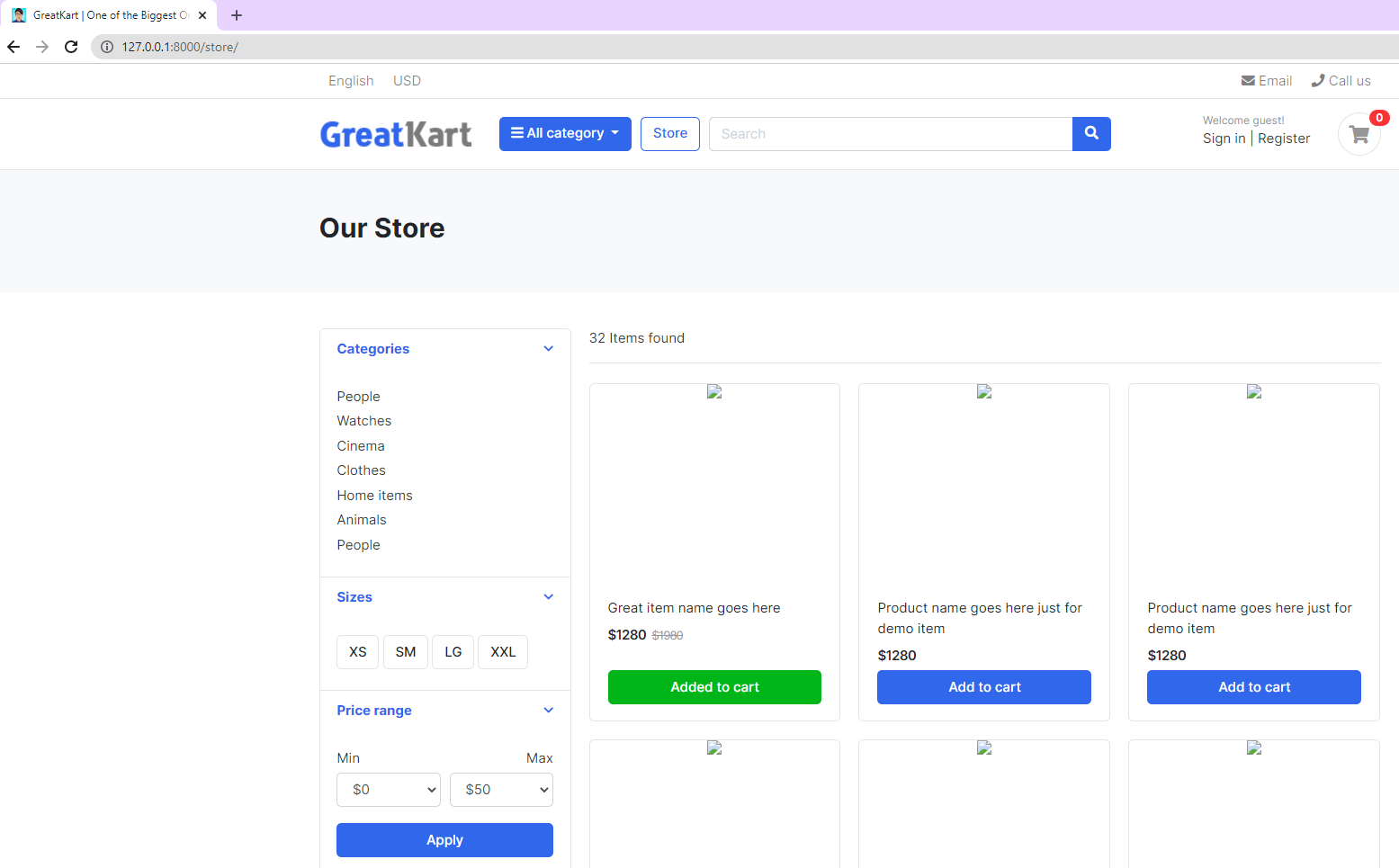
{% endblock %}

<!-- footer -->

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



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



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

{% extends 'base.html' %}

{% load static %}

{% block content %}{% load static %}

1. Replace image code

<img src="images/items/1.jpg">

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

1. 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)

1. 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.// -->

1. 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.// -->

1. 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>

1. Add line to adjust the prices

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

1. Delete 1 line

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

1. 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>

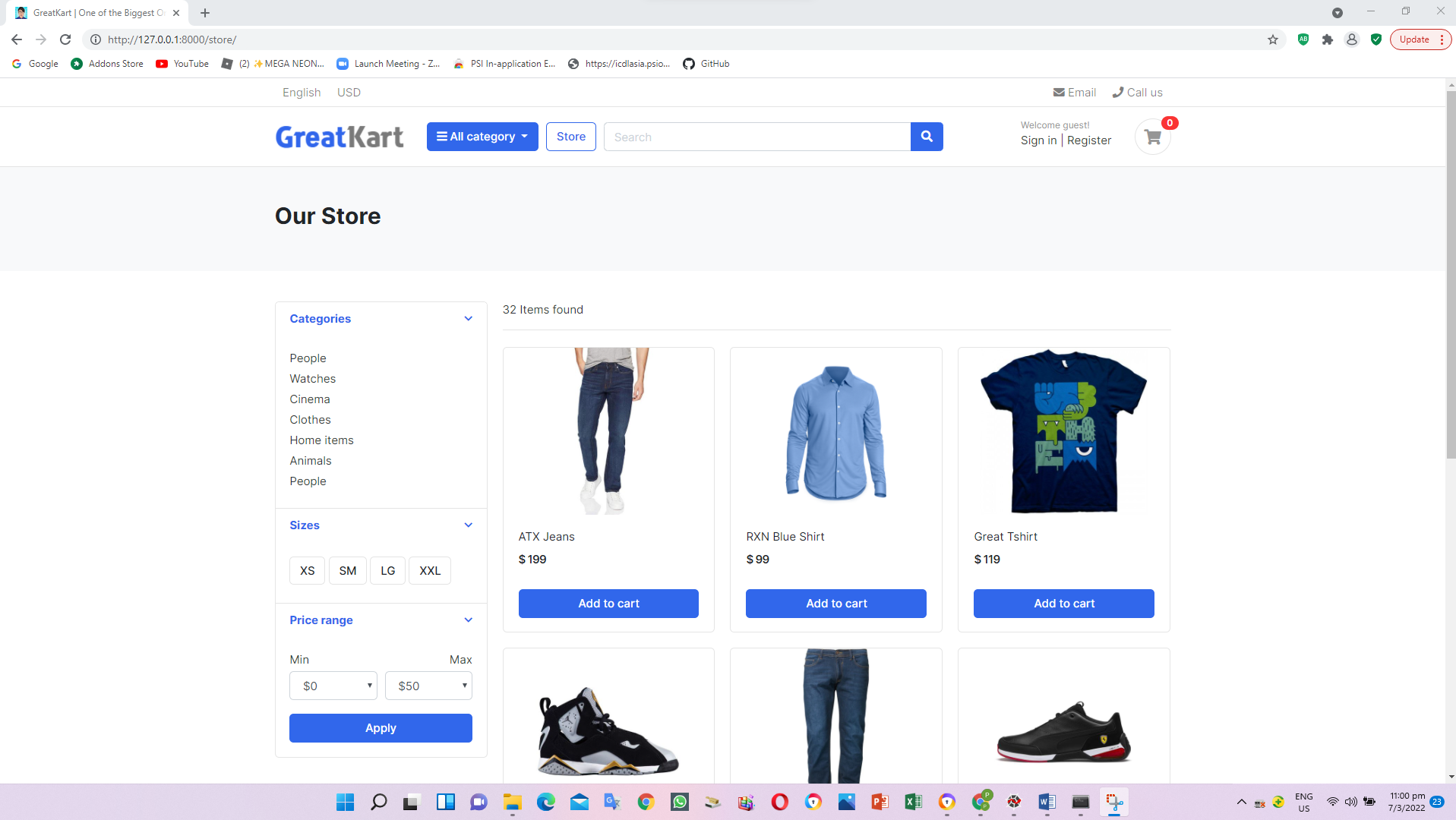
</figcaption>

</figure>

</div> <!-- col.// -->

{% endfor %}

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



1. 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,

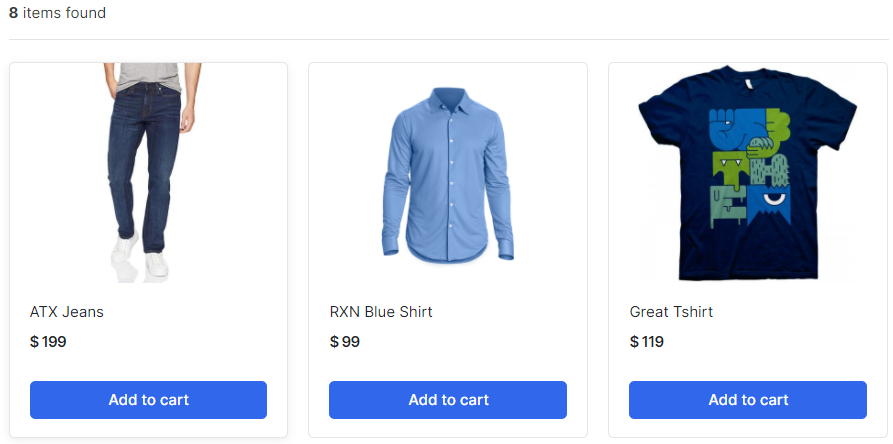
}

return render(request, 'store/store.html', context)

1. 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>



1. 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'),

]

1. 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()

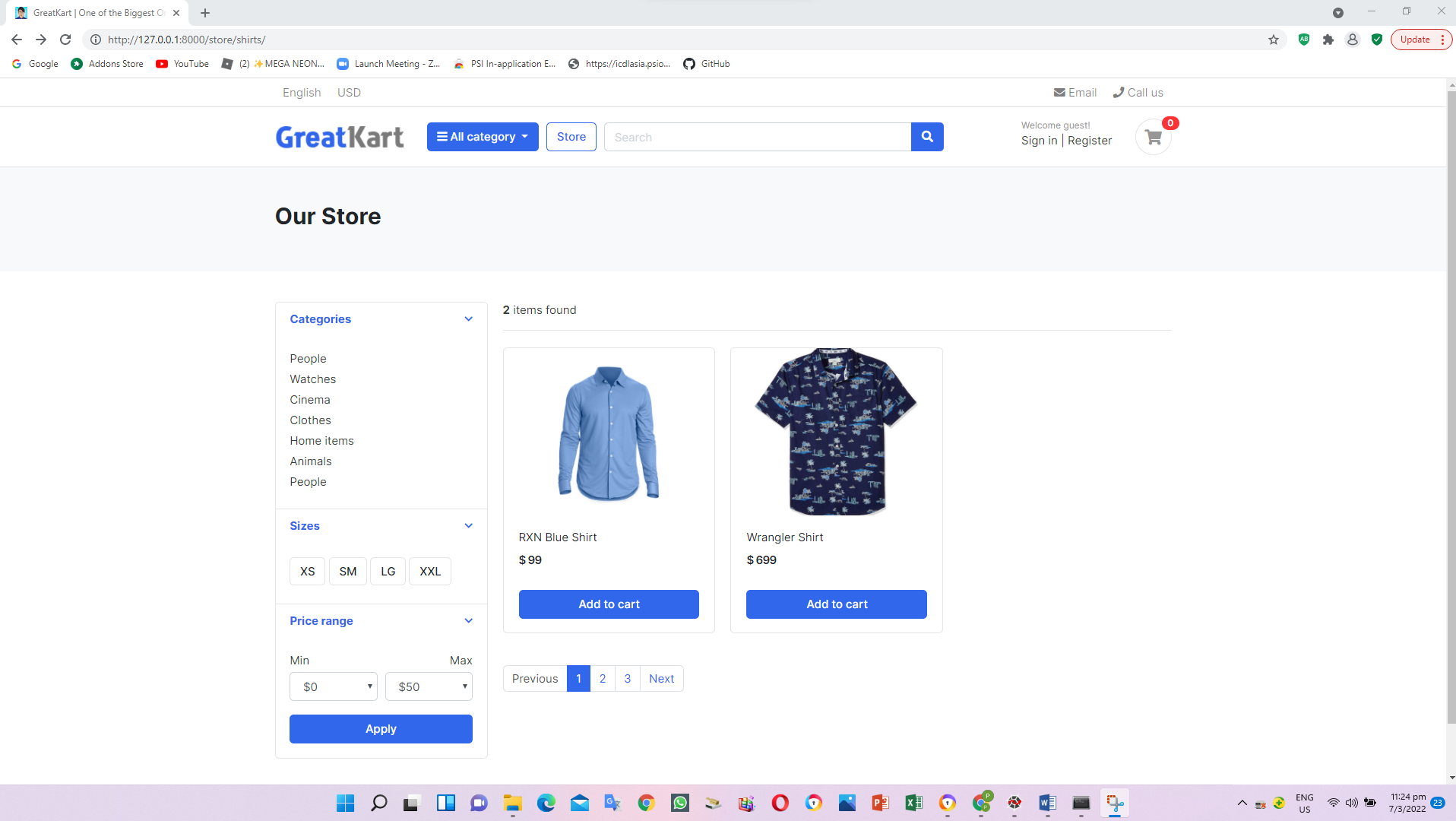
context = {

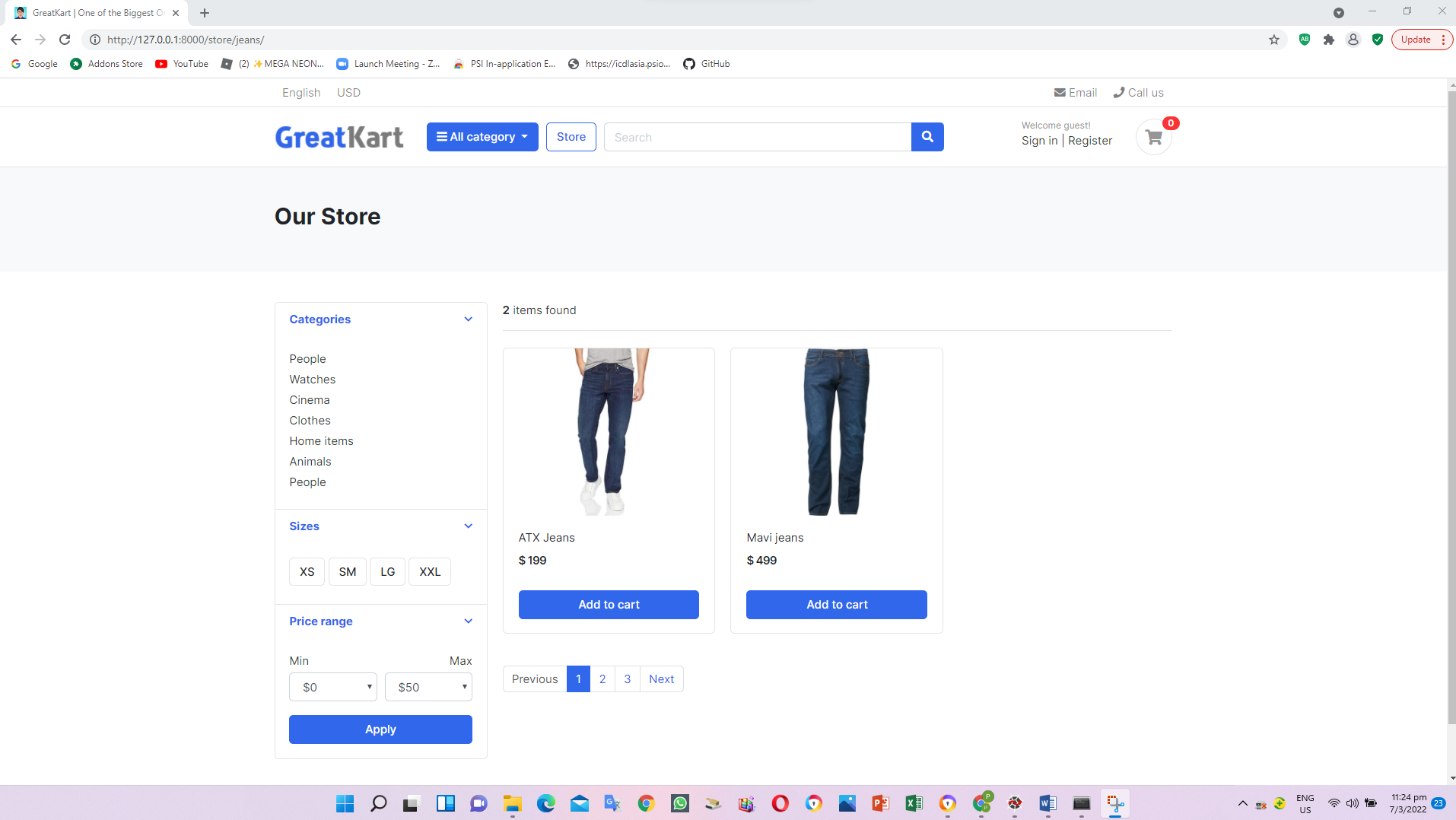
'products': products,

'product\_count': product\_count,

}

return render(request, 'store/store.html', context)





1. 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)

1. 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',

],

},

},

]

1. 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">

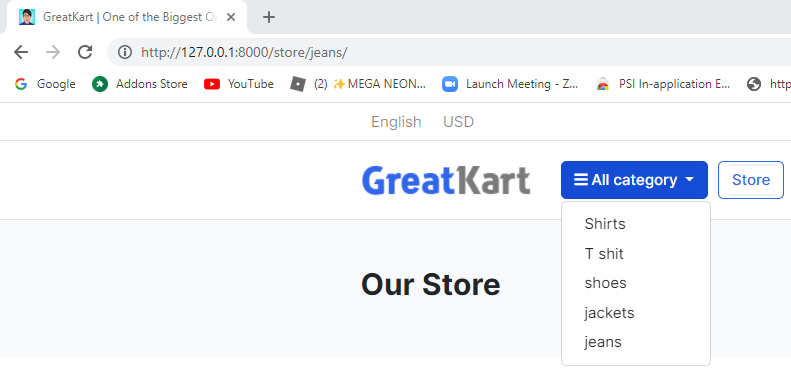
{% for category in links %}

<a class="dropdown-item" href="{{ category.get\_url }}">{{ category.category\_name}}</a>

{% endfor%}

</div>

</div> <!-- category-wrap.// -->



1. 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):

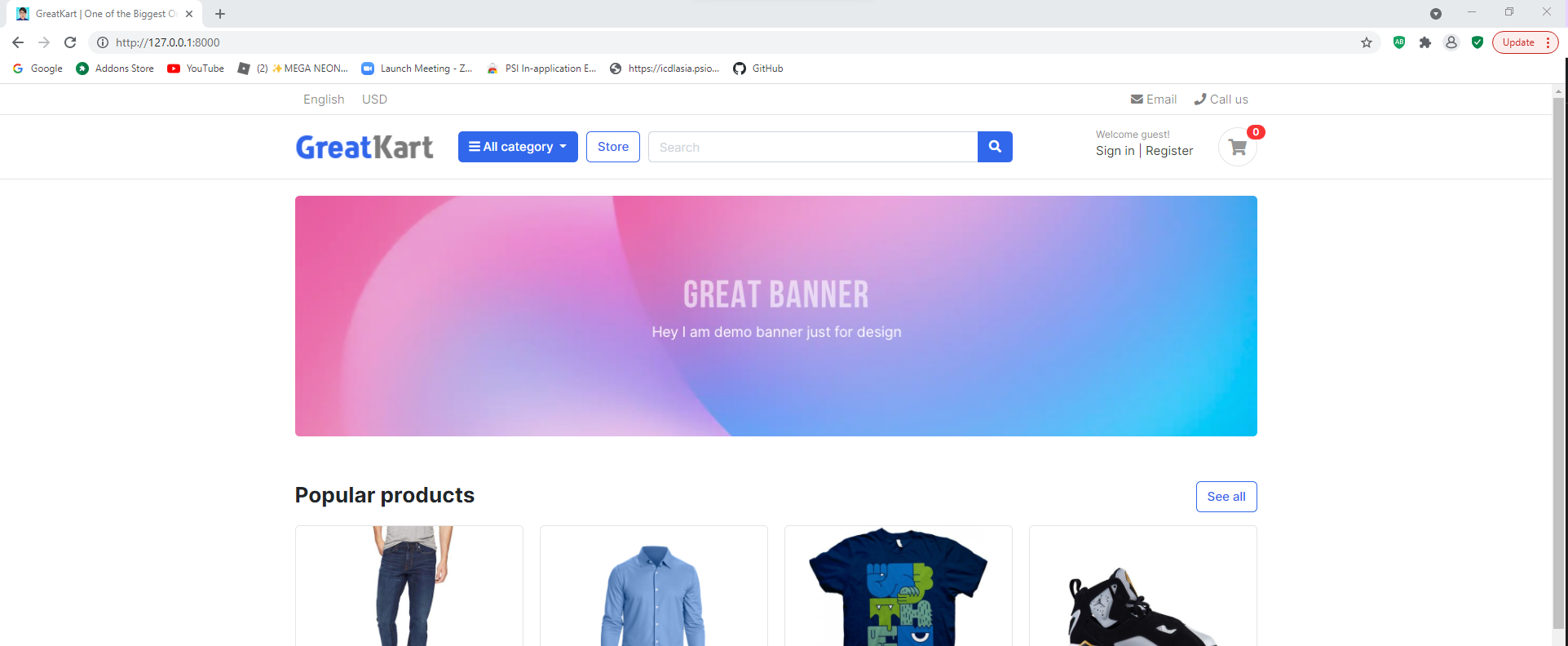
return reverse('products\_by\_category', args=[self.slug])

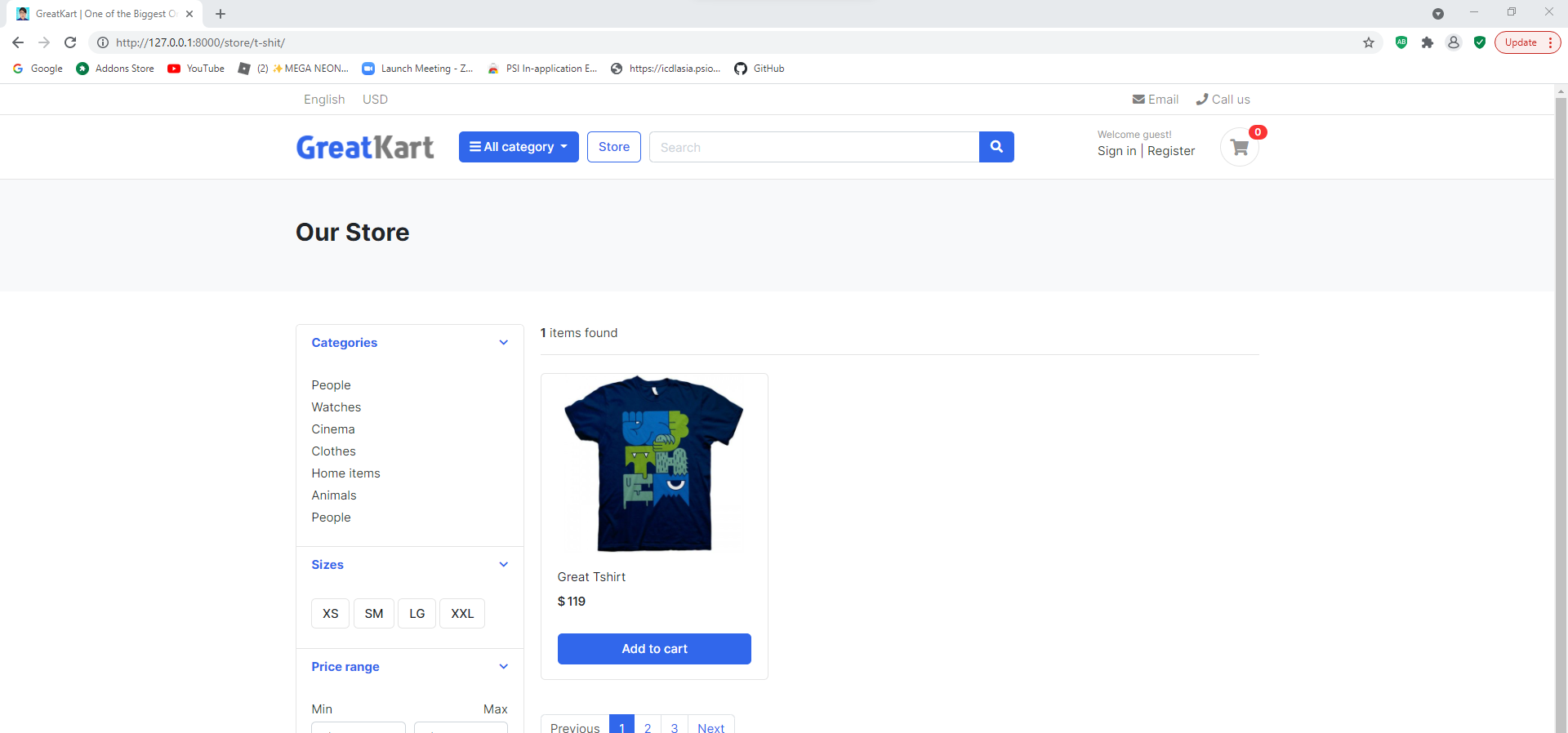
def \_\_str\_\_(self):

return self.category\_name

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

{% block content %}





1. 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.// -->



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

<div class="dropdown-menu">

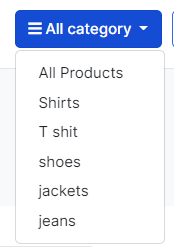
<a class="dropdown-item" href="{% url 'store' %}">All Products</a>

{% for category in links %}

<a class="dropdown-item" href="{{ category.get\_url }}">{{ category.category\_name}}</a>

{% endfor %}

</div>



1. 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'),

]

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

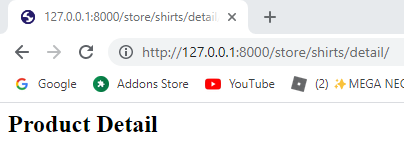
return render(request, 'store/store.html', context)

def product\_detail(request,category\_slug, product\_slug):

return render(request, 'store/product\_detail.html')

1. Create product\_details.html in the store folder in the templates folder

<h2>Product Detail</h2>

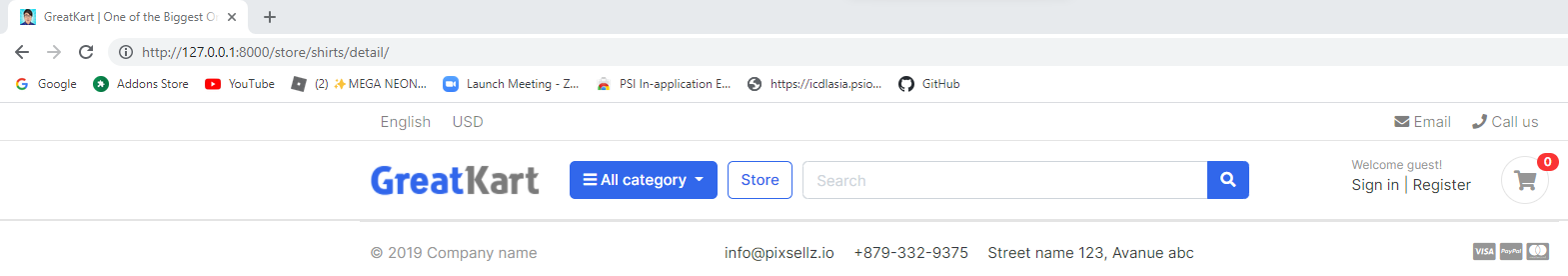


1. Replace the codes in the product\_details.html in the store folder in the templates folder

{% extends 'base.html' %}

{% block content %}

{% endblock %}



1. 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// ========================= -->

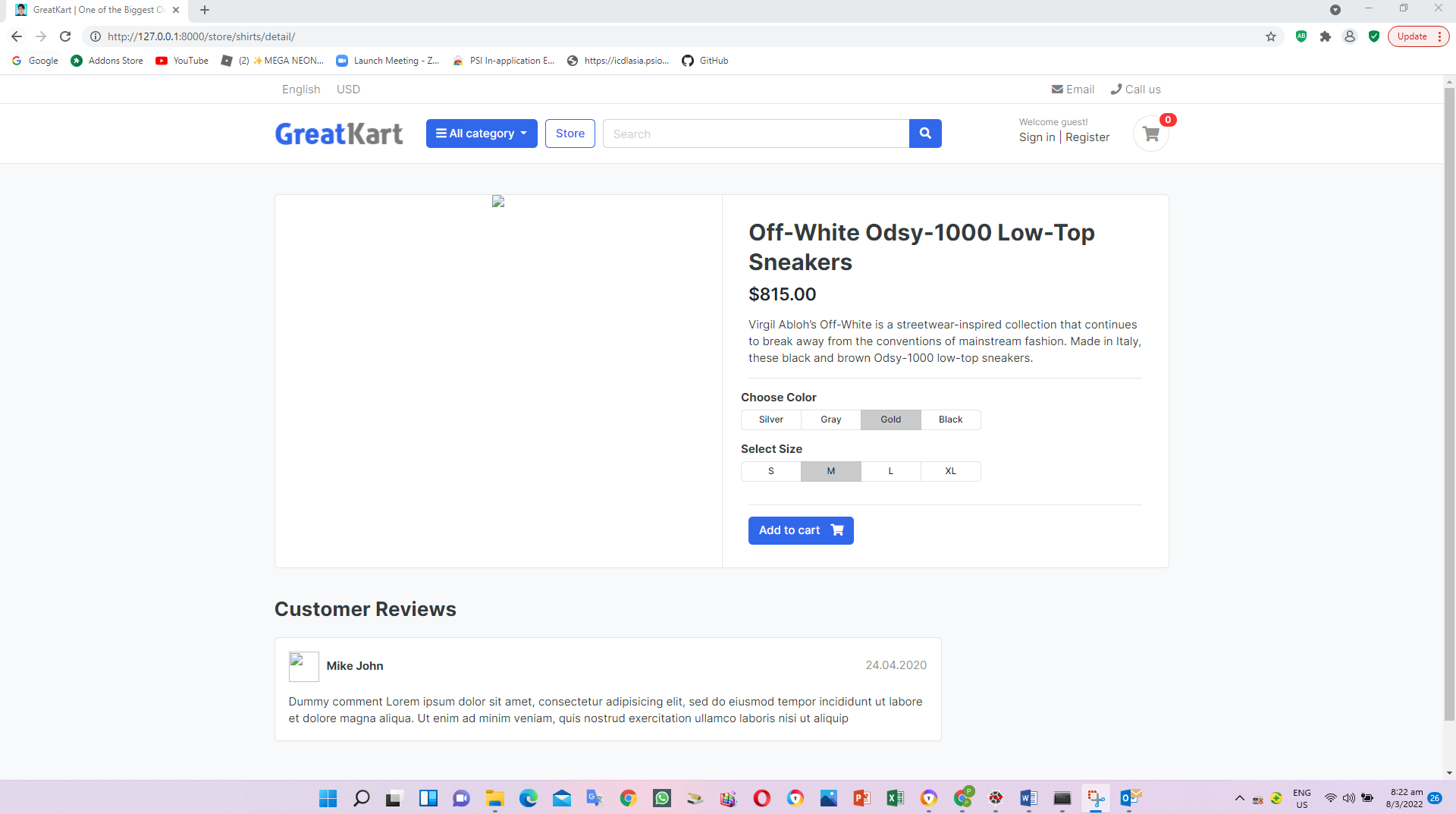
Paste

{% extends 'base.html' %}

{% block content %}

…

{% endblock %}



1. 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">

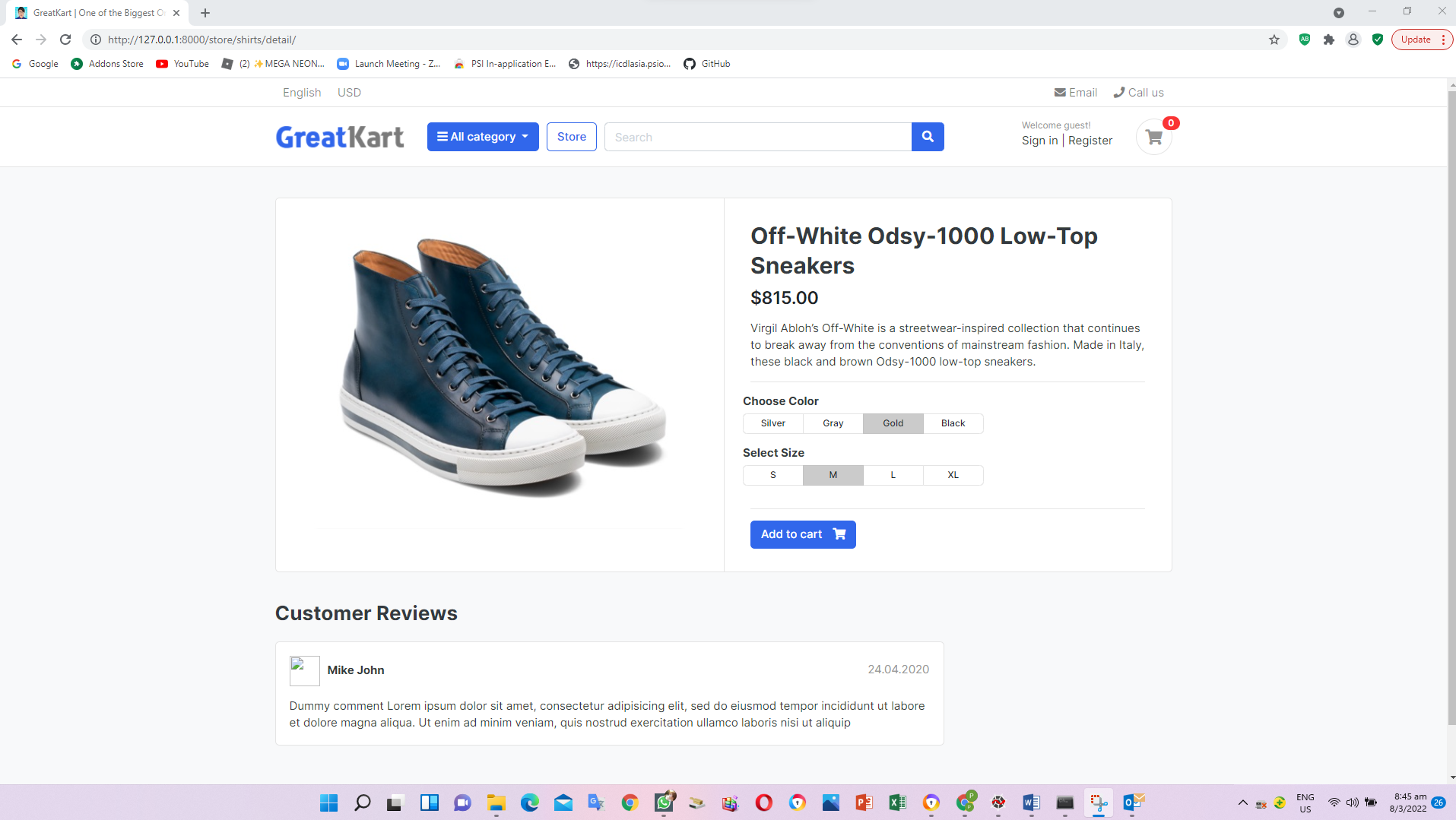
<div class="row no-gutters">

<aside class="col-md-6">

<article class="gallery-wrap">

<div class="img-big-wrap">

<a href="#"><img src="{% static './images/items/12.jpg' %}"></a>



1. Adjust codes in the product\_details.html in the store folder in the templates folder

<article class="box mb-3">

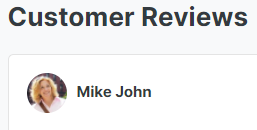
<div class="icontext w-100">

<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>



1. 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)

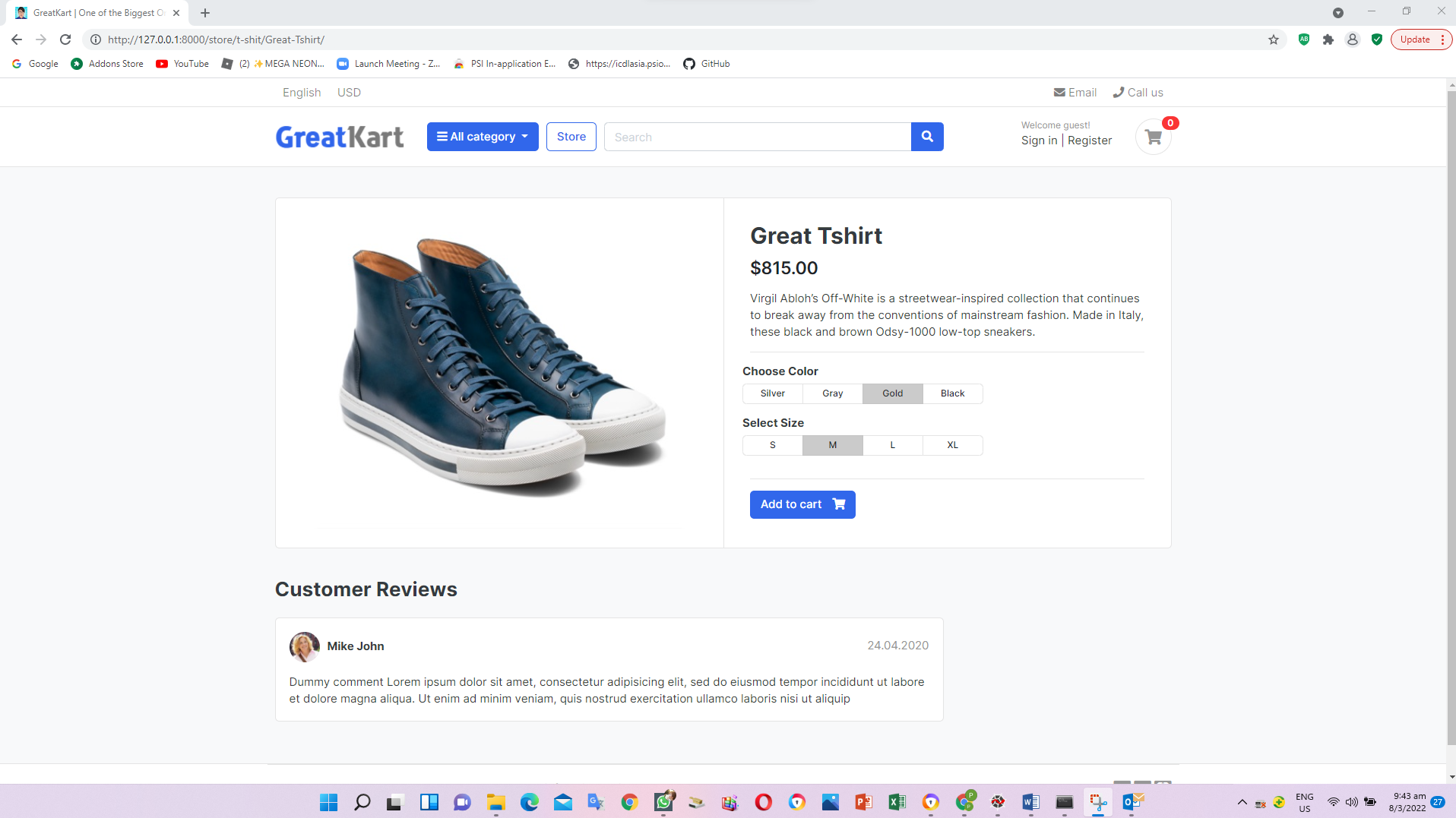
1. Adjust codes in the product\_details.html in the store folder in the templates folder

<article class="content-body">

<h2 class="title">{{ single\_product.product\_name }}</h2>

<div class="mb-3">

<var class="price h4">$815.00</var>



1. 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>

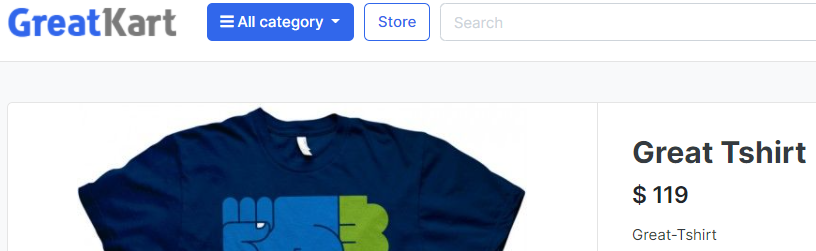
<div class="mb-3">

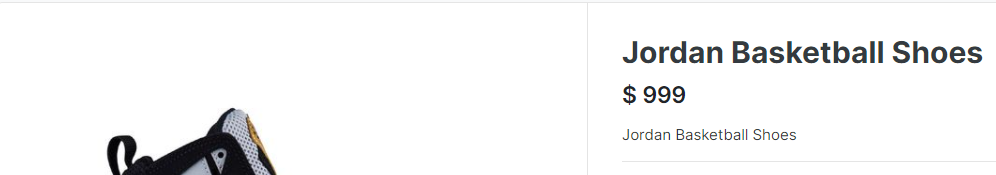
<var class="price h4">$ {{ single\_product.price }}</var>

</div>

<p>{{ single\_product.description }}</p>

<hr>





1. 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">

1. 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>

1. 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

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

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

1. Product available in the product\_details.html in the store folder in the templates folder

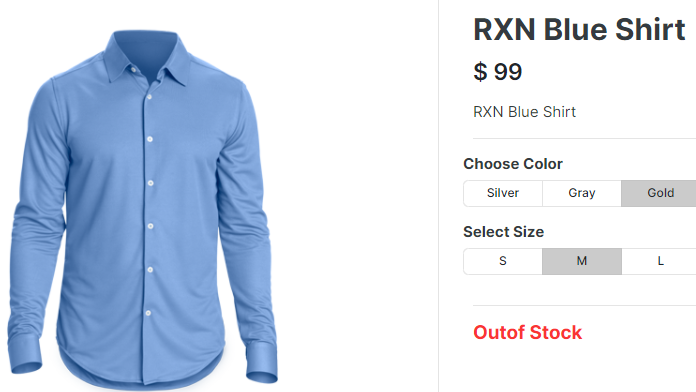
{% if single\_product.stock <= 0 %}

<h5 class="text-danger">Outof Stock</h5>

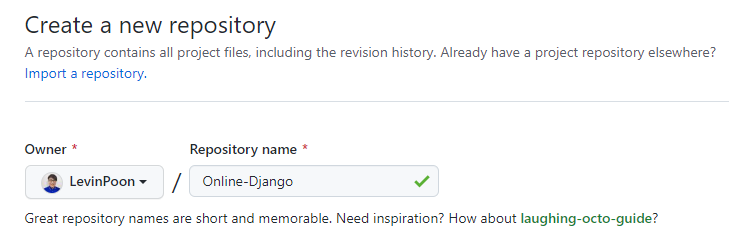
{% else %}

<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 %}



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



1. Enter command in the Anaconda Prompt (Anaconda3)

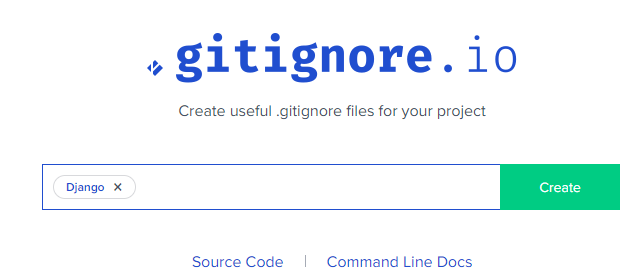
cd C:\Users\angel\OneDrive\Desktop\Python Django\env\Scripts

activate

cd C:\Users\angel\OneDrive\Desktop\Python Django

git init

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



1. Copy and paste code from website to .gitignore file
2. Add codes to git repository

Git add -A

1. 1
2. 1
3. 1
4. 1
5. 1
6. 1
7. 1
8. 1
9. 1
10. 1
11. 1
12. 1
13. 1
14. 1
15. 1
16. 1
17. 1
18. 1
19. 1
20. 1
21. 1
22. 1
23. 1
24. 1
25. 1
26. 1
27. 1
28. 1
29. 1
30. 1
31. 1
32. 1
33. 1
34. 1
35. 1
36. 1
37. 1
38. 1
39. 1
40. 1
41. 1
42. 1
43. 1
44. 1
45. 1
46. 1
47. 1
48. 1
49. 1
50. 1
51. 1